Research Program Co-chairs

Yaser Al-Onaizan  IBM T.J. Watson Research Center
Michel Simard  National Research Council Canada

Research Program Committee

Wilker Aziz  University of Sheffield
Graeme Blackwood  IBM Research
Marine Carpuat  National Research Council Canada
Daniel Cer  Google
Boxing Chen  National Research Council Canada
Colin Cherry  National Research Council Canada
David Chiang  University of Notre Dame
Trevor Cohn  University of Melbourne
Adrià de Gispert  SDL Research
Steve DeNeefe  SDL Language Weaver
Michael Denkowski  Carnegie Mellon University
Jacob Devlin  Raytheon BBN Technologies
Mike Dillinger  eBay Inc
Markus Dreyer  SDL Language Weaver
Marc Dymetman  Xerox Research Centre Europe
Ahmad Emami  IBM T J Watson Research Center
Marcello Federico  FBK
Minwei Feng  RWTH Aachen University
Mikel Forcada  Universität d’Alacant
George Foster  Google
Kevin Gimpel  Toyota Technological Institute at Chicago
Spence Green  Stanford University
Han Hassan  Microsoft Research
Xiaodong He  Microsoft Research
Yifan He  New York University
Kenneth Heafield  Bloomberg
Fei Huang  Temple University
Abe Ittycheriah  IBM
Philipp Koehn  Johns Hopkins University
Roland Kuhn  National Research Council Canada
Shankar Kumar  Google
Phillippe Langlais  Université de Montréal
Alon Lavie  Carnegie Mellon University
Gregor Leusch  APPTEK
Lemao Liu  The City University of New York
Qun Liu  Dublin City University
Wolfgang Macherey  Google
Klaus Macherey  Google
Saab Mansour  RWTH Aachen University
Daniel Marcu  ISI/USC
<table>
<thead>
<tr>
<th>Name</th>
<th>Affiliation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yuval Marton</td>
<td>Microsoft</td>
</tr>
<tr>
<td>Arne Mauser</td>
<td>Google, Inc</td>
</tr>
<tr>
<td>Jonathan May</td>
<td>USC Information Sciences Institute</td>
</tr>
<tr>
<td>Haitao Mi</td>
<td>IBM Watson Research Center</td>
</tr>
<tr>
<td>Hwee Tou Ng</td>
<td>National University of Singapore</td>
</tr>
<tr>
<td>Baskaran Sankaran</td>
<td>Simon Fraser University</td>
</tr>
<tr>
<td>Hassan Sawaf</td>
<td>eBay Inc</td>
</tr>
<tr>
<td>Lucia Specia</td>
<td>University of Sheffield</td>
</tr>
<tr>
<td>Jörg Tiedemann</td>
<td>Uppsala University</td>
</tr>
<tr>
<td>Christoph Tillmann</td>
<td>TJ Watson IBM Research</td>
</tr>
<tr>
<td>Taro Watanabe</td>
<td>NICT</td>
</tr>
<tr>
<td>Andy Way</td>
<td>CNGL, Dublin City University</td>
</tr>
<tr>
<td>Bing Xiang</td>
<td>IBM</td>
</tr>
<tr>
<td>Muyun Yang</td>
<td>Harbin Institute of Technology</td>
</tr>
<tr>
<td>François Yvon</td>
<td>LIMSI/CNRS</td>
</tr>
<tr>
<td>Rabih Zbib</td>
<td>Raytheon BBN Technologies</td>
</tr>
<tr>
<td>Richard Zens</td>
<td>Google</td>
</tr>
<tr>
<td>Bing Zhao</td>
<td>SRI International</td>
</tr>
</tbody>
</table>
Contents

Expressive Hierarchical Rule Extraction for Left-to-Right Translation
Maryam Siahbani and Anoop Sarkar 1

Bayesian Iterative-cascade Framework for Hierarchical Phrase-based Translation
Baskaran Sankaran and Anoop Sarkar 15

Coarse “split and lump” bilingual language models for richer source information in SMT
Darlene Stewart, Roland Kuhn, Eric Joanis and George Foster 28

Using any MT source for fuzzy-match repair in a computer-aided translation setting
John Ortega, Felipe Sánchez-Martínez and Mikel L. Forcada 42

Enhancing Statistical Machine Translation with Bilingual Terminology in a CAT Environment
Mihael Arcan, Marco Turchi, Sara Tonelli and Paul Buitelaar 54

Clean Data for Training Statistical MT: The Case of MT Contamination
Michel Simard 69

Bilingual phrase-to-phrase alignment for arbitrarily-small datasets
Kevin Flanagan 83

A Probabilistic Feature-Based Fill-up for SMT
Jian Zhang, Liangyou Li, Andy Way and Qun Liu 96

Document-level Re-ranking with Soft Lexical and Semantic Features for SMT
Chenchen Ding, Masao Utiyama and Eiichiro Sumita 110

A Comparison of Mixture and Vector Space Techniques for Translation Model Adaptation
Boxing Chen, Roland Kuhn and George Foster 124

Combining Domain and Topic Adaptation for SMT
Eva Hasler, Barry Haddow and Philipp Koehn 139

Online Multi-User Adaptive Statistical Machine Translation
Prashant Mathur, Mauro Cettolo, Marcello Federico and José G. C. de Souza 152

The Repetition Rate of Text as a Predictor of the Effectiveness of Machine Translation Adaptation
Mauro Cettolo, Nicola Bertoldi and Marcello Federico 166

Expanding Machine Translation Training Data with an Out-of-Domain Corpus using Language Modeling based Vocabulary Saturation
Burak Aydın and Arzuçan Özgür 180

Comparison of Data Selection Techniques for the Translation of Video Lectures
Joern Wuebker, Hermann Ney, Adrià Martínez-Villaronga, Adrià Giménez, Alfons Juan, Christophe Servan, Marc Dymetman and Shachar Mirkin 193

Review and Analysis of China Workshop on Machine Translation 2013 Evaluation
Sitong Yang, Heng Yu, Hongmei Zhao, Qun Liu and Yajuan Iü 208
Combining Techniques from different NN-based Language Models for Machine Translation
Jan Niehues, Alexander Allauzen, François Yvon and Alex Waibel 222

Katsuhito Sudoh, Masaaki Nagata, Shinsuke Mori and Tatsuya Kawahara 234

A Discriminative Framework of Integrating Translation Memory Features into SMT
Liangyou Li, Andy Way and Qun Liu 249

Assessing the Impact of Speech Recognition Errors on Machine Translation Quality
Nicholas Ruiz and Marcello Federico 261

Using Noun Class Information to Model Selectional Preferences for Translating Prepositions in SMT
Marion Weller, Sabine Schulte im Walde and Alexander Fraser 275

Predicting Human Translation Quality
Lucia Specia and Kashif Shah 288

Data Selection for Compact Adapted SMT Models
Shachar Mirkin and Laurent Besacier 301

Pivot-based Triangulation for Low-Resource Languages
Rohit Dholakia and Anoop Sarkar 315

An Arabizi-English Social Media Statistical Machine Translation System
Jonathan May, Yassine Benjira and Abdessamad Echihabi 329

Automatic Dialect Classification for Statistical Machine Translation
Saab Mansour, Yaser Al-Onaizan, Graeme Blackwood and Christoph Tillmann 342

A Tunable Language Model for Statistical Machine Translation
Junfei Guo, Juan Liu, Qi Han and Andreas Maletti 356
Expressive Hierarchical Rule Extraction for Left-to-Right Translation

Maryam Siahbani
msiahban@cs.sfu.ca
Anoop Sarkar
anoop@cs.sfu.ca
School of Computing Science,
Simon Fraser University,
Burnaby, V5A 1S6, Canada

Abstract

Left-to-right (LR) decoding Watanabe et al. (2006) is a promising decoding algorithm for hierarchical phrase-based translation (Hiero) that visits input spans in arbitrary order producing the output translation in left to right order. This leads to far fewer language model calls. But the constrained SCFG grammar used in LR-Hiero (GNF) with at most two non-terminals is unable to account for some complex phrasal reordering. Allowing more non-terminals in the rules results in a more expressive grammar. LR-decoding can be used to decode with SCFGs with more than two non-terminals, but the CKY decoders used for Hiero systems cannot deal with such expressive grammars due to a blowup in computational complexity. In this paper we present a dynamic programming algorithm for GNF rule extraction which efficiently extracts sentence level SCFG rule sets with an arbitrary number of non-terminals. We analyze the performance of the obtained grammar for statistical machine translation on three language pairs.

1 Introduction

Hierarchical phrase-based translation (Hiero) (Chiang, 2007) uses a lexicalized synchronous context-free grammar (SCFG) extracted from word and phrase alignments of a bitext. Decoding for Hiero is typically done with CKY-style decoding with time complexity $O(n^3)$ for source input with $n$ words. Computing the language model score for each hypothesis within CKY decoding requires two histories, the left and the right edge of each span. This is due to the fact that the target side is built inside-out from sub-spans (Heafield et al., 2011, 2013).

LR-decoding algorithms exist for phrase-based (Koehn, 2004; Galley and Manning, 2010) and syntax-based (Huang and Mi, 2010; Feng et al., 2012) models and also for hierarchical phrase-based models (Watanabe et al., 2006; Siahbani et al., 2013), which is our focus in this paper.

Watanabe et al. (2006) was the first to propose a left-to-right (LR) decoding algorithm for Hiero (henceforth we refer to LR decoding for Hiero as LR-Hiero) which uses beam search and runs in $O(n^2b)$ (in practice) where $n$ is the length of source sentence and $b$ is the size of beam (Huang and Mi, 2010). To simplify target generation, synchronous context-free grammar (SCFG) rules are constrained to be prefix-lexicalized on target side, aka Greibach Normal Form (GNF). Throughout this paper we abuse the notation for simplicity and use the term GNF grammars for such SCFGs. Siahbani et al. (2013) propose an augmented version of LR decoding to

\footnote{Although any monolingual context-free grammar can be converted to Greibach Normal Form, there is no algorithm}
address some limitations in the original LR-Hiero algorithm in terms of translation quality and time efficiency.

Hiero (and LR-Hiero) rule extraction heuristics apply constraints on the length of initial phrase pairs considered for rule extraction, number and configuration of non-terminals in order to avoid excessively large grammars. Thus, obtained rules cannot capture all possible alignments on language pairs with complex reordering. Allowing more non-terminals in the rules is not practical in CKY based decoders because the computational complexity of decoding increases exponentially with the increase in the rank of the grammar (that is, the number of non-terminals permitted in the right hand side of the CFG rules). However, LR decoding is a viable alternative which can efficiently apply these types of rules while keeping quadratic time complexity by using a variant of the dotted rules used in the Earley parsing algorithm for parsing monolingual CFGs.

Standard Hiero rule extraction used to extract GNF grammars (Watanabe et al., 2006; Siahbani et al., 2013) is a brute-force search algorithm which considers all possible replacement of sub-phrases with non-terminals. Despite the constraints on rule configuration, rule extraction is still a bottleneck and it is usually achieved by way of parallelization and optimization. Increasing the length of initial phrase pairs or number of non-terminals exponentially increases the time complexity. In this paper we propose a dynamic programming algorithm for GNF rule extraction that is linear in the output length (the number of GNF rules). We use this algorithm to extract GNF rules with different number of non-terminals including sentence level rules and analyze the effect of these rules in LR-Hiero translation system on three language pairs: Chinese-English, Czech-English and German-English.

2 Left-to-Right Decoding

LR-Hiero uses a constrained lexicalized SCFG. The target-side rules are constrained to be prefix lexicalized, for simplicity called GNF rules:

$$X \rightarrow \langle \gamma, \bar{b} \beta \rangle$$

where $\gamma$ is a string of non-terminal and terminal symbols, $\bar{b}$ is a string of terminal symbols and $\beta$ is a possibly empty sequence of non-terminals. This ensures that as each rule is used in a derivation, the target string is generated from left to right.

Algorithm 1 shows the pseudocode for LR-Hiero decoding with cube pruning (Chiang, 2007) (CP). LR-Hiero with CP was introduced in Siahbani et al. (2013). Each source side non-terminal is instantiated with the legal spans given the input source string, e.g. if there is a Hiero rule $\langle aX_1, a'X_1 \rangle$ and if $a$ only occurs at position 3 in the input then source side $X_1$ is instantiated to span [4, n], for input of length n. A worked out example of how the decoder works is shown in Figure 1. Each partial hypothesis $h$ is a 4-tuple $(h_t, h_s, h_{cov}, h_c)$: consisting of a translation prefix $h_t$, a (LIFO-ordered) list $h_s$ of uncovered spans, source words coverage set $h_{cov}$, and the hypothesis cost $h_c$ which includes future cost and a score computed based on feature values (using a log-linear model). The initial hypothesis is a null string with just a sentence-initial marker $\langle s \rangle$ and the list $h_s$ containing a span of the whole sentence, [0, n]. The hypotheses are stored in stacks $S_0, \ldots, S_p$, where $S_p$ contains hypotheses covering $p$ source words just as in stack decoding for phrase-based SMT (Koehn et al., 2003).

to convert an arbitrary SCFG to a weakly equivalent SCFG with rules constrained to be prefix-lexicalized on the target side.

Greibach Normal Form (GNF). Just the target side is prefix lexicalized (GNF form), not the synchronous grammar.

The future cost is precomputed in a way similar to the phrase-based models (Koehn et al., 2007) using only the terminal rules of the grammar.
Algorithm 1 LR-Hiero Decoding with CP

1: Input sentence: \( f = f_0 f_1 \ldots f_n \) (Precompute future cost\(^3\) for spans)
2: \( \mathcal{F} = \text{FutureCost}(f) \) (Create empty initial stack)
3: \( S_0 = \{\} \) (Initial hypothesis 4-tuple)
4: \( h_0 = (\{s\}, [0, n], \emptyset, \mathcal{F}[0, n]) \) (Push initial hyp into first Stack)
5: Add \( h_0 \) to \( S_0 \)
6: for \( i = 1, \ldots, n \) do
7:  \( \text{cubeList} = \{\} \) (MRL is max rule length)
8:  for \( p = \max(i - \text{MRL}, 0), \ldots, i - 1 \) do
9:   \( \{G\} = \text{Grouped}(S_p) \) (based on the first uncovered span)
10:  for \( g \in \{G\} \) do
11:   \[u, v\] = \( g_{span} \) (gspan)
12:   \( R = \text{GetSpanRules}([u, v]) \)
13:   for \( R_s \in R \) do
14:     cube = \([g_{hyps}, R_s]\) (Create stack \( S_i \) and add new hypotheses to it)
15:     Add cube to cubeList
16:   end for
17:   \( S_i = \text{Merge(cubeList, } \mathcal{F}) \)
18: end for
19: return argmin \( h_c \) \( \in S_n \)
20: \( \text{heapQ} = \{\} \)
21: for each \((H, R)\) in cubeList do
22:  \( h' = \text{GetBestHypotheses}(H, R, \mathcal{F}) \) (best hypotheses of cubes)
23:  push(\( \text{heapQ}, (h'_c, h', [H, R]) \)) (Push new hyp in the queue)
24: hypList = \( \{\} \)
25: while \( |\text{heapQ}| > 0 \) and \( |\text{hypList}| < K \) do
26:  \( (h'_c, h', [H, R]) = \text{pop(} \text{heapQ}) \) (pop the best hypothesis)
27:  push(\( \text{heapQ}, \text{GetNeighbours}([H, R]) \)) (Push neighbours to queue)
28:  Add \( h' \) to hypList
29: return hypList

To fill stack \( S_i \) we consider hypotheses in each stack \( S_p \), which are first partitioned into a set of groups \( \{G\} \), based on their first uncovered span (line 9). Each group \( g \) is a 2-tuple \((g_{span}, g_{hyps})\), where \( g_{hyps} \) is a list of hypotheses which share the same first uncovered span \( g_{span} \). Rules matching the span \( g_{span} \) are obtained from routine \( \text{GetSpanRules} \). Each \( g_{hyps} \) and possible \( R_s \) create a cube which is added to \( \text{cubeList} \).

The \( \text{Merge} \) routine gets the best hypotheses from all cubes. \( \text{GetBestHypotheses}((H, R), \mathcal{F}) \) uses current hypothesis \( H \) and rule \( R \) to produce new hypotheses. The first best hypothesis, \( h'_c \) along with its score \( h'_c \) and corresponding cube \((H, R)\) is placed in a priority queue \( \text{heapQ} \) (line 22 in Algorithm 1). Iteratively the \( K \) best hypotheses in the queue are popped (line 25) and for each hypothesis its neighbours in the cube are added to the priority queue (line 26). Decoding finishes when stack \( S_n \) has been filled.

3 Rule Extraction

Hiero uses a synchronous context free grammar (SCFG), \( X \rightarrow \langle \gamma, \alpha \rangle \), where \( X \) is a non-terminal, \( \gamma \) and \( \alpha \) are strings of terminals and non-terminals (Chiang, 2005, 2007). Unlike typical SCFGs, the rules are lexicalized on the right hand side with at least one aligned word pair in source and target.

\(^4\)As the length of rules is limited (at most MRL), we can ignore stacks with index less than \( i - \text{MRL} \).
Table 1: The process of translating German-English sentence pair in LR-Hiero. Word alignment is shown in Figure 4(a). Left side shows the rules used in the derivation (G indicates glue rules as defined in Watanabe et al. (2006)). The hypotheses column shows 4-tuple partial hypotheses: the translation prefix, $h_t$, the ordered list of yet-to-be-covered spans, $h_s$, source word coverage vector, $h_{cov}$, and cost $h_c$ (cost includes future cost and hypothesis cost, but we just show hypothesis cost in this figure).

Chiang (2007) places certain constraints on the extracted rules in order to simplify decoding. This includes limiting the maximum number of non-terminals (rule arity) to two and disallowing any rule with consecutive non-terminals on the source language side. It further limits the length of the initial phrase-pair to a maximum phrase length. For translating sentences longer than the maximum phrase pair length, the decoder relies on additional glue rules $S \rightarrow \langle X, X \rangle$ and $S \rightarrow \langle SX, SX \rangle$ that allow monotone combination of phrases. The glue rules are used when no rules could match or the span length is larger than the maximum phrase-pair length.

LR-Hiero generates the target hypotheses left to right, but for synchronous context-free grammar (SCFG) as used in Hiero. Therefore LR-Hiero restricts the grammar to GNF rules (equation 1). The rules are obtained from a word and phrase aligned bitext using a rule extraction algorithm (see Section 3.1). To overcome data sparsity and obtain better generalization, four glue rules are added for each terminal rule $\langle f, e \rangle$. The glue rules allow reordering as well as monotone combination of phrases:

$X \rightarrow \langle fX_1, eX_1 \rangle \quad X \rightarrow \langle X_1fX_2, eX_1X_2 \rangle \quad X \rightarrow \langle X_1f, eX_1 \rangle \quad X \rightarrow \langle X_1fX_2, eX_2X_1 \rangle$ (2)

### 3.1 Hiero Rule Extraction

The Hiero grammar extraction (Chiang, 2007) starts from the set of initial phrases that are identified by growing the word alignments into longer phrases. Given the initial phrases of a sentence pair, the extraction algorithm first designates the smaller initial phrases as terminal rules. Then it extracts hierarchical rules by substituting the smaller spans within the larger phrases by a non-terminal $X$. It extracts all possible rules from the initial phrases subject to a maximum of two non-terminals in a rule such that they are not adjacent in the source side. The Hiero extraction assumes unit count for each initial phrase and distributes this uniformly to all the rules extracted from the phrase. The parameter estimation then proceeds by relative frequency estimation. LR-Hiero uses similar method for grammar extraction, except any rules violating GNF form on the target side are excluded (Watanabe et al., 2006; Siahbani et al., 2013).

This algorithm is a brute-force search which considers all possible replacement of sub-phrases with non-terminals. Although Hiero and LR-Hiero use initial phrase pairs of limited length (usually 10) and grammar is limited to at most two non-terminals, rule extraction is still a bottleneck and it is generally achieved by way of parallelization and optimization. Increasing the length of initial phrase pairs or number of non-terminals exponentially increases the time complexity. In section 3.3 we propose a dynamic programming algorithm for GNF rules...
extraction from a sentence pair that is linear in the output length (the number of GNF rules).

\[
\begin{array}{cccccc}
e_0 & e_1 & e_2 & e_3 & e_4 & e_5 \\
f_0 & f_1 & f_2 & f_3 & f_4 \\
\end{array}
\]

Figure 2: Example phrase pair with alignments.

\[
\begin{array}{cccc}
([0,5],[0,4]) & ([0,2],[0,2]) & ([4,5],[3,4]) \\
([0,1],[0,1]) & ([2,2],[2,2]) \\
([0,0],[0,0]) & ([1,1],[1,1]) \\
\end{array}
\]

Figure 3: Decomposed alignment tree for the example alignment in Fig. 2.

3.2 Phrase Pair Extraction

Unlike Hiero rule extraction, we do not limit the length of initial phrase pairs and extract rules from all phrase pairs (including whole sentence pairs) in the training data. A modified version of the algorithm by (Zhang et al., 2008) is used to efficiently extract phrase pairs. For a phrase pair with a given alignment as shown in Figure 2, Zhang et al. (2008) generalize the \(O(n + K)\) time algorithm for computing all \(K\) common intervals of two different permutations of length \(n\). The contiguous blocks of the alignment are captured as the nodes in the alignment tree and the tree structure (for example, phrase pair in Figure 2 is shown in Figure 3). The italicized nodes form a left-branching chain in the alignment tree and the sub-spans of this chain also lead to alignment nodes that are not explicitly captured in the tree (Please refer to (Zhang et al., 2008) for details).

3.3 GNF Extraction

We first explain the rule extraction algorithm using a working example, then discuss correctness of the algorithm. Let \(pp = (\bar{f}, \bar{e})\) be a source-target phrase pair, where \(\bar{f}\) and \(\bar{e}\) are corresponding phrases on source and target side. We define largest right sub-phrase, for a target interval \([i, j]\), as the largest phrase pair (in terms of length of target side) with right boundary \(j\) on the target side, and denote it by \(LRS(i, j)\):

\[
LRS(i, j) = \arg\max_{(\bar{f}, \bar{e}) \in S(i, j)} |\bar{e}|
\]

\[
S(i, j) = \{(\bar{f}, \bar{e}) | (\bar{f}, \bar{e}) \in \mathcal{P}, |\bar{e}| < |j - i|, \bar{e}.\text{end()} = j\}
\]  

where \(\mathcal{P}\) is a set of all phrase pairs, \(|\bar{e}|\) denotes length of \(\bar{e}\), \(\bar{e}.\text{end()}\) returns the last index of the span corresponding to \(\bar{e}\) (in the target sentence). \(S\) is empty set for phrase pair with target spans of length one \((i = j)\). For example in Figure 4, the \(LRS[1, 5]\) is \(<\text{noch nicht gemacht}, \text{not yet done}>\)\(^5\). \(LRS\) can be precomputed for all span lengths in

\(^5\)In Figure 4, we identify phrase pairs to target spans, \(LRS[1, 5] = (3, 5)\).
Algorithm 2 GNF Rule Extraction

1: Input $f(f_1 \ldots f_n), e(e_1 \ldots e_m), A$ (A is alignment)
2: $P = \text{ExtractPhrases}(f, e, A)$ (generate all possible phrase pairs, increasingly sorted based on length of target side)
3: $LRS = \text{RightSubPhrases}(P, m)$ (precompute largest right sub-phrases)
4: $R = \{\}$
5: for $pp \in P$ do
6:   $(i, j) = \bar{e}_{pp}$ (target span of $pp$)
7:   $R_{i,j} = \{\}$ (rules for target span $[i,j]$)
8:   $\text{curr_rule} = pp$ (create a terminal rule)
9:   $\text{AddRule}(R_{i,j}, \text{curr_rule})$
10: $t = j$
11: while $t \geq i$ do
12:   $pp' = LRS[(i, t)]$
13:   if $pp'$ is None then
14:     break
15:   $(k, t) = \bar{e}_{pp'}$ (target span of $pp'$)
16:   for each $r \in R_{k,t}$ do
17:     $r' = \text{Substitute}(\text{curr_rule}, pp', r)$
18:     $\text{AddRule}(R_{i,j}, r')$
19:   $\text{curr_rule} = \text{Substitute}(\text{curr_rule}, pp', X)$ (replace subphrase with a non-terminal)
20:   $\text{AddRule}(R_{i,j}, \text{curr_rule})$
21:   $t = k - 1$
22: $LRS[(i,j)] = pp$ (update $LRS$)
23: $\text{Add } R_{i,j} \text{ to } R$
24: return $R$

25: $\text{RightSubPhrases}(P, m)$
26: $LRS = \{\}$
27: for $l = 2, \ldots, m$ do
28:   for $i = 1, \ldots, m - l$ do
29:     $j = i + l - 1$
30:     if $\exists pp \in P, \bar{e}_{pp} = (i+1,j)$ then
31:       $LRS[(i,j)] = pp$
32:     elif $(i+1,j) \in LRS$ then
33:       $LRS[(i,j)] = LRS[(i+1,j)]$
34: return $LRS$

$O(n^2)$, where $n$ is target sentence length (routine $\text{RightSubPhrases}$ in Algorithm 2). Figure 4 (b) shows the chart of $LRS$ computed by $\text{RightSubPhrases}$ for sentence pair in Figure 4 (a). Each cell corresponds to a span on the target side.

Algorithm 2 shows the pseudocode for GNF rule extraction. It is a dynamic programming algorithm that extracts GNF rules for phrase pairs (gradually from small to large phrase pairs). It works bottom up and fills a chart, $R$, on the target sentence. Each cell $R_{i,j}$ keeps all rules that can cover a phrase pair $pp = (f, e)$, where $\bar{e}$ corresponds to span $[i, j]$ on the target sentence. At the end, it returns $R$ which is the union of rules for all target spans (i.e. all possible phrase pairs).

First, routine $\text{ExtractPhrases}$ extracts all phrase pairs $P$ and sorts them increasingly based
[3,4]
1) $X \rightarrow \langle \text{noch nicht} / \text{not yet} \rangle$
2) $X \rightarrow \langle X, \text{not} \rangle$

[6,7]
1) $X \rightarrow \langle \text{ihre arbeit} / \text{their work} \rangle$
2) $X \rightarrow \langle \text{ihre}, \text{their} \rangle$
3) $X \rightarrow \langle X, \text{not} \rangle$

[3,7]
1) $X \rightarrow \langle \text{noch nicht gemacht} / \text{not yet done} \rangle$
2) $X \rightarrow \langle X, \text{not} \rangle$
3) $X \rightarrow \langle X, \text{not} \rangle$

[2,8]
1) $X \rightarrow \langle \text{ihre arbeit noch nicht gemacht haben} / \text{have not yet done their work} \rangle$
2) $X \rightarrow \langle \text{ihre arbeit noch nicht gemacht haben X, have not yet done their work X} \rangle$
3) $X \rightarrow \langle \text{ihre X, noch nicht gemacht haben X, have not yet done their work X} \rangle$
4) $X \rightarrow \langle X, \text{noch nicht gemacht haben X, have not yet done X, X} \rangle$
5) $X \rightarrow \langle X, \text{noch nicht haben X, have not yet X, X} \rangle$
6) $X \rightarrow \langle X, \text{haben X, have X} \rangle$

Figure 4: GNF rule extraction for a German-English sentence pair. (a) bars above (below) the source (target) words indicate phrase-pairs. (b) LRS chart for this sentence, filled by RightSubPhrase (green arrows shows some cells corresponding to phrase pairs which are updated during rule extraction). The span above each set of rules shows the target side of the corresponding phrase pair. (c) Extracting rules for span [3,7]: rule #2 is created using rules of span [6,7], #3 replacing [6,7] with non-terminal, rules #4, #5 created from span [3,5]. Invalid rules are shown in grey. (d) Extracting rules for span [2,8].

on their target length (line 2). LRS is computed for all target spans by RightSubPhrases. Then, in a for loop on all phrase pairs, chart of the rules will be filled in a bottom up manner, small to large spans (lines 5-23). For each initial phrase pair a terminal rule is created and added to $R_{i,j}$ (line 8). Then, using rules from smaller phrase pairs, more rules are generated (line 11-21).

The largest right sub-phrase, $pp'$ is obtained for initial phrase pair $pp$ in line 12 (note that $t$ is set to the right boundary of $pp$ (i.e. $j$) at the beginning). Target span of $pp'$, $[k, t]$, is used to retrieve rules for $pp'$, stored in $R_{k,t}$. Replacing each rule of $pp'$ in our $curr\_rule$.

---

7It is the initial phrase pair $pp$ at the beginning.
routine) results in a new rule for \( pp \) (lines 16-18). And as the last rule that can be generated using \( pp' \), the whole \( pp' \) in \( curr\_pbr \) is replaced with a non-terminal (line 19).

All rules for \( pp \) which include rules from \( pp' \) have been generated, thus we can safely replace \( pp' \) with a non-terminal and continue to generate more rules by replacing other parts of \( pp \) with non-terminals. \( curr\_rule \) is updated to the last rule, \( t \) is updated to the index span of \( pp' \) on the target (line 21)\(^8\). The algorithm repeats the loop to find another sub-phrase pair in \( curr\_rule \), and continues until no sub-phrase pair is found (or we reach the beginning of target phrase (\( t \) equals \( i \)). When all the rules for \( pp \) are computed, \( LRS[(i, j)] \) is updated to \( pp \) so that it can be used in larger phrases. In fact \( LRS[(i, j)] \) should always show the largest right subphrase which its rules have already been extracted. Note that only if \( [i, j] \) corresponds to a phrase-pair, \( LRS[(i, j)] \) is updated (in Figure 4(b), some updated cells are shown in green).

Figure 4(c) shows how algorithm extracts rules for span \([3, 7] \); at first \( curr\_rule \) is equal to the initial phrase pair. rule #1 is a terminal rule; \( LRS[3, 7] \) is \( \langle \text{the work} \rangle \) (target span \([6, 7] \)), rule #2 is generated using rules for \([6, 7] \); rule #3 is the result of replacing \([6, 7] \) with a non-terminal. Then \( curr\_rule \) and \( t \) are updated. \( LRS[3, 5] \) is sub-phrase pair \( \langle \text{not yet done} \rangle \), (rules for this phrase pair have been already computed and consequently \( LRS[3, 5] \) has been updated to \([4,6],[3,5] \)). Rules #4 and #5 are generated using rules for span \([3, 5] \). Then \( curr\_rule \) and \( t \) are updated. As no lexical item remains in the target side of \( curr\_rule \), the algorithm stops.

\( AddRule \) verifies the rule configuration like the number of non-terminals and non-adjacent non-terminals on the source side. If the rule is valid, it is added to the corresponding cell, \( R_{i,j} \) (e.g. rule #5 is not valid because of adjacent non-terminals on the source side).

### 3.3.1 Correctness

We show that for a given phrase pair, this algorithm extracts all possible Hiero style SCFG rules which are in GNF format on the target side (the same as Hiero brute-force rule extraction). Given a phrase pair \( pp = (f, e) \) with target span \([i, j] \), \( LRS(i, j) \) shows the largest subproblem that can be optimally used to generate rules for \( pp \), denoted by \( R_{i,j} \).

**Optimal structure:** \( R_{i,j} \) consists of two disjoint sets

\[
R_s = \{ r | r = \text{Substitute}(pp, pp', r') \forall r' \in R_{i',j'} \} \\
R_x = \{ r | r.pos(pp') = X \}
\]

where \( r.pos(pp') \) denotes the interval of \( pp' \) in \( r \). \( R_s \) is the set of rules obtained by replacing \( pp' \) in \( pp \) with each rule of \( R_{i',j'} \), while \( R_x \) is the set of rules having non-terminal \( X \) in position of \( pp' \) (in source and target side). GNF rules on the target side (equation 1), ends with a non-terminal (if there is one) and there is no lexical item between non-terminals. Assuming this we can consider two states for each \( r \in R_{i,j} \): (a) \( r \) has some lexical term in \( r.pos(pp') \); (b) \( r \) has a non-terminal in position \( pp' \). Case (a) is equal to set \( R_s \): this type of rules can have non-terminal just in the interval of \( pp' \) (because any non-terminal out of \( pp' \) violates GNF format on the target side). And if there is a rule of this type it corresponds to a rule in \( R_{i',j'} \). Consequently case (b) is equal to set \( R_x \). It means that any \( r \notin R_s \), should replace a non-terminal instead of \( pp' \) (otherwise violates GNF format on the target side). Computing \( R_x \) corresponds to a smaller problem \([i, i' - 1] \) (let’s define \( t = i' - 1 \)) which can be solved in a similar way. If \([i, t] \) is target side of a phrase pair (like \([3, 5] \) in Figure 4(c)), we just need to use rules in \( R_{i,t} \) to generate more rules and keep all valid rules. Otherwise we repeat the process: find the largest sub-problem, \( LRS[i, t] = (k, t) \), use \( R_{k,t} \) to generate more rules, then replace \([k, t] \) in the rule

---

\(^8\)[i, t] always shows the lexical part of the target side.
Table 1: Corpus statistics in number of sentences. Tuning and test sets for Chinese-English has 4 references.

<table>
<thead>
<tr>
<th>Corpus</th>
<th>Train/Dev/Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cs-En</td>
<td>Europarl(v7) + CzEng(v0.9); News commentary(nc) 2008&amp;2009; nc 2011</td>
</tr>
<tr>
<td>De-En</td>
<td>Europarl(v7); WMT2006; WMT2006</td>
</tr>
<tr>
<td>Zh-En</td>
<td>HK parallel-tex + GALE ph-1; MTC parts 1&amp;3; MTC part4</td>
</tr>
</tbody>
</table>

Table 2: Model sizes in millions of rules. Maximum source length (msl) is shown in brackets.

<table>
<thead>
<tr>
<th>Model</th>
<th>Cs-En</th>
<th>De-En</th>
<th>Zh-En</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCFG</td>
<td>1,961.6</td>
<td>858.5</td>
<td>471.8</td>
</tr>
<tr>
<td>GNF</td>
<td>306.3</td>
<td>116.0</td>
<td>100.9</td>
</tr>
<tr>
<td>GNF-4</td>
<td>380.9</td>
<td>214.9</td>
<td>190.0</td>
</tr>
</tbody>
</table>

Table 3: No. of sentence covered in forced decoding.

<table>
<thead>
<tr>
<th>Model</th>
<th>Cs-En</th>
<th>De-En</th>
<th>Zh-En</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCFG</td>
<td>318</td>
<td>351</td>
<td>187</td>
</tr>
<tr>
<td>GNF-2</td>
<td>278</td>
<td>300</td>
<td>132</td>
</tr>
<tr>
<td>GNF-4</td>
<td>306</td>
<td>375</td>
<td>163</td>
</tr>
</tbody>
</table>

4 Experiments

To evaluate our rule extraction algorithm, we use it to extract the grammar for LR-Hiero on three language pairs: German-English (De-En), Czech-English (Cs-En) and Chinese-English (Zh-En). Table 1 shows the details of datasets.

We use 2 baselines: (i) LR-Hiero in Python (we use the implementation described in (Siahbani and Sarkar, 2014)); (ii) Kriya (Sankaran et al., 2012b), an open-source implementation of Hiero in Python (available on https://github.com/sfu-natlang/Kriya) which performs comparably to other open-source Hiero systems. Both systems are in Python and use the same LM wrapper which allows us to make a fair comparison of LM calls and time differences in decoding.

We use rule extraction of Kriya to extract Hiero (SCFG) and modify it to extract LR-Hiero (GNF). Both grammars use similar configuration and settings: rule arity 2, maximum source length 7, initial phrase pairs of length at most 10. We use our rule extraction algorithm to extract GNF rules from all initial phrase pairs (any length), rule arity 1 to 4, maximum source rule length 10. Like Hiero, we filter rules with adjacent non-terminals on the source side. Terminal rules are constrained to maximum source rule length 7. We use rule count estimation heuristic similar to Hiero. Table 2 shows model sizes for LR-Hiero (GNF), Hiero (SCFG) and GNF grammar with at most 4 non-terminals (GNF-4). Typical Hiero rule extraction excludes phrase-pairs with unaligned words on boundaries (loose phrases). We include loose phrase-pairs as terminal rules in all GNF grammars.

To evaluate our grammar, we use all grammars in LR-Hiero decoder and compare them with SCFG grammar in Hiero decoder. We use a 5-gram LM trained on the Gigaword corpus.
<table>
<thead>
<tr>
<th>Model</th>
<th>Cs-En</th>
<th>De-En</th>
<th>Zh-En</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hiero</td>
<td>20.77</td>
<td>25.72</td>
<td>27.69</td>
</tr>
<tr>
<td>LR-Hiero Watanabe et al. (2006)</td>
<td>20.72</td>
<td>25.05</td>
<td>25.99</td>
</tr>
<tr>
<td>LR-Hiero+CP</td>
<td>20.52</td>
<td>25.07</td>
<td>26.10</td>
</tr>
<tr>
<td>LR-Hiero+CP (GNF-1)</td>
<td>20.38</td>
<td>24.20</td>
<td>25.81</td>
</tr>
<tr>
<td>LR-Hiero+CP (GNF-2)</td>
<td>20.49</td>
<td>25.32</td>
<td>25.92</td>
</tr>
<tr>
<td>LR-Hiero+CP (GNF-3)</td>
<td>20.50</td>
<td>25.34</td>
<td>26.13</td>
</tr>
<tr>
<td>LR-Hiero+CP (GNF-4)</td>
<td>20.50</td>
<td>25.34</td>
<td>26.10</td>
</tr>
</tbody>
</table>

Table 4: BLEU scores for Hiero and LR-Hiero with and without cube pruning (CP). GNF-x: GNF grammars with at most \( x \) non-terminals using the proposed rule extraction algorithm.

Figure 5: Average number of language model queries. (GNF4) denotes new GNF grammar with 4 non-terminals.

and use KenLM (Heafield, 2011). Pop limit for Hiero and LR-Hiero is 500. To make the results comparable we use the same feature set for all baselines which includes standard features of Hiero: two relative-frequency probabilities \( p(e|f) \) and \( p(f|e) \), two lexically weighted probabilities \( \text{lex}(e|f) \) and \( \text{lex}(f|e) \), a language model probability, word penalty, phrase penalty, and glue rule penalty, and we add distortion features (seperated for regular and glue rules in LR-Hiero) proposed by Siahbani et al. (2013). Weights are tuned by minimizing BLEU loss on the dev set through MERT (Och, 2003) and BLEU scores on test set are reported.

Table 4 shows the BLEU score for different decoders and grammars. The last 4 rows are GNF grammar with 1 to 4 non-terminals extracted by our rule extraction. To show how adding more non-terminals affect the alignment coverage, we translate the devset sentences with different grammars in forced decoding mode. We use CKY decoding for SCFG and LR-decoder for GNF grammars. Table 3 shows the size of the reachable subset by forced decoding for different grammars. It shows that adding more non-terminals considerably improves the alignment coverage on De-En and Zh-En (average 24%).

Comparing Tables 4 and 3 is interesting. While adding rules with more than 2 non-terminals does not change BLEU score it improves the alignment coverage. In our analysis we notice that LR-Decoder rarely uses rules with 3 or 4 non-terminals in \( K \)-best list. It is probably because, rules with less non-terminals are generally more frequent and hypotheses which use them have got higher score during decoding. Here we just use Hiero and LR-Hiero standard features which are not designed for rules with more complex reordering. The next step is to elaborate features for rules with 3 and 4 non-terminals\(^9\).

To evaluate the effect of the grammars on decoding process in terms of speed, we use

\(^9\)In another experiment not reported here, we extract rules with unlimited number of non-terminals and source rule length for Cs-En (while we keep non-adjacent non-terminals on the source side). But filtering rules on dev and test sets results in rules with at most 5 non-terminals.
number of language model calls since that directly corresponds to the number of hypotheses considered by the decoder, consequently the speed of decoder. Figure 5 shows the results in terms of average number of language model queries and times in milliseconds on a sample set of 50 sentences from test sets.

5 Related Work

Many approaches have been developed to improve SCFG rules for Hiero. Some of the works have employed generative methods using Bayesian techniques to induce SCFG (Blunsom et al., 2008, 2009; Levenberg et al., 2012; Sankaran et al., 2012a) directly from bilingual data without word alignments. de Gispert et al. (2010) extract rules based on posterior distributions provided by the HMM word-to-word alignment model, rather than a single alignment which is used in original Hiero. Most of these approaches restrict the grammar to rules with one or at most two non-terminals to be able to use the grammar in decoding (Blunsom et al., 2008; de Gispert et al., 2010; Sankaran et al., 2012a).

Recently Levenberg et al. (2012) propose an approach to learn grammars with unrestricted number of non-terminals but do not use the grammar directly in the decoder. The obtained SCFG rules are used to obtain the word alignments rather than the SCFG rules for decoding. Unrestricted number of non-terminals makes the induced grammar unusable in CKY based decoders.

Zhang et al. (2008) encode the word aligned sentence pair as a normalized decomposition tree (a hierarchical representation of all the phrase pairs in linear time, which yields a set of minimal Hiero (SCFG) rules. They discuss that the method can be modified to extract all Hiero rules. But the algorithm is just applied as an analytical tool for aligned bilingual data.

Syntax-based translation systems, tree-to-tree (Ding and Palmer, 2005), tree-to-string (Liu et al., 2006; Huang, 2006) and string-to-tree (Galley et al., 2006), extract sentence level rules, but they extract rules from parse trees (on source or target) rather word aligned sentence pairs which we discussed in this paper.

Braune et al. (2012) extend Hiero by extracting an additional and separate set of rules for long-distance reorderings. They modify Hiero extractor based on some analysis on long-distance German-to-English movement and filtered them based on linguistic information. New rules are applied to long spans (11 to 50) but do not improve translation quality in terms of BLEU (in some case BLEU scores reduce by 0.4). However they show that their approach helps in terms of improving the reordering between source and target (using LRscore (Birch and Osborne, 2011) evaluation scores and some manual evaluation).

6 Conclusion

We propose a dynamic programming algorithm for GNF rule extraction that is linear in the number of GNF rules. We use the sentence level GNF rules with different number of non-terminals in LR-decoder and analyze the effect of these rules in LR-Hiero translation system on different language pairs. New rules with more non-terminals improve the alignment coverage (24% on average) on language pairs with more complex reordering, while it marginally affects the decoding speed. Using rules with more non-terminals is a promising approach in Hiero translation systems which is practical using LR-decoding.

Acknowledgments

This research was partially supported by NSERC, Canada RGPIN: 262313 and RGPAS: 446348 grants to the second author. The authors wish to thank Ramtin Mehdizadeh Seraj for his valuable discussions and the anonymous reviewers for their helpful comments.
References


Watanabe, T., Tsukada, H., and Isozaki, H. (2006). Left-to-right target generation for hierarchi-
cal phrase-based translation. In *Proc. of ACL*.

word-level alignments in linear time. In *Proceedings of the 22nd International Conference
on Computational Linguistics (COLING-08)*, pages 1081–1088, Manchester, UK.
Bayesian Iterative-cascade Framework for Hierarchical Phrase-based Translation

Baskaran Sankaran  
Anoop Sarkar  
School of Computing Science, Simon Fraser University, Burnaby, BC. V5A 1S6. Canada

Abstract

The typical training of a hierarchical phrase-based machine translation involves a pipeline of multiple steps where mistakes in early steps of the pipeline are propagated without any scope for rectifying them. Additionally the alignments are trained independent of and without being informed of the end goal and hence are not optimized for translation. We introduce a novel Bayesian iterative-cascade framework for training Hiero-style model that learns the alignments together with the synchronous translation grammar in an iterative setting. Our framework addresses the above mentioned issues and provides an elegant and principled alternative to the existing training pipeline. Based on the validation experiments involving two language pairs, our proposed iterative-cascade framework shows consistent gains over the traditional training pipeline for hierarchical translation.

1 Introduction

Hierarchical phrase-based translation, similar to other statistical machine translation (SMT) models are trained in a series of steps that are disparate and often invoke heuristics. The training complexity as well as the modelling deficiencies in learning the translation rules using such multi-step, heuristic ridden pipeline have been documented in many previous publications (Burkett et al., 2010; DeNero and Klein, 2010; Saers et al., 2013a).

Secondly the early steps in the training pipeline, are unaware of and are almost always at odds with, the final goal of training a translation model. As a specific example, the alignment models are trained early in the pipeline, isolated from the step that extracts translations and this could lead to sub-optimal alignments (DeNero and Klein, 2010). This is also true for the syntactic models that rely on word alignments to extract the translation rules that are consistent with those alignments (Galley et al., 2006; Chiang, 2007, inter alia).

Consider the example word-aligned phrase pair shown in Figure 1. The baseline Giza++ alignment incorrectly aligns the English the to the Chinese word 联合国 (united nations). While the aligner is not going to be perfect, the present serial pipeline does not allow the aligner to correct such mistakes or to adapt the alignments to yield better translation rules.

Further the serial pipeline results in the propagation of the modelling deficiencies from the
early steps to the latter steps. For the above example, the rule extractor of the original Hiero is likely to learn several translation rules some of which are shown in Figure 2. Some of these rules marked by an asterisk (*) encode the incorrect alignment and will lead to patently wrong translations, when applied in a slightly different context.

\[
\begin{align*}
\ast X & \rightarrow \langle 當聯盟 ||| the \rangle \\
\ast X & \rightarrow \langle 月 , 當聯盟 ||| 月 , the \rangle \\
\ast X & \rightarrow \langle 當聯盟 X_1 ||| the X_1 \rangle \\
X & \rightarrow \langle 月 , X_1 ||| 月 , X_1 \rangle \\
X & \rightarrow \langle 當聯盟 難民 专员 公署 ||| the unhcr \rangle \\
X & \rightarrow \langle X_1 當聯盟 X_2 ||| X_1 the X_2 \rangle \\
X & \rightarrow \langle X_1 , X_2 難民 专员 公署 ||| X_1 , X_2 unhcr \rangle 
\end{align*}
\]

Figure 2: Some extracted translation rules for the phrase-pair in Figure 1. Rules marked * will lead to wrong translations when used in other contexts.

We present a novel iterative-cascade framework as a way to address these issues in the training pipeline of Hiero-style systems. Our framework reduces the disparate multi-step pipeline with a simple two-step cascade model embedded in an iterative setting that allows the individual steps to improve based on some feedback from the other.

## 2 Iterative-cascade Framework

We now explain the intuitive idea behind our framework. The key idea of the framework is to separate the inference of alignments and hierarchical translation grammar in two successive steps and then enclose the two steps in an iterative setup. Given the dissimilarity between the alignments and SCFG rules this separation makes it easier for the models to handle the two structures at different steps. Thus the first phase reasons over the sentence pairs to find overlapping alignments, yielding a segmentation for sentence pairs, as phrase-pairs. Subsequently the second phase, searches over the space of derivations (of the phrase-pairs) in order to learn the optimal ones leading to better grammar.

The framework consists of two steps, viz. i) generating phrase alignments of different granularities and ii) extracting SCFG rules that are consistent with the alignments. The two
phases of the iterative-cascade framework are then repeated in an iterative setup. While we could possibly come up with a single model to achieve this, we intend to validate our framework in this work using a simpler approach. We do this by using existing Bayesian models for each step in this paper.

We use the Bayesian hierarchical ITG alignment model (Neubig et al., 2011) for getting the phrasal alignments at the first step. For the phrase extraction step, we use the Bayesian model motivated by a lexical alignment prior employing Variational-Bayesian inference proposed by Sankaran et al. (2013), which operates on the extracted phrasal alignments in the earlier step. We now explain the two models briefly.

### 2.1 Alignments

The joint model proposed by (Neubig et al., 2011) uses a phrasal-ITG based hierarchical model with a Pitman-Yor Process (PYP) prior. Unlike the earlier models (DeNero et al., 2008; Zhang et al., 2008) that extract minimal many-to-many phrase alignments, Neubig’s model extracts phrases of varying granularities. This is achieved by inverting the order to first generate the entire sentence/phrase pair from a phrase distribution ($P_t$) and then falls back to ITG derivation to divide the sentence/phrase pair into shorter phrase-pairs (this effectively avoids the sparsity problem).

Under this model each phrase pair gets some probability distribution $P_{\text{hier}}(\langle e,f \rangle : \theta_x, \theta_t)$, where $\theta_x$ and $\theta_t$ are the parameters of symbol distribution and phrase table respectively. The phrase table parameters $\theta_t$ are given by a PYP prior as

$$\theta_t \sim \text{PYP}(d, s, P_{\text{dac}})$$

where, $d$ and $s$ are discount and strength parameters. The base measure $P_{\text{dac}}$ adopts a "divide-and-conquer" strategy of recursively breaking up a longer phrase-pair into two shorter phrases through an ITG derivation. The entire generative process for begins from the full sentence pair (say $s$) and follows the script given below.

1. Generate the entire phrase-pair $s$ from the phrase-table distribution $P_t$. Now fall back to break the phrase-pair through ITG-style derivations employing $P_{\text{dac}}$

2. Decide the ITG derivation type $I_d$ from symbol distribution $\theta_x$ which can be BASE, REG or INV

   (a) If $I_d = \text{BASE}$, directly generate a new terminal phrase-pair from $P_{\text{base}}$, based on IBM Model 1 word alignment probabilities, defined similar to (DeNero et al., 2008)

   (b) If $I_d = \text{REG}$, recursively generate smaller biphrases $\langle e_1, f_1 \rangle$ and $\langle e_2, f_2 \rangle$ from $P_{\text{hier}}$ and concatenate them as $\langle e_1 e_2, f_1 f_2 \rangle$

   (c) If $I_d = \text{INV}$, recursively generate smaller biphrases $\langle e_1, f_1 \rangle$ and $\langle e_2, f_2 \rangle$ from $P_{\text{hier}}$ and concatenate them as $\langle e_1 e_2, f_2 f_1 \rangle$

    For inference it uses a sentence-level block sampler exploring the space of ITG-phrase alignments. In order to reduce the time complexity in sampling, it uses a heuristic beam search approximation that prunes the alignment spans based on a probability threshold (see Neubig et al. (2011) for details).
2.2 Grammar Extraction

After extracting the phrasal alignments in the previous step, we now need to learn hierarchical translation grammar along with the rule parameters. We have chosen the Bayesian model proposed by (Sankaran et al., 2013) for this step because, unlike other Bayesian models, their model can infer a Hiero-style grammar\(^1\) that can be used directly by a hierarchical phrase-based decoder.

Their model assumes the existence of initial phrase-pairs obtained from bidirectional symmetrization of word alignments (traditional SMT training pipeline). In our case these are obtained by the earlier step. The model generates an aligned phrase pair \(x\) from the hierarchical translation rules using the following two-step generative story.

1. First decide the derivation type \(z_d\) for generating the aligned phrase pair \(x\). It can either be a terminal derivation or hierarchical derivation with one/two gaps,\(^2\) i.e. \(z_d = \{\text{TERM, HIER-A1, HIER-A2}\}\). Following Chiang (2007), we allow a maximum of two gaps or two non-terminals in the SCFG because the hierarchical phrase-based decoder becomes prohibitively computationally expensive with more than two non-terminals.

2. Identify the constituent rules \(r\) in the derivation to generate the phrase pair.

\[
\begin{align*}
\phi^z & \sim \text{Dirichlet}(\alpha_z) & & \text{[draw derivation type parameters]} \\
\theta & \sim \text{Dirichlet}(\alpha_h, p_0) & & \text{[draw rule parameters]} \\
\phi^z & \sim \text{Dirichlet}(\alpha_z) & & \text{[decide the derivation type]} \\
r | r \in d_x & \sim \text{Multinomial}(\phi^z) & & \text{[generate rules deriving phrase-pair \(x\)]}
\end{align*}
\]

Figure 3: SCFG Extraction model: Definition

Under this model the probability of a particular derivation \(d \in \phi_x\) for a given phrase pair \(x\) will be given by:

\[
p(d) \propto p(z_d) \prod_{r \in d} p(r|G, \theta)
\]

(2)

where \(r\) is a rule in grammar \(G\) and \(\theta\) is the grammar parameter.

Figure 3 depicts the generative story of this model, while its corresponding graphical representation is shown in Figure 4. The derivation-type \(z_d\) is sampled from a multinomial distribution parameterized by \(\phi^z\), where \(\phi^z\) is distributed itself by a Dirichlet distribution with hyper-parameter \(\alpha_z\). The grammar rules are generated from a multinomial distribution parameterized by \(\theta\), where \(\theta\) itself is distributed according to a Dirichlet distribution parameterized by a concentration parameter \(\alpha_h\) and a base distribution \(p_0\). For the base distribution, we again follow Sankaran et al. (2013) and use an informative prior based on geometric mean of the bidirectional alignment scores. This ensures that the model only considers the derivations that are

\(^{1}\)As we mentioned earlier most of the other Bayesian models are merely alignment models and employ an additional heuristic step for extracting Hiero-style grammar.

\(^{2}\)This refers to the maximum arity of a rule involved in the derivation.
Figure 4: Graphical model depicting SCFG rule extraction in phase-2. The generative process first decides the derivation type $z_j$ from a Multinomial parametrized by $\phi$. It then generates the rules $r_{kj}$ in the derivation by using a Dirichlet distribution $\theta$ with base measure $P_0$ and concentration parameter $\alpha$. There are $K$ rules in the derivation which yields the phrase-pair $x_j$, consistent with the underlying word alignments.\(^3\) This setting closely mimics the Hiero heuristic extraction approach (Chiang, 2007), which constrains the rule extraction to be consistent with the alignments.

Our goal in this phase is to infer the joint posterior $p(\theta, \Phi|\alpha_h, p_0, \alpha_z, \mathcal{X})$, where $\theta$ are the model parameters and $\Phi$ the latent derivations over all the phrase pairs. This could be factorized by using Variational approximation, yielding the posterior distributions $\theta$ (over grammar parameters) and $\Phi$ (over latent derivations).

$$p(\theta, \Phi|\alpha_h, p_0, \alpha_z, \mathcal{X}) \approx q(\theta|u)q(\Phi|\pi)$$

where $u$ and $\pi$ are the parameters of the variational distributions.

The inference is then performed in an EM-style algorithm, similar to Sankaran et al. (2013) - by iteratively updating the parameters $u$ and $\pi$. We initialize $u^0 := \alpha_hp_0$, which is then updated with expected rule counts in subsequent iterations. The expected count for a rule $r$ at time-step $t$ can be written as:

$$\mathbb{E}[p^t] = \sum_{d \in \phi_x} p(d|\pi^{t-1}, x)f_d(r)$$

where $p(d|\pi^{t-1}, x)$ is the probability of the derivation $d$ for the phrase pair $x$ and $f_d(r)$ is the frequency of the rule $r$ in derivation $d$. The $p(d,\cdot)$ term in Equation 3 can then be written in terms of $\pi$ as:

$$p(d|\pi^{t-1}, x) \propto p(z_d) \prod_{r \in d} \pi_r^{t-1}$$

The $p(d,\cdot)$ are normalized across all the derivations of a given phrase pair to yield probabilities. For each derivation type $z_d$, its expected count (at time $t$) is the sum of the probabilities of all

\(^3\)While a non-parametric prior would be better from a Bayesian perspective, we leave it for future consideration.
the derivations of its type.

\[ E[z_d^t] = \sum_x \sum_{\{z_d = z_{d'} \mid d' \in \phi_x\}} p(d'\mid \pi^{t-1}, x) \] (5)

We initialize the Dirichlet hyperparameters \( \alpha_{z_d} \) using a Gamma prior ranging between \( 10^{-1} \) and \( 10^{3} \): \( \alpha_{z_d} \sim \text{Gamma}(10^{-1}, 10^3) \). We run inference for a fixed number of iterations\(^4\) and use the grammar along with their posterior counts from the last iteration for the translation table.

### 2.3 Iterative Cascade Framework

The formulation of the framework with independent modules allow us to easily experiment with existing models for alignment and SCFG rule extraction. This also helps us quickly validate the effectiveness of our framework.

The many-to-many alignments extracted by pialign is directly fed to the rule extraction model in phase-2 for extracting the Hiero-style grammar. In the reverse direction, we could parse the sentences in the training set with the extracted Hiero-style grammar and use the resulting alignments to initialize the aligner in the next iteration. However, we decided to use a simple setting for this paper; hence we iterate the two steps of the iterative framework without the feedback in the reverse direction. Using the existing models as described above, we run the iterative cascade framework as below.

1. Run the alignment model described in Section 2.1 for a fixed burn-in iterations (set to 9) and collect alignment samples from the next iteration. (Phase-1)

2. Recompute lexical probabilities based on the current alignments for computing the prior in the phase-2.

3. Extract hierarchical translation grammar from the phrasal alignments that were obtained in phase-1 and using the Variational-Bayesian inference explained in Section 2.2. We run the VB inference for a fixed number (set to 10) of iterations. (Phase-2)

4. Repeat steps 1 through 3 for a small number of times (we use 3 runs) and at each iteration collect the samples independently.

Figures 5 and 6 depict the alignments and the extracted rules at the end of first and second iterations of the cascade framework. Each figure consists of three parts i) word alignments on the left, ii) alignment matrix in the middle and iii) the extracted rules on the right. It can be seen that the incorrect word alignment (marked in red in Figure 5) is correctly aligned at the end of second iteration in Figure 6.

At the end of 3 iterations of the cascade framework, we do model combination to aggregate the resulting Hiero-style grammars. The parameters for the rules are then estimated using relative frequency estimation as is done in the original Hiero rule extraction method.

---

\(^4\)In our experiments, we set the number of iterations to 10.
### 3 Experiments

We evaluate our iterative-cascade framework on two language pairs: Korean-English and Arabic-English. In both language pairs we limit the sentence length of the training set to 60 in order to run the Gibbs sampler in phase-1 efficiently (mainly due to the limitation of the Gibbs sampler employed by the aligner).

We use the Rochester corpus for Korean-English and remove the sentences longer than 60 words, resulting in about 52K sentence pairs for training. We retain 1118 sentence pairs each for tuning and testing. For Arabic-English, we randomly sample the ISI parallel-corpus to select 120K sentence pairs that satisfy the sentence length criterion. For tuning and testing, we use 1982 and 987 sentence pairs from the same corpus. The statistics of the two corpora are shown in Table 1.

<table>
<thead>
<tr>
<th>Lang.</th>
<th>Corpus</th>
<th>Train (# of words)</th>
<th>Dev/ test (# of sents)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Korean-English</td>
<td>University of Rochester</td>
<td>1.5M/ 1.4M</td>
<td>1118/ 1118</td>
</tr>
<tr>
<td>Arabic-English</td>
<td>ISI web-crawled</td>
<td>3.1M/ 3.3M</td>
<td>1982/ 987</td>
</tr>
</tbody>
</table>

Table 1: Hiero-style binary grammar extraction: Corpus statistics for iterative-cascade experiments. The sentences are restricted to have at most 60 words due to the limitation of the aligner.

We use our implementation of conventional Hiero system for training the baseline models and our CKY-style hierarchical phrase-based decoder for decoding in all our experiments. We use the respective systems for the two steps of our cascade-framework, i.e. pialign (Neubig et al., 2011) and Variational inference models. For experiments involving pialign, we ran the aligner for 10 iterations with 9 burn-in iterations. The samples were read off from the last iteration. For extracting SCFG grammar we use the initial phrase-pairs obtained by pialign and pass them through either the heuristic (Chiang, 2007) or the Variational-Bayesian (Section 2.2)
extractor. We tuned the feature weights using MERT and decoded the test set with the optimal weights. For language model, we use a 5-gram model trained on the gigaword corpus with Kneser-Ney smoothing using the SRILM toolkit. For the iterative-cascade setting, we iterate the two steps of the framework for three runs and do a sample combination to get the final grammar.

<table>
<thead>
<tr>
<th>Aligner</th>
<th>Extractor</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Giza++</td>
<td>Heuristic</td>
<td>7.97</td>
</tr>
<tr>
<td>Giza++</td>
<td>Var. Bayes</td>
<td>8.03</td>
</tr>
<tr>
<td>Pialign</td>
<td>Heuristic</td>
<td>7.70</td>
</tr>
<tr>
<td>Pialign</td>
<td>Var. Bayes</td>
<td>7.54</td>
</tr>
<tr>
<td></td>
<td>Iterative-cascade (3 iters)</td>
<td><strong>8.19</strong></td>
</tr>
</tbody>
</table>

Table 2: Iterative-cascade framework: Korean-English BLEU scores. For the iterative-cascade framework we ran Pialign and VB inferences for three iterations and did a sample combination. The BLEU scores that are less than the baseline Moses (Giza++, Heuristic) BLEU of 8.23 by a statistically significant margin are italicized. The best BLEU score is in boldface.

The results of the iterative-cascade inference are summarized in Tables 2 and 3 for Korean-English and Arabic-English settings respectively. We use four baseline hierarchical translation systems that arise from different combinations of the aligner and extractor as listed in the tables.

The first two baselines use Giza++ aligner and then use the two different (heuristic and VB) methods for extracting translation rules, which are then tuned/tested with our CKY-style decoder. Baselines 3 and 4 differ from the earlier ones in that, these baselines use pialign to generate many-to-many alignments. The last row corresponds to the iterative-cascade grammar setting, where we run the iterative inference three times and then aggregate the grammars.

In both language pairs, baselines employing pialign perform marginally worse and the first iteration of iterative-cascade model in fact results in statistically significant BLEU reduction compared to phrase-based baseline of 8.23. However when we run our cascade framework for three iterations, we see consistent BLEU score improvements ranging between 0.2 and 0.65 as compared to other baselines in the table.

One can also compare these scores to the phrase-based model for the sake of completeness. We consider two phrase-based models one using the regular heuristic training pipeline as Koehn et al. (2003) and the other using pialign. For pialign, we use the phrase table extracted by pialign and directly used it with Moses for tuning and decoding. Note that this baseline uses two additional features including span probability (see Neubig et al. (2011)) that are not used in the standard baseline or in the later models in the tables. The two phrase-based models obtained BLEU scores of 8.23 and 8.30 respectively and these are comparable to the performance of our iterative-cascade model.

Now turning our attention to the Arabic-English language pair we again notice a very similar behaviour as we saw for Korean-English. The only difference is that the scale of improvement is marginally less and our iterative-cascade framework improves the BLEU scores in the range of 0.25 and 0.5 over the other baselines. A phrase-based model using Moses achieves 25.34 BLEU score, while the pialign achieves 24.90.
<table>
<thead>
<tr>
<th>Aligner</th>
<th>Extractor</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Giza++</td>
<td>Heuristic</td>
<td>25.13</td>
</tr>
<tr>
<td>Giza++</td>
<td>Var. Bayes</td>
<td>25.20</td>
</tr>
<tr>
<td>Pialign</td>
<td>Heuristic</td>
<td>24.97</td>
</tr>
<tr>
<td>Pialign</td>
<td>Var. Bayes</td>
<td>25.09</td>
</tr>
<tr>
<td>Iterative-cascade (3 iters)</td>
<td>25.45</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Iterative-cascade framework: Arabic-English BLEU scores. For the iterative-cascade framework we ran Pialign and VB inferences independently for three runs and did a sample combination. The best BLEU score is in **boldface**.

4 Related work

*On models learning Alignments or Hiero-style grammar:* The potential incompatibility between the word alignments and the translation rules for the syntactic translation models have been noticed earlier (DeNero and Klein, 2007). Apart from showing the incompatibility, they also propose an unsupervised HMM alignment model that soft constrains the alignments conditioned on the target sentences and the corresponding (automatically generated) parse trees. The main difference is that our approach seek to improve alignments of different granularities and not just the word-level alignments.

Several other works have focussed on learning phrase alignments from synchronous derivations using non-lexicalized (Blunsom et al., 2008) or lexicalized (Hiero-style) ITG (Blunsom et al., 2009; Levenberg et al., 2012) rules and apply them for hierarchical phrase-based models. While these models extract ITG-style rules, they use them only for obtaining the alignment information. In other words, the extracted ITG-style rules are not directly used by a hierarchical translation decoder, which are in fact obtained from the alignments suggested by these rules. Thus the biggest drawback is that these models, strictly speaking, are alignment models and they use the heuristic rule extractor (Chiang, 2007) for learning the Hiero-style translation grammar.

In contrast to these, Sankaran et al. (2012, 2013) proposed a set of Bayesian models that directly learns the SCFG grammar. However these models only focus on the rule extraction part and rely on the heuristically extracted phrasal alignments. Instead our iterative-cascade framework simplifies the entire hierarchical translation training pipeline.

*On Joint models for PBMT:* Several works have exploited word alignments to improving the performance of parsing (Burkett and Klein, 2008; Snyder et al., 2009) outside the machine translation setting. In the reverse direction syntactic parsing has been used to get better alignments (May and Knight, 2007; DeNero and Klein, 2007; Fossum et al., 2008) in the context of machine translations.

Joint models for learning alignments and translation rules have been a fairly recent direction. A joint model using two syntactic parsers and combined with an ITG derivation to model alignments, enables the trees to diverge if required and otherwise encouraging the derivation to synchronize with the trees (Burkett et al., 2010). However it requires a parallel treebank and gold alignments to train on in addition to parsers for the source and target languages, thus
severely limiting its applicability. DeNero and Klein (2010) proposed a supervised model for extracting all overlapping bispans called extraction sets under a discriminative model by using phrase-level features in addition to the one-to-one alignments.

In contrast, Neubig et al. (2011) proposed an unsupervised hierarchical ITG model for jointly learning the alignments and translations as we explained in section 2.1. The extracted translation rules are then directly used in a phrase-based decoder. While, their alignments are based on the ITG, it uses with a flat (phrase-based) model for translation. We extend their approach through our iterative-cascade framework and extract SCFG rules in a separate phase.

Iterative approach has been used for directly training the phrase translation models for a phrase-based system (Wuebker et al., 2010). This method employs force decoding the training set (based on the IBM word alignments) to obtain phrase segmentations. Phrase probabilities are then estimated using leaving-one-out technique, in order to avoid overfitting. Our present work on learning hierarchical translation model differs from this in obvious way; additionally we just use model aggregation as opposed to their iterative force decoding.

Heger et al. (2010) extend the Wuebker’s work by combining the iteratively learned phrase alignments with the heuristically learned hierarchical translation model for Hiero-style system. This approach is similar in spirit to our goal of learning a hierarchical translation model that is consistent throughout. However their approach does not learn the hierarchical SCFG rules through force alignments, but only combines the iteratively learned phrase table with hierarchical translation grammar extracted traditionally. Secondly our framework allow both alignments and SCFG rules to be improved iteratively unlike theirs, where only the phrase alignments are improved. Another major difference is that we use Variational-Bayesian inference to aggregate rule counts globally as opposed to the leave-one-out counting.

On Joint models for (Hiero-style) ITG: Recently, Saers et al. (2013b,a,c) proposed a Bayesian maximum a posteriori (MAP) driven model for extracting bracketing inversion transduction grammar. This approach aims to improve coherence and model consistency between the training and test. Unlike Neubig et al. (2011), this line of work is motivated to employ same (lexicalized-ITG) model in both training and testing. While this approach is elegant, the emphasis on a single coherent model is too restrictive and is unable to integrate well with other feature functions such as better reordering or language models, while our method is able to do so.

Similar to Saers et al. (2013b,a,c), our iterative-cascade framework simplifies the training pipeline of the hierarchical phrase-based system in order to obtain novel alignments and translation rules to be used in the SMT decoder. However, their approach stands separate from conventional phrase-based and hierarchical phrase-based SMT models. In contrast, while our approach does compute new alignments and translation rules, it can also be combined with some of the recent advancements in SMT, for example in reordering model and language model.

Finally many of the previous work on Bayesian grammar induction are trained and tested on datasets that have simple and short sentences (Blunsom et al., 2008; Saers et al., 2013b,a,c). Typically they use the IWSLT Chinese-English corpus consisting of sentences in the travel domain, where the average sentence length on English side is around 7 words. On the other hand,
we use realistic datasets with fairly long (average sentence length > 27 words) and complex sentences.

5 Conclusion and Future Directions

Our iterative-cascade framework reduces the serial, multi-component and heuristic-ridden Hiero training pipeline with a simple two-step iterative pipeline. The simplicity of the framework further enables any appropriate model to be plugged in for the alignment and rule extraction steps. Validation experiments with existing models demonstrate small but consistent gains over the traditional Hiero training baseline involving heuristic steps for two language pairs. Further the resulting synchronous context-free grammar has a sparse distribution, where the probability mass is concentrated on few rules unlike the flat distribution of rules generated by the conventional pipeline. Unlike the earlier research on Hiero-style Bayesian grammar induction (Blunsom et al., 2008, 2009; Levenberg et al., 2012), the grammar induced by our iterative-cascade framework are directly used by the CKY decoder.

A minor shortcoming of our work relates to the smaller training corpus we use for the experiments in this paper. However, as we noted earlier our experimental results are based on realistic SMT datasets that contain longer and complex sentences. This is unlike some earlier approaches that rely on some corpus consisting of shorter sentences with much simpler structure, where a large number of the sentences might share the same structure due to the nature of the domain. Further, we intend to address this shortcoming in near future (see below).

As a future work, we are currently working on adding the feedback loop to improve the alignments by using the information from the hierarchical translation grammar extracted in the second step. One could use the simple approach of parsing the training corpus using the SCFG extracted in the second step of iterative-cascade and use the resulting alignments to initialize the aligner in the next iteration. We are also exploring other approaches for doing this. Secondly, we also intend to replace the current ITG aligner to avoid the sampling issues due to the approximation employed by its beam search sampler. This would enable us to run experiments on large parallel corpora for better validation of our iterative-cascade framework.

Acknowledgments

We would like to thank Gholamreza Haffari, Adam Lopez and Taro Watanabe for discussions and useful suggestions.

References


Coarse “split and lump” bilingual language models for richer source information in SMT

Darlene Stewart  
Darlene.Stewart@nrc.gc.ca  
Roland Kuhn  
Roland.Kuhn@nrc.gc.ca  
Eric Joanis  
Eric.Joanis@nrc.gc.ca  
George Foster*  
George.Foster@nrc.gc.ca  
All authors originally at: National Research Council, Ottawa, Canada K1A 0R6  
*This author is now at Google Inc, Mountain View, California 94043

Abstract

Recently, there has been interest in automatically generated word classes for improving statistical machine translation (SMT) quality: e.g., (Wuebker et al., 2013). We create new models by replacing words with word classes in features applied during decoding; we call these “coarse models”. We find that coarse versions of the bilingual language models (biLMs) of Niehues et al. (2011) yield larger BLEU gains than the original biLMs. BiLMs provide phrase-based systems with rich contextual information from the source sentence; because they have a large number of types, they suffer from data sparsity. Niehues et al. (2011) mitigated this problem by replacing source or target words with parts of speech (POSs). We vary their approach in two ways: by clustering words on the source or target side over a range of granularities (word clustering), and by clustering the bilingual units that make up biLMs (bitoken clustering). We find that loglinear combinations of the resulting coarse biLMs with each other and with coarse LMs (LMs based on word classes) yield even higher scores than single coarse models. When we add an appealing “generic” coarse configuration chosen on English > French devtest data to four language pairs (keeping the structure fixed, but providing language-pair-specific models for each pair), BLEU gains on blind test data against strong baselines averaged over 5 runs are +0.80 for English > French, +0.35 for French > English, +1.0 for Arabic > English, and +0.6 for Chinese > English.

1. Introduction

This work aims to provide rich contextual information to phrase-based SMT, in order to mitigate data sparsity. We cluster the basic units of the bilingual language model (biLM) of Niehues et al (2011) and of standard language models (LMs). A “generic”, symmetric configuration chosen on English > French devtest yields BLEU gains over strong baselines on blind test data of +0.80 for English > French (henceforth “Eng-Fre”), +0.35 for French > English (“Fre-Eng”), +1.0 for Arabic > English (“Ara-Eng”), and +0.6 for Chinese > English (“Chi-Eng”). If we apply the configuration with the highest devtest score on a given language pair to blind data, the gains are +0.85 for Eng-Fre, +0.46 for Fre-Eng, and +1.2 for Ara-Eng, but still +0.6 for Chi-Eng.

1.1. Coarse bilingual language models (biLMs) for source context

Though coarse biLMs are the focus of this paper, we explored other coarse models: models where words are replaced by word classes. E.g., we obtained good gains in earlier experiments with coarse LMs. Since others have explored that terrain before us (see 1.2), this paper focuses on the “bilingual language model” (biLM) of Niehues et al (2011). In phrase-based
SMT, information from source words outside the current phrase pair is incorporated only indirectly, via target words that are translations of these source words, if the relevant target words are close enough to the current target word to affect LM scores. BiLMs address this by aligning each target word in the training data with source words to create “bitokens”. An N-gram bitoken LM is then trained. A coarse biLM is one whose words and/or bitokens have been clustered into classes. Our best results were obtained by combining coarse biLMs with coarse LMs. We tune our system with batch lattice MIRA (Cherry and Foster, 2012), which supports loglinear combinations that have many features.

**Figure 1** shows word-based and coarse biLMs for Eng→Fre. A target word and its aligned source words define a bitoken. Unaligned target words (*e.g.*, French word “d’” in the example) are aligned with NULL. Unaligned source words (*e.g.*, “very”) are dropped. A source word aligned with more than one target word (*e.g.*, “we”, aligned with two instances of “nous”) is duplicated: each target word aligned with it receives a copy of that source word.

![Diagram](image)

**Figure 1.** Creating bitokens & bitoken classes for a bilingual language model (biLM)

BiLMs can easily be incorporated into a phrase-based architecture. The decoder still uses phrase pairs from a phrase table to create hypotheses. However, a new LM with a wide context span of source information can now score hypotheses, along with the standard LM (Niehues et al found it was best to retain the latter). Unfortunately, the bitoken vocabulary of a biLM will be much bigger than the target-language vocabulary, because a target word is often split into different bitokens. *E.g.*, the word “être” might be split into three bitokens: “être_be”, “être_being”, and “être_to-be”. One solution to the sparsity problem is to lump bitokens into new classes. *E.g.*, one could replace each English or French word above with its part of speech (POS). In (Niehues et al, 2011), this “split and lump” process was applied to
both sides of a biLM for Ara>Eng SMT. When the biLM was added as a new loglinear feature to a system with a word-based biLM, it yielded a modest gain of about +0.2 BLEU. In our work, we try other versions of “split and lump”. Instead of using taggers to define POSs, we use a program called *mkcls* (see 1.2) to create clusters. Unlike Niehues et al. (2011), we vary the granularity of word clustering, and sometimes cluster the bitokens themselves, calling the resulting models “coarse biLMs”.

*Figure 1* also shows three ways of building coarse biLMs: 1. clustering source and/or target words, then creating bitokens. 2. clustering the word-based bitokens themselves, with *mkcls* using bitoken perplexity as its criterion (in *Fig. 1*, “bitoken clustering 1”). 3. clustering bitokens whose source and/or target words have been “preclustered” (“bitoken clustering 2”). Here, E1, E2, etc., and F1, F2, etc., are word classes generated by *mkcls* operating on source (English) and target (French) text respectively; B1, B2, etc. denote bitoken classes.

**Figure 2.** Two-pass construction of bitokens for a coarse biLM

---

*Figure 2* illustrates the three types of coarse biLM, each shown by a dotted-line oval. A key aspect of a biLM is its bitoken vocabulary size. *E.g.*, for the Eng>Fre experiments on which *Figure 2* is based, the original word biLM had a vocabulary of 7.6 million bitokens. Coarse biLMs result from two passes of bitoken vocabulary compression, both optional: a first pass of clustering of source and/or target words, and a second pass of bitoken clustering. Skipping both passes yields the word-based biLM. The X coordinates in *Figure 2* give the biLM vocabulary size after the first pass, and the Y coordinates give its size after the second pass (as a percentage of its original size). The original biLM (“word biLM”) is in the upper right-hand corner: both passes are null operations, so the coordinates are (100%,100%).
We denote word clustering by \((n_1, n_2)\), where \(n_1\) and \(n_2\) are number of source and target word classes respectively. \(|S|\) and \(|T|\) are the original sizes of the source and target vocabularies. The top oval in Figure 2 contains coarse biLMs obtained by pass 1 (word clustering) but not pass 2 (bitoken clustering). E.g., “(400, 200)” is the coarse biLM obtained by using 400 and 200 word classes for English and French respectively; “(1600, |T|)” is the coarse biLM obtained with 1600 English classes and no clustering for French. The oval on the far right contains biLMs created when only pass 2 is applied. E.g., “50 bi” is the coarse biLM obtained by clustering the 7.6M word-based bitokens down to only 50. The third oval contains coarse biLMs produced by applying pass 1, then pass 2. E.g., “400 bi(400,200)” is the coarse biLM obtained by creating bitokens with 400 and 200 source and target word classes respectively, then clustering these bitokens into 400 classes. A defect of the figure is that its axes don’t represent the difference between word clustering on the source vs. target sides. However, the figure conveys our greatest problem: the vast number of possible coarse biLMs.

There is a big space in the middle of the figure that wasn’t explored in our experiments: there are no final biLMs that have between 0.01% and 10% of the original number of word biLMs, because mkcls becomes very slow as the number of word classes grows: creating coarse biLMs like “(10,000, 10,000)” or “10,000 bi(400,400)” is infeasible.

1.2. Related work

This section will discuss work on coarse models, source-side contextual information for SMT, and lexical clustering techniques (including mkcls, used for our experiments).

Uszkoreit and Brant (2008) explored coarse LMs for SMT. Wuebker et al (2013) describe coarse LMs, translation models (TM), and reordering models (RM). Best performance was obtained with a system containing both word-based and coarse models. Prior to our current work, we experimented with discriminative hierarchical RMs (DHRMs) (Cherry, 2013). These combine the hierarchical RM (HRM) of (Galley and Manning, 2008) with sparse features conditioned on word classes for phrases involved in reordering; word classes are obtained from mkcls. Like Cherry (2013), we found that DHRM outperformed the HRM version for Ara->Eng and Chi->Eng. However, experiments with English-French Hansard data showed only small gains for DHRM over HRM. Thus, while all the Ara->Eng and Chi->Eng experiments reported in this paper employ DHRM - a coarse reordering model - none of the Eng<->Fre experiments do. In prior experiments, we also studied coarse phrase translation models, but unlike Wuebker et al (2013), we found they did not yield significant improvements to our system, except when there is little training data. Many experiments in this paper involve coarse language models. These are particularly effective for morphologically rich languages (e.g., Ammar et al, 2013; Bisazza and Monz, 2014). In unpublished earlier experiments, we found that coarse LM combinations can yield better results than using just one.

Besides biLMs (Niehues et al, 2011), there are other ways of incorporating additional source-language information in SMT. These include spectral clustering for HMM-based SMT (Zhao, Xing and Waibel, 2005), stochastic finite state transducers based on bilingual ngrams (Casacuberta and Vidal, 2004; Mariño et al, 2006; Crego and Yvon, 2010; Zhang et al, 2013), the lexicalized approach of (Hasan et al, 2008), factored Markov backoff models (Feng et al, 2014) and the “operation sequence model” (OSD) of (Durrani et al, 2011 and 2014). (Durrani et al 2014) appeared after our current paper was submitted. Our work and theirs shares an underlying motivation in which mkcls is applied to make earlier models more powerful, though the OSD models and ours are very different. We chose to implement biLMs primarily because this is easy to do in a phrase-based system.

Automatic word clustering was described in (Jelinek, 1991; Brown et al, 1992). In “Brown” or “IBM” clustering, each word in vocabulary \(V\) initially defines a single class.
These classes are clustered bottom-up into a binary tree of classes, with classes iteratively merging to minimize text perplexity under a class-based bigram LM, until the desired number of classes $C$ is attained. Martin, Liermann, and Ney (1998) describe “exchange” clustering. Each word in $V$ is initially assigned to a single class in some fashion. Then, each word in turn is reassigned to a class so as to minimize perplexity; movement of words between classes continues until a stopping criterion is met. When these authors compared various word clustering methods, the perplexity results were almost identical. The lowest bigram perplexity is usually obtained when each of the most frequent words is in a different class; different word clustering methods typically all ended up with arrangements similar to this. These authors obtained their best perplexity and speech recognition results when the clustering criterion was based on trigrams as well as bigrams, but this makes clustering expensive. Och (1999) focuses on bilingual word clustering and discusses ideas similar to bitoken clustering, though not in the context of phrase-based SMT. Uszkoreit and Brant (2008) describe a highly efficient distributed version of exchange clustering. Faruqui and Dyer (2013) propose a bilingual word clustering method whose objective function combines same-language and cross-language mutual information. Applied to named entity recognition (NER), this yields significant improvements. Turian, Ratinov and Bengio (2010) apply Brown clustering to NER and chunking. Finally, Blunsom and Cohn (2011) improve Brown clustering by using a Bayesian prior to smooth estimates, by incorporating trigrams, and by exploiting morphological information.

For word clustering, we chose a widely used program, mkcls: Blunsom and Cohn (2011) note its strong performance. We could have used POSs, but they have definitions that vary across languages; mkcls can be applied in a uniform way (though with the disadvantage that it gives each word a fixed class, instead of several possible classes as with POSs). Niesler et al (1998) found that automatically derived word classes outperform POSs. Until recently, the only document describing mkcls was in German (Och, 1995); accurate English information was unavailable. It is often suggested that mkcls implements (Och, 1999), but this is only partly true. Fortunately, Dr. Chris Dyer now provides an accurate description on his blog: http://statmt.blogspot.ca/2014/07/understanding-mkcls.html. Basically, mkcls executes an ensemble of optimizers and merges their results; the criterion for all steps is minimal bigram perplexity. Dr. Dyer estimates that the perplexity of an LM built from the resulting word classes is typically 20-40% lower than for bottom-up Brown clustering on its own.

2. Experiments

2.1. Experimental approach

Using four diverse large-scale machine translation tasks (Eng>Fre, Fre>Eng, Ara>Eng, Chi>Eng), we studied the impact of coarse BiLMs in isolation and in combination with coarse LMs. Our challenge was to explore the most interesting possibilities without doing innumerable experiments. Ammar et al (2013) note that coarse models are particularly effective when the target language has complex morphology. We thus decided to use Eng>Fre experiments on devtest data to carry out initial explorations: this language pair would be a sensitive one. Our metric was average BLEU over Devtest1 and Devtest2 for a Hansard system (see Table 1). There is insufficient space to report all these Eng>Fre devtest experiments. The first round of experiments made us decide to explore coarse LMs and coarse BiLMs, but not coarse TMs (these only gave appreciable gains for small amounts of training data); we would employ Witten-Bell smoothing for the coarse models (coarse models generate counts of counts that the SRILM implementation of Kneser-Ney can’t cope with, and Witten-Bell slightly outperformed Good-Turing for coarse models); we would use 8-gram coarse models (results differed only slightly along the range from 6-grams to 8-grams, but were marginally better for 8-
grams). We then began a second round of Eng>Fre experiments with the same devtest (see 2.4 and Table 4); the results informed all subsequent experiments.

### 2.2. Experimental data

For English-French experiments in both directions, we used the high-quality Hansard corpus of Canadian parliamentary proceedings from 2001-2009 (Foster et al, 2010). We reserved the most recent five documents (from December 2009) for development and testing material, and extracted the dev and test corpora shown in Table 1. Some of the documents were much larger than typical devtest sizes, so we sampled subsets of them for the dev and test sets.

<table>
<thead>
<tr>
<th>Corpus</th>
<th># sentence pairs</th>
<th># words (English)</th>
<th># words (French)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>2.9M</td>
<td>60.5M</td>
<td>68.6M</td>
</tr>
<tr>
<td>Tune</td>
<td>2.002</td>
<td>40K</td>
<td>45K</td>
</tr>
<tr>
<td>Devtest1</td>
<td>2.148</td>
<td>43K</td>
<td>48K</td>
</tr>
<tr>
<td>Devtest2</td>
<td>2.166</td>
<td>45K</td>
<td>50K</td>
</tr>
<tr>
<td>Test1 (blind test)</td>
<td>1,975</td>
<td>39K</td>
<td>44K</td>
</tr>
<tr>
<td>Test2 (blind test)</td>
<td>2,340</td>
<td>49K</td>
<td>55K</td>
</tr>
</tbody>
</table>

Table 1. Corpus sizes for English<>French Hansard data

For Ara>Eng and Chi>Eng, we used large-scale training conditions defined in the DARPA BOLT project; Tables 2 and 3 give statistics. For Arabic, “all” includes 15 genres and MSA/Egyptian/Levantine/untagged dialects; “small” is “all” minus UN data; “webforum” is the webforum subset of “small”. Tune, Test, and SysCombTune are webforum genre, and a similar dialect mix. For Chinese all training sets are mixed genre; “good” is “all” minus UN, HK, and ISI data; Tune, SysCombTune, and Test are forum genre. NIST Open MT 2012 test data was the held-out data; for Arabic it is a mix of weblog/newsgroup genres; for Chinese it contains these two genres plus unknown genre. In Tables 2 and 3, the number of English words for Tune, Devtest1, Devtest2, and Test is averaged over four references.

<table>
<thead>
<tr>
<th>Corpus</th>
<th># sentence pairs</th>
<th># words (Arabic)</th>
<th># words (English)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train1: “all”</td>
<td>8.5M</td>
<td>261.7M</td>
<td>207.5M</td>
</tr>
<tr>
<td>Train2: “small”</td>
<td>2.1M</td>
<td>42.4M</td>
<td>37.2M</td>
</tr>
<tr>
<td>Train3: “webforum”</td>
<td>92K</td>
<td>1.6M</td>
<td>1.8M</td>
</tr>
<tr>
<td>Tune</td>
<td>4,147</td>
<td>66K</td>
<td>72K</td>
</tr>
<tr>
<td>Devtest1: “Test”</td>
<td>2,453</td>
<td>37K</td>
<td>40K</td>
</tr>
<tr>
<td>Devtest2: “SysCombTune”</td>
<td>2,175</td>
<td>35K</td>
<td>38K</td>
</tr>
<tr>
<td>Test (blind): MT12 Arabic test</td>
<td>5,812</td>
<td>229K</td>
<td>209K</td>
</tr>
</tbody>
</table>

Table 2. Corpus sizes for tokenized Arabic-English data

<table>
<thead>
<tr>
<th>Corpus</th>
<th># sentence pairs</th>
<th># words (Chinese)</th>
<th># words (English)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train1: “all”</td>
<td>12M</td>
<td>234M</td>
<td>254M</td>
</tr>
<tr>
<td>Train2: “good”</td>
<td>1.5M</td>
<td>32M</td>
<td>38M</td>
</tr>
<tr>
<td>Tune</td>
<td>2,748</td>
<td>62K</td>
<td>77K</td>
</tr>
<tr>
<td>Devtest1: “Test”</td>
<td>1,224</td>
<td>29K</td>
<td>36K</td>
</tr>
<tr>
<td>Devtest2: “SysCombTune”</td>
<td>1,429</td>
<td>31K</td>
<td>38K</td>
</tr>
<tr>
<td>Test (blind): MT12 Chinese test</td>
<td>8,714</td>
<td>224K</td>
<td>261K</td>
</tr>
</tbody>
</table>

Table 3. Corpus sizes for tokenized Chinese-English data
2.3. Experimental systems

Eng<>Fre Hansard experiments were performed with Portage, the National Research Council of Canada’s phrase-based system (this is the system described in Foster et al., 2013). The corpus was word-aligned with HMM and IBM2 models; the phrase table was the union of phrase pairs from these alignments, with a length limit of 7. We applied Kneser-Ney smoothing to find bidirectional conditional phrase pair estimates, and obtained bidirectional Zens-Ney lexical estimates (Chen et al., 2011). Hierarchical lexical reordering (Galley and Manning, 2008) was used. Additional features included standard distortion and word penalties (2 features) and a 4-gram LM trained on the target side of the parallel data: 13 features in total. The decoder used cube pruning and a distortion limit of 7.

Our Chinese and Arabic baselines are strong phrase-based systems, similar to our entries in evaluations like NIST. The hierarchical lexical reordering model (HRM) of (Galley and Manning, 2008) along with the sparse reordering features of (Cherry, 2013) was used. Phrase extraction pools counts over symmetrized word alignments from IBM2, HMM, IBM4, Fastalign (Dyer et al., 2013), and forced leave-one-out phrase alignment; the HRM pools counts in the same way. Phrase tables were Kneser-Ney smoothed as for the Eng<>Fre experiments, and combined with mixture adaptation (Foster, 2007); indicator features tracked which extraction techniques produced each phrase. The Chinese system incorporated additional adaptation features (Foster et al., 2013). For both Arabic and Chinese, four LMs per system were trained: one LM on the English Gigaword corpus (5-gram with Good-Turing smoothing), one LM on monolingual webforum data and two LMs trained on selected material from the parallel corpora (4-gram with Kneser-Ney smoothing); in the case of Chinese, the latter two LMs were mixture-adapted. Both systems used the sparse features of (Hopkins and May, 2011; Cherry, 2013). The decoder used cube pruning and a distortion limit of 8.

Tuning for all systems was performed with batch lattice MIRA (Cherry and Foster, 2012). The metric is the original IBM BLEU, with case-insensitive matching of n-grams up to n = 4. For all systems, we performed five random replications of parameter tuning (Clark et al., 2011).

For Eng<>Fre, coarse models were trained on all of “Train”. For Ara>Eng, word classes and two static coarse LMs were trained on “all” and “webforum” (no linear mixing), but biLMs were trained on “small”. For Chi>Eng, word classes and a large static mix coarse LM were trained on “all”, but a smaller dynamic mix coarse LM and all the biLMs were trained on “good”. Bitokens for all language pairs were derived from word-aligned sentence pairs by two word alignment techniques. Two copies of each pair were made; one was aligned using HMMs, the other using IBM2. Since the Arabic and Chinese phrase tables were created not only with these alignment techniques, but with others, the decoder for these languages may use bitokens not found in the biLMs (“out-of-biLM-vocabulary” bitokens).

2.4. English > French experiments

First, we explored single Eng>Fre coarse models on devtest data. For models involving word classes, we looked at 50, 100, 200, 400, 800, and 1600 classes. The number of bitoken types was huge (7.6M), so we were only able to obtain up to 800 bitoken classes (generating 1600 classes would have taken too long). There is insufficient space to show all results: Table 4 shows how many variants of each coarse model type we tried, along with the lowest-scoring and highest-scoring variant of each type (using average score on Devtest1 and Devtest2). The table uses the notation for biLMs given towards the end of section 1.1. The range of scores from lowest to highest is shown with the standard deviation (SD) over five tuning runs (more precisely, we show whichever is greater, the SD of the lowest or of the highest score).
System type (#variants tried) | Lowest scorer | Highest scorer | Range of scores | Range of gains
--- | --- | --- | --- | ---
Cluster src biLM (6) | (800, [T]) | (400, [T]) | 40.20-40.31 ±0.06 | +0.08-0.19
Cluster tgt biLM (6) | (|S|, 50) | (|S|, 100) | 40.33-40.41 ±0.04 | +0.21-0.29
Cluster src & tgt biLM | (1600, 1600) | (400, 200) | 40.32-40.66 ±0.05 | +0.20-0.54
Cluster bitoken biLM (5) | 50 bi | 800 bi | 40.30-40.71 ±0.02 | +0.18-0.59
Clstr src/tgt → clstr bitok. biLM (11) | 400 bi(50,50) | **400 bi(400,400)** | 40.51-40.76 ±0.01 | +0.39-0.64
Word-based biLM = (|S|, [T]) (1) | N/A | N/A | 40.34 ±0.05 | +0.22
Baseline for biLMs (1) | N/A | N/A | 40.12 ±0.02 | (0.0)
Coarse LM (7) | 50 tgt classes | 800 tgt classes | 40.31-40.53 ±0.02 | +0.30-0.52
*(Baseline for coarse LMs)* (1) | N/A | N/A | 40.02 ±0.03 | (0.0)

**Table 4.** Preliminary experiments - Eng>Fre average(Devtest1,Devtest2) BLEU for single coarse models

We did not try all 36 combinations of source and target word clusters, but explored along the “diagonal” where the number of classes is the same for both sides: *i.e.*, we tried (50, 50), . . . , (1600, 1600). Then we tried coarse biLMs in the neighbourhood of the best diagonal ones, eventually trying 22 different biLMs clustered on both sides. For bitoken clustering, we also carried out this kind of greedy search. Word preclustering shortens the time required for bitoken clustering. *E.g.*, on our machines, training the highest-scoring biLM, 400 bi(400,400), took 19 hours for (400,400) preclustering and then 112 hours for bitoken clustering: 131 hours total. Training “400 bi” with no preclustering took 277 hours. The shorter time with preclustering is because mkcls takes time proportional to the number of types: 7.6M bitokens without preclustering but only 3.2M with (400, 400) preclustering. **Table 4** shows that preclustering followed by bitoken clustering also yielded the best results: the worst-scoring biLM of this type performed about +0.4 better than the baseline, and the best-performing one gained more than +0.6. The last two rows of the table pertain to coarse LMs. The **Table 4** experiments were performed two months earlier than the rest, with a slightly different version of the system, one with a standard rather than hierarchical lexicalized reordering model (HRM) (tables 5 & 6 below show results with HRMs, and tables 7 & 8 with discriminative HRMs (DHRMs)).

Next, we explored loglinear combinations of the highest-scoring coarse models on the same devtest, again doing a kind of greedy search (and using HRMs). Because of the poor results for clustering on only source or target words (not both) in **Table 4**, we did not try these biLMs in combinations. Almost all the combinations we tried scored significantly higher than the coarse models of which they were composed, as shown in **Table 5** (in descending order of devtest score). A pattern emerged: many of the highest-scoring combinations had one coarse biLM with clustered bitokens (sometimes preclustered, sometimes not), one coarse biLM with clustered source and target words, and two or three coarse LMs of very different granularity. Presumably, these information sources complement each other. We chose one of the configurations that scored highest on Eng>Fre devtest to be tried on all four language pairs – in each case, keeping the structure but using language-pair-specific models. This “generic” configuration often scores lower for other language pairs on blind test data than combinations that have
been chosen via experiments on devtest data for a given pair, but is a reasonable choice for system builders who don’t wish to spend a lot of time on preliminary experiments.

In addition to “generic” (underlined) and the two other best combinations, Table 5 shows individual models inside coarse model combinations (in italics), the word-based biLM, and the baseline (the notation here was defined in section 1.1 above). Average scores and standard deviations from five runs are shown. For individual biLMs, “[B]” is the number of bitoken types. E.g., clustering English and French words to 400 and 200 classes respectively shrinks the number of different bitokens from 7.6 million to 2.9 million. Results on blind test data (Test1 and Test2) are lower than on devtest but in roughly the same order; coarse model combinations score higher than their components on both devtest and test data. The “generic” configuration we chose was the one with symmetric biLMs (no difference between number of source and target word classes in its biLMs) that scores highest on devtest. It has a biLM clustered to 400 classes for each language, a biLM obtained from the former by clustering it to 400 bitokens, and two coarse LMs of very different granularities (100 and 1600 classes).

Table 5. Eng>Fre BLEU for coarse model combinations (single components in italics)

<table>
<thead>
<tr>
<th>System</th>
<th>avg(Devtest1, Devtest2)</th>
<th>Gain on devtest</th>
<th>avg(Test1, Test2)</th>
<th>Gain on test</th>
</tr>
</thead>
<tbody>
<tr>
<td>400bi(400,400)&amp;(400,200)&amp;100tgt&amp;1600tgt</td>
<td>41.25±0.03</td>
<td>+1.13</td>
<td>42.64±0.02</td>
<td>+0.85</td>
</tr>
<tr>
<td>400bi(400,400)&amp;(400,400)&amp;100tgt&amp;1600tgt = “Generic”</td>
<td>41.19±0.01</td>
<td>+1.07</td>
<td>42.59±0.03</td>
<td>+0.80</td>
</tr>
<tr>
<td>200bi&amp;(400,200)&amp;100tgt&amp;1600tgt</td>
<td>41.19±0.02</td>
<td>+1.07</td>
<td>42.60±0.04</td>
<td>+0.81</td>
</tr>
<tr>
<td>Single biLM: 400bi400src400tgt,</td>
<td></td>
<td></td>
<td>40.76±0.01</td>
<td>±0.64</td>
</tr>
<tr>
<td>Single biLM: 200 bi,</td>
<td></td>
<td></td>
<td>40.69±0.03</td>
<td>±0.57</td>
</tr>
<tr>
<td>Single biLM: (400,200),</td>
<td></td>
<td></td>
<td>40.66±0.02</td>
<td>+0.54</td>
</tr>
<tr>
<td>Single biLM: (400,400),</td>
<td></td>
<td></td>
<td>40.51±0.04</td>
<td>+0.39</td>
</tr>
<tr>
<td>Single biLM: word-based,</td>
<td></td>
<td></td>
<td>40.34±0.05</td>
<td>+0.22</td>
</tr>
<tr>
<td>Single coarse LM: 1600tgt</td>
<td>40.67±0.03</td>
<td>+0.55</td>
<td>42.36±0.03</td>
<td>+0.56</td>
</tr>
<tr>
<td>Single coarse LM: 100tgt</td>
<td>40.61±0.03</td>
<td>+0.49</td>
<td>42.11±0.02</td>
<td>+0.32</td>
</tr>
<tr>
<td>Baseline</td>
<td>40.12±0.02</td>
<td>(0.0)</td>
<td>41.79±0.02</td>
<td>(0.0)</td>
</tr>
</tbody>
</table>

2.5. Experiments with other language pairs

Experiments with the other language pairs were carried out as with Eng>Fre; greedy search over single coarse models followed by greedy search over model combinations, using scores on devtest for each pair to make decisions. Results on devtest and blind test data are shown in Tables 6 – 8 for the two combinations scoring highest on devtest for each pair. For each pair, we also tested the “generic” configuration (underlined) chosen on the basis of Eng>Fre devtest results. All results shown are averaged over 5 tuning runs.
### Table 6. Fre>Eng BLEU for coarse model combinations (single components in italics)

<table>
<thead>
<tr>
<th>System</th>
<th>avg(Devtest1, Devtest2)</th>
<th>Gain on devtest</th>
<th>avg(Test1, Test2)</th>
<th>Gain on test</th>
</tr>
</thead>
<tbody>
<tr>
<td>400bi&amp;(1600,1600)&amp;100tgt&amp;800tgt&amp;1600tgt</td>
<td>40.80±0.03</td>
<td>+0.79</td>
<td>42.49±0.04</td>
<td>+0.46</td>
</tr>
<tr>
<td>400bi&amp;(1600,1600)&amp;100tgt&amp;1600tgt</td>
<td>40.79±0.03</td>
<td>+0.78</td>
<td>42.47±0.05</td>
<td>+0.45</td>
</tr>
<tr>
<td>400bi(400,400)&amp;(400,400)&amp;100tgt&amp;1600tgt = &quot;Generic&quot;</td>
<td>40.67±0.03</td>
<td>+0.66</td>
<td>42.37±0.03</td>
<td>+0.35</td>
</tr>
<tr>
<td>Single biLM: 400bi,</td>
<td>B</td>
<td>=400</td>
<td>40.41±0.02</td>
<td>+0.39</td>
</tr>
<tr>
<td>Single biLM: (1600,1600),</td>
<td>B</td>
<td>=6.2M</td>
<td>40.37±0.02</td>
<td>+0.36</td>
</tr>
<tr>
<td>Single biLM: 400bi(400,400),</td>
<td>B</td>
<td>=400</td>
<td>40.35±0.04</td>
<td>+0.34</td>
</tr>
<tr>
<td>Single biLM: (400,400),</td>
<td>B</td>
<td>=4.2M</td>
<td>40.21±0.02</td>
<td>+0.19</td>
</tr>
<tr>
<td>Single biLM: word-based,</td>
<td>B</td>
<td>=8.6M</td>
<td>40.25±0.03</td>
<td>+0.23</td>
</tr>
<tr>
<td>Single coarse LM: 100tgt</td>
<td>40.36±0.04</td>
<td>+0.35</td>
<td>42.14±0.02</td>
<td>+0.12</td>
</tr>
<tr>
<td>Single coarse LM: 800tgt</td>
<td>40.33±0.02</td>
<td>+0.32</td>
<td>42.19±0.04</td>
<td>+0.16</td>
</tr>
<tr>
<td>Single coarse LM: 1600tgt</td>
<td>40.30±0.01</td>
<td>+0.28</td>
<td>42.09±0.02</td>
<td>+0.07</td>
</tr>
<tr>
<td>Baseline</td>
<td>40.01±0.02</td>
<td>(0.0)</td>
<td>42.02±0.03</td>
<td>(0.0)</td>
</tr>
</tbody>
</table>

3. Discussion and Future Work

BiLMs provide phrase-based SMT systems with richer source-side context during decoding. In experiments with highly competitive baselines, pure word-based 8-gram biLMs yield only modest gains in the range +0.1–0.2 BLEU for four language pairs. This is probably due to training data sparsity caused by the large number of bitoken types. Indeed, when we replace word-based 8-gram biLMs with coarse 8-gram biLMs, we get much greater gains from the latter. Our results also show that coarse biLMs and coarse LMs of different granularities contain partially complementary information: for each of the language pairs, loglinear combinations of coarse models score higher than single coarse models on blind test data.

We defined a “generic” coarse configuration by looking at Eng>Fre devtest results: 400bi(400,400)&(400,400)&100tgt&1600tgt (a loglinear combination of two types of coarse biLM and two coarse LMs of very different granularities). For this configuration, BLEU gains over language-specific baselines on blind data were +0.80 for Eng>Fre, +0.35 for Fre>Eng, +1.0 for Ara>Eng, and +0.6 for Chi>Eng. If we apply the configuration with the highest devtest score on a given language pair to blind data, the gains are +0.85 for Eng>Fre, +0.46 for Fre>Eng, +1.2 for Ara>Eng, and still +0.6 for Chi>Eng. The consensus in the literature is that coarse models help most when the target language has complex morphology, so we expected the largest gains to be for Eng>Fre: we were surprised by the large gains for Ara>Eng. It looks as though source context information is especially valuable for Ara>Eng.

The Eng>Fre baseline system’s LM took only 0.1G of storage, but adding the “generic” coarse LM-biLM configuration brought this to 2.4G. Adding “generic” took Fre>Eng LM size from 0.1G to 2.0G. Because they have four LMs, baselines for the other two language
pairs have much higher total LM sizes than the Eng<>Fre baselines: adding “generic” took total LM size from 15.0G to 18.4G for Ara>Eng, and from 7.6G to 11.0G for Chi>Eng. We didn’t measure time or virtual memory (VM) during decoding impacts precisely: very roughly, adding “generic” increased decoding time about 30% for all four systems, and increased VM size about 60% for the Eng<>Fre systems and about 20% for the other two.

<table>
<thead>
<tr>
<th>System</th>
<th>avg(Devtest1, Devtest2)</th>
<th>Gain on devtest</th>
<th>Test</th>
<th>Gain on test</th>
</tr>
</thead>
<tbody>
<tr>
<td>400bi&amp;(1600,1600)&amp;100tgt&amp;1600tgt</td>
<td>40.56±0.09</td>
<td>+0.76</td>
<td>46.03±0.07</td>
<td>+1.20</td>
</tr>
<tr>
<td>400bi(800,800)&amp;100tgt&amp;1600tgt</td>
<td>40.51±0.12</td>
<td>+0.72</td>
<td>45.74±0.13</td>
<td>+0.91</td>
</tr>
<tr>
<td>400bi(400,400)&amp;(400,400)&amp;100tgt&amp;1600tgt = “Generic”</td>
<td>40.43 ±0.08</td>
<td>+0.63</td>
<td>45.80±0.14</td>
<td>+0.97</td>
</tr>
<tr>
<td>Single biLM: 400bi,</td>
<td>40.38±0.13</td>
<td>+0.59</td>
<td>45.43±0.07</td>
<td>+0.60</td>
</tr>
<tr>
<td>Single biLM: 400bi(800,800),</td>
<td>40.23±0.04</td>
<td>+0.44</td>
<td>45.35±0.08</td>
<td>+0.52</td>
</tr>
<tr>
<td>Single biLM: 400bi(400,400),</td>
<td>40.18±0.09</td>
<td>+0.38</td>
<td>44.94±0.18</td>
<td>+0.11</td>
</tr>
<tr>
<td>Single biLM: (400,400),</td>
<td>40.15±0.08</td>
<td>+0.36</td>
<td>45.02±0.07</td>
<td>+0.19</td>
</tr>
<tr>
<td>Single biLM: (1600,1600),</td>
<td>40.03±0.04</td>
<td>+0.23</td>
<td>45.10±0.12</td>
<td>+0.27</td>
</tr>
<tr>
<td>Single biLM: word-based,</td>
<td>40.06±0.06</td>
<td>+0.26</td>
<td>44.96±0.09</td>
<td>+0.13</td>
</tr>
<tr>
<td>Single coarse LM: 1600tgt</td>
<td>40.14±0.06</td>
<td>+0.34</td>
<td>45.12±0.09</td>
<td>+0.29</td>
</tr>
<tr>
<td>Single coarse LM: 100tgt</td>
<td>40.03±0.04</td>
<td>+0.23</td>
<td>45.10±0.05</td>
<td>+0.27</td>
</tr>
<tr>
<td>Baseline</td>
<td>39.80±0.05</td>
<td>(0.0)</td>
<td>44.83±0.08</td>
<td>(0.0)</td>
</tr>
</tbody>
</table>

Table 7. Ara>Eng BLEU for coarse model combinations (single components in italics)

There are several directions for future work:

- The method for hard-clustering words/bitokens could be improved – e.g., as in Blunsom and Cohn (2011). As a reviewer helpfully pointed out, coarse models of the same type but different granularities could be trained more efficiently with true IBM clustering (Brown et al., 1992) to create a hierarchy for words or bitokens that would yield many different granularities after a single run, rather than by running mkcls several times (once per granularity).

- Coarse models could be used for domain adaptation - e.g., via mixture models that combine in-domain and out-of-domain or general-domain data (Koehn and Schroeder, 2007; Foster and Kuhn, 2007; Sennrich, 2012). In-domain statistics will be better-estimated in a coarse mixture than in a word-based one.

- “Mirror-image” word-based or coarse target-to-source biLMs could be used to rescore N-best lists or lattices. If there has been word reordering, these would apply context information not seen during decoding. E.g., let source “A B C D E F G H” generate hypothesis “a b f h g c d e”, and “A” be aligned with “a”, “B” with “b”, etc. With trigram word-based biLMs, the trigrams involving “f” seen during decoding are “a_A b_B f_F”, “b_B f_F h_H”, and “f_F h_H g_G”. During mirror-image rescoring, the biLM trigrams involving
“f” that are consulted are “D_d E_e F_f”,” “E_e F_f G_g”, and “F_f G_g H_h” – a different set of trigrams, potentially containing additional information.

- In this work, the most time-consuming task was finding the best combination of coarse models for a given language pair/corpus. We hope to devise a computationally cheap way of finding the best combination on devtest data. An approach based on minimizing the perplexity of held-out data might work, if the correlation between this and SMT quality turns out to be sufficiently high.

An interesting direction for future work is comparison between coarse models and neural net (NN) approaches. In principle, everything learned by coarse LMs or coarse biLMs could be learned by a neural net (NN) trained on the same data. Will NNs make coarse models obsolete? Only thorough experimentation will show which of NNs or coarse model combinations really yield better translations – perhaps they complement each other. Currently, an advantage of coarse models over NNs is quicker training times; one could further shrink training time for coarse models by incrementally adapting word clusterings trained on generic data to new domains. However, incremental adaptation is also a possible strategy for NNs. Analysis of these tradeoffs between coarse models and NNs – in terms of model quality, speed of training, ease of incremental adaptation, etc. – is our top priority for future work.

<table>
<thead>
<tr>
<th>System</th>
<th>avg(Devtest1, Devtest2)</th>
<th>Gain on devtest</th>
<th>Test</th>
<th>Gain on test</th>
</tr>
</thead>
<tbody>
<tr>
<td>400bi(1600,1600tgt)&amp;(800,800)&amp;100tgt&amp;400tgt</td>
<td>30.38±0.08</td>
<td>+0.82</td>
<td>32.46±0.04</td>
<td>+0.61</td>
</tr>
<tr>
<td>400bi(1600,1600)&amp;(800,800)&amp;100tgt&amp;1600tgt</td>
<td>30.36±0.08</td>
<td>+0.80</td>
<td>32.44±0.05</td>
<td>+0.59</td>
</tr>
<tr>
<td>400bi(400,400)&amp;&amp;(400,400)&amp;100tgt&amp;1600tgt = “Generic”</td>
<td>30.16±0.11</td>
<td>+0.60</td>
<td>32.41±0.05</td>
<td>+0.56</td>
</tr>
<tr>
<td>Single biLM: 400bi(1600,1600),</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>=400</td>
<td>30.02±0.08</td>
<td>+0.47</td>
</tr>
<tr>
<td>Single biLM: (800,800),</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>=3.4M</td>
<td>29.90±0.10</td>
<td>+0.34</td>
</tr>
<tr>
<td>Single biLM: (400,400),</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>=2.8M</td>
<td>29.76±0.08</td>
<td>+0.21</td>
</tr>
<tr>
<td>Single biLM: 400bi(400,400),</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>=400</td>
<td>29.94±0.07</td>
<td>+0.38</td>
</tr>
<tr>
<td>Single biLM:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>word-based,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>=6.9M</td>
<td>29.74±0.09</td>
<td>+0.19</td>
</tr>
<tr>
<td>Single coarse LM: 100tgt</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>29.82±0.02</td>
<td>+0.26</td>
<td>32.14±0.06</td>
<td>+0.29</td>
</tr>
<tr>
<td>Single coarse LM: 400tgt</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>29.83±0.09</td>
<td>+0.28</td>
<td>32.17±0.08</td>
<td>+0.31</td>
</tr>
<tr>
<td>Single coarse LM: 1600tgt</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>29.82±0.06</td>
<td>+0.26</td>
<td>32.11±0.05</td>
<td>+0.26</td>
</tr>
<tr>
<td>Baseline</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>29.56±0.05</td>
<td>(0.0)</td>
<td>31.85±0.08</td>
<td>(0.0)</td>
</tr>
</tbody>
</table>

Table 8. Chi>Eng BLEU for coarse model combinations (single components in italic)

Acknowledgement

This research was supported in part by DARPA contract HR0011-12-C-0014 under subcontract to Raytheon BBN Technologies.

References


Using any machine translation source for fuzzy-match repair in a computer-aided translation setting

John E. Ortega
ejoe10@alu.ua.es
Felipe Sánchez-Martínez
fsanchez@dlsi.ua.es
Mikel L. Forcada
mlf@dlsi.ua.es
Dept. de LLenguatges i Sistemes Informàtics, Universitat d’Alacant, E-03071, Alacant, Spain

Abstract
When a computer-assisted translation (CAT) tool does not find an exact match for the source segment to translate in its translation memory (TM), translators must use fuzzy matches that come from translation units in the translation memory that do not completely match the source segment. We explore the use of a fuzzy-match repair technique called patching to repair translation proposals from a TM in a CAT environment using any available machine translation system, or any external bilingual source, regardless of its internals. Patching attempts to aid CAT tool users by repairing fuzzy matches and proposing improved translations. Our results show that patching improves the quality of translation proposals and reduces the amount of edit operations to perform, especially when a specific set of restrictions is applied.

1 Introduction
Computer-aided translation (CAT) tools based on translation memories (TM) are one of the most popular technologies among professional translators (Bowker, 2002; Somers, 2003). CAT tools exploit existing, segment-aligned translations to help the translator translate a new document by recycling as much target-language (TL) text as possible. To do so, CAT tools first split the source-language (SL) document to translate into segments, and for each SL segment $s'$ they look up the translation memory for segment pairs $(s, t)$ (called translation units) where $t$ is the translation of $s$ and $s$ is similar to $s'$. These translation units are then shown to the translator in decreasing order of similarity, together with an indication of the words in $s$ that do not match those in $s'$. Finally, the translator decides which translation unit to use and which parts of its TL segment $t$ have to be edited to produce $t'$, the desired translation of $s'$, and performs such editions.

The similarity between $s$ and $s'$ is computed by means of a fuzzy-match score (FMS) function whose output is between 0% (no match at all) and 100% (a perfect match, $s = s'$). Commercial CAT systems implement proprietary versions of FMS, but a reasonable approximation is given by:

$$FMS(s, s') = \left(1 - \frac{ED(s, s')}{\max(|s|, |s'|)}\right) \cdot 100\% \tag{1}$$

where $ED(s, s')$ is the (word-based) edit distance (Wagner and Fischer, 1974) between $s$ and $s'$ —the minimum number of one-word deletions, insertions and substitutions needed to transform $s$ into $s'$— and $|x|$ stands for the number of words in segment $x$. Many times translation tools
use a *fuzzy-match score threshold* (FMT), for instance 80%, to reduce the number of translation proposals.

When a perfect match is not found in the translation memory, and before making any changes to the TL segment \( t \) in the proposed translation unit, the translator has to identify the sub-segments of \( t \) that correspond to the sub-segments of \( s \) that are not common to \( s' \). To help in finding the sub-segments of \( t \) that need to be edited, but without actually editing them, Esplà-Gomis et al. (2011) use machine translation (MT) to find sub-segment alignments between \( s \) and \( t \), and train a classifier to classify the words in \( t \) as words to be kept unedited or words to be changed to transform \( t \) into the desired translation \( t' \).

Other researchers have gone one step further and have explored different ways to combine the TL segment of the proposed translation unit and the output of a *statistical machine translation* (SMT) system to produce a translation closer to \( t' \). Biçici and Dymetman (2008), for example, use a phrase-based SMT system trained on a bilingual corpus in the same domain as the TM and combine it with the TM’s fuzzy match by extracting a phrase table that is dynamically added to the usual set of bi-phrase sets used for decoding the source. Their implementation augments the internal translation table in the SMT system with bilingual discontiguous sub-segments (phrases) that have source sub-segments in common with \( s' \). Alignments in the system created by Biçici and Dymetman (2008) are detected using word alignments directly obtained from the SMT system training process and are used to find the parts of \( t \) that need to be edited (mismatches).

Similarly, Simard and Isabelle (2009) use a phrase-based SMT system by adding phrase pairs (sub-segment pairs) of any length (obtained using a statistical aligner on the TM) to the SMT system’s phrase table and introduce a feature to indicate that the phrase-pairs came from their TM. After that, they optimize the weighting of the TM-based phrase table in a regular SMT decoder. By means of optimization and phrase table inclusion they are able to make their SMT system produce a translation close to the desired translation \( t' \).

Additional work done by Zhechev and Genabith (2010) makes use of a phrase-based SMT system along with an alignment method that, like Simard and Isabelle (2009), connects sub-segments from the target translation \( t \) with those in \( s \). The alignment method Zhechev and Genabith (2010) use takes advantage of a tree-based structural alignment created from a bilingual dictionary after training their SMT system with phrase pairs. After aligning the words in \( s \) with those in \( t \), Zhechev and Genabith (2010) are able to identify words that should appear in the final translation \( t' \).

Koehn and Senellart (2010) take a similar approach to Biçici and Dymetman (2008). They first align words in \( s' \) and \( s \) to find mismatches. Then, they align the words in \( s \) and \( t \) to identify target matches and remove the words in \( t \) that are aligned to the mismatched words in \( s \). Target mismatches are sent to the SMT decoder for translation. Mismatched words in Koehn and Senellart (2010)’s system are treated separately; that is, context around a mismatch, while indirectly taken into account by the language model, is not directly taken into account of when applying phrase pairs.

Ma et al. (2011), on the other hand, decided to research the shortcomings of using a fuzzy-match score as a threshold for determining translation unit matches that serve as translations for other segments. Ma et al. (2011)’s approach uses discriminative learning and support vector machines to salvage translations of matched words to select a translation unit that would have been otherwise thrown away due to the fuzzy-match score being used as a threshold. Their work, unlike Koehn and Senellart (2010), takes matched parts in \( s \) and replaces them with their counterparts in \( t \). The main drawback of the approaches from Ma et al. (2011), Koehn and Senellart (2010), Zhechev and Genabith (2010), and Biçici and Dymetman (2008) is that they are all based on SMT and either have access to the internals of an SMT system trained on the
user’s or related data or modify its behavior in some way.

Other research work (Hewavitharana et al., 2005; Dandapat et al., 2011) focuses on the identification of the sub-segments in the TL segment \( t \) of the translation unit \((s,t)\) needed to produce \( t' \) and then produces a translation by applying a set of edit operations over \( t \). In particular, Hewavitharana et al. (2005) first align the mismatches in \( s \) to their TL translations in \( t \) by means of a modified IBM model 1 and then apply the same edit operations—substitutions, deletions and insertions—that are needed to convert \( s \) into \( s' \) to the TL segment \( t \). Their resulting translation may contain agreement and reordering errors because their method assumes that edit operations applied on the SL are exactly the ones needed in the TL and do not take word context into account.

Dandapat et al. (2011) were also able to successfully translate texts in the TL by marking mismatched words for translation. Dandapat et al. (2011)’s example-based machine translation (EBMT) and SMT work marks sub-segments for translation from \( s' \) and \( s \) in a manner similar to that of this paper. Their work involves creating sub-segment pairs to form a sub-segment TM, marking mismatched words, aligning matched words, and finally substituting words marked for translation in what they call a recombination step. Their recombination step substitutes words using a sub-segment TM, that is, a mismatched phrase table obtained from the user’s TM in an SMT training job. Sub-segments are translated and “plugged” (i.e., inserted or replaced) into \( t \) according to how they are found in the source text without taking into account other context around the mismatched sub-segments. Plugging and similar approaches, like the one from Hewavitharana et al. (2005), have some shortcomings due to the lack of contextual information around a mismatched sub-segment and differ from our approach in this respect.

We investigate fuzzy-match repair using a technique called patching that uses any external bilingual source to translate mismatched sub-segments; patching could use a glossary, a terminological database (Bowker, 2003), or another translation memory containing smaller segment pairs. Our approach, while related to the research described above, exhibits three main novelties: (i) it removes the dependency on knowledge of the internal workings of the MT system used, (ii) it removes the need to modify an MT system’s behavior in some way and (iii) avoids having to pre-process a user’s TM. We repair the mismatched sub-segments in a translation unit using a simple, yet novel, method that, unlike those by Hewavitharana et al. (2005) and Dandapat et al. (2011), takes context around mismatched words into account. Patching uses overlapping sub-segments as powerful anchors much like Brown et al. (2003)’s maximal left-overlap compositional (example-based) MT system where the use of overlapping sub-segments reduces ”boundary friction” problems and increases the likelihood of producing a correct translation. Since patching treats sources of external bilingual translation information as black boxes, we generate translations on the fly without training SMT models on the user’s TM. We are not aware of any research work that: (a) uses any source of bilingual information for translation or (b) uses the context around mismatched words for repair.

In the following sections, we show how the fuzzy-match repair method mentioned can be applied in a CAT setting. The rest of the paper is organized as follows. The next section describes in detail our approach and illustrates how it works with an example. Section 3 describes the experiments we have conducted and the results achieved. The paper closes with some concluding remarks and potential future research.

2 Methodology

We begin with a foundation similar to Esplà-Gomis et al. (2011): an engine-agnostic approach which, unlike other work done so far, only requires that the CAT tool is able to invoke the external translation system, in order to translate short source-side sub-segments \( \sigma \) of \( s \) to obtain the corresponding short target-side sub-segments \( \tau \) of \( t \). As Esplà-Gomis et al. (2011), the
method described here can use online MT or any MT system “out of the box” — indeed, as described earlier, any source of such sub-segmental translation units \((\sigma, \tau)\) may in principle be queried.

Our methodology, unlike Esplà-Gomis et al. (2011), does not only mark words from \(t\) for editing, it goes one step further and edits them using a patching method that can be described in 5 steps:

1. Align the words in the SL segment \(s'\) to be translated to those in the SL segment \(s\) of the translation unit and find mismatched words.

2. Translate the sub-segments \(\sigma\) of \(s\) and \(\sigma'\) of \(s'\) containing at least one mismatched word, up to a given sub-segment length, by querying the sources of bilingual information available.

3. Match each translated sub-segment \(\sigma\) of the SL segment \(s\) to those sub-segments \(\tau\) in the TL segment \(t\) of the translation unit (as in Esplà-Gomis et al. (2011)).

4. Pair the translation \(\tau\) of the (mismatched) sub-segments \(\sigma\) in \(s\) for which a match has been found in \(t\) to the translation \(\tau'\) obtained for sub-segments \(\sigma'\) in \(s'\). The pairs \((\tau, \tau')\) are the patching operators which replace mismatched sub-segments in \(t\) with the translation of the corresponding mismatched sub-segments in \(s'\) to generate an improved translation candidate \(t''\).

5. Apply the patching operators to build all possible translation hypotheses by selecting valid sets of patching operators that can be applied to form a final proposal. Restrictions can be applied to limit the amount of patching operators considered valid.

As patching can, in general, yield more than one solution, translation hypotheses could then be ranked according to their estimated quality so that the best one is shown to the translator for validation or post-editing. Here, in the absence of a quality estimation (QE) method that we plan as future work, we experiment with restrictions to discard less reliable patching operators and reduce the number of translation hypotheses to generate. With restrictions in place, we then evaluate the average quality of the resulting repaired sentences on a test set as well as the quality of the best proposal.

In the following sub-sections, we illustrate the patching process (steps 1 through 5 above) in detail by building patching operators for an English segment and then applying them to produce a translation in Spanish.

### 2.1 Step 1: align and find mismatches

We first find mismatched sub-segments from the source side \((s', s)\) segments of the document and translation memory using fuzzy matching. Imagine we are translating from English to Spanish and the new segment to be translated is:

\[ s' = "The blue dog barks loud when it rains at night" \]

The system shows a fuzzy-match \((s, t)\) from the translation memory and marks mismatched words (in bold below):

\[ s' = "The blue dog barks loud when it rains at night" \]

\[ s = "The red dog barks loud sometimes when it rains at night" \]

\[ t = "El perro rojo ladra fuerte a veces cuando llueve por la noche" \]

According to Eq. (1) the fuzzy-match score is \(\text{FMS}(s', s) = 81.8\%\), and as a side result of the computation of the edit distance, the alignment between the words in \(s'\) and those in \(s\) is produced:
It is clear that there are two words in \( s \) that do not match \( s' \): red and sometimes. Using the edit distance (Wagner and Fischer, 1974), the edit operations to convert \( s \) into \( s' \) would consist of:

- one substitution - replace the word red in \( s \) for the word blue from \( s' \) and
- one deletion - delete the word sometimes from \( s \) between the words loud and when.

### 2.2 Step 2: translate source-side sub-segments covering mismatches

Each one of the sub-segments \( \sigma \) from \( s \) covering a mismatch is sent to an external machine translation system or bilingual source of information to find their corresponding TL translations \( \tau \). The hope is to find a translation \( \tau \) that exists in \( t \) so that the corresponding sub-segment can be later modified to produce the desired translation \( t' \). When \( \tau \) is a substring of \( t \), it is marked as applicable for patching.

If we choose Apertium (Forcada et al., 2011) as the external source of bilingual information to translate the mismatched \((\sigma, \tau)\) pairs and the maximum length of the sub-segments to translate is set to 3, we generate the following translated pairs:

- (the red dog, el perro rojo)
- (the red, el rojo)
- (red dog, perro rojo)
- (red, rojo)
- (loud sometimes when, fuerte a veces cuando)
- (loud sometimes, fuerte a veces)
- (sometimes when, a veces cuando)
- (sometimes, a veces)

### 2.3 Step 3: match the source-side translations to \( t \)

In step 3, we identify the \((\sigma, \tau)\) pairs that have a matching sub-segment in \( t \) and can, therefore, be used to build the patching operators. We keep the following \((\sigma, \tau)\) pairs whose \( \tau \) appears in \( t \):

<table>
<thead>
<tr>
<th>( \sigma )</th>
<th>position in ( s )</th>
<th>( \tau )</th>
<th>position in ( t )</th>
<th>outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>the red dog</td>
<td>1–3</td>
<td>el perro rojo</td>
<td>1–3</td>
<td>kept</td>
</tr>
<tr>
<td>the red</td>
<td>1–2</td>
<td>el rojo</td>
<td>none</td>
<td>discarded</td>
</tr>
<tr>
<td>red dog</td>
<td>2–3</td>
<td>perro rojo</td>
<td>2–3</td>
<td>kept</td>
</tr>
<tr>
<td>red</td>
<td>2–2</td>
<td>rojo</td>
<td>3–3</td>
<td>kept</td>
</tr>
<tr>
<td>loud sometimes when</td>
<td>5–7</td>
<td>fuerte a veces cuando</td>
<td>5–8</td>
<td>kept</td>
</tr>
<tr>
<td>loud sometimes</td>
<td>5–6</td>
<td>fuerte a veces</td>
<td>5–7</td>
<td>kept</td>
</tr>
<tr>
<td>sometimes when</td>
<td>6–7</td>
<td>a veces cuando</td>
<td>6–8</td>
<td>kept</td>
</tr>
<tr>
<td>sometimes</td>
<td>6–6</td>
<td>a veces</td>
<td>6–7</td>
<td>kept</td>
</tr>
</tbody>
</table>

Sub-segments that do not have a match in \( t \) (e.g. the second one in the example above) are discarded and not used further in the patching process. Word positions in \( t \) not covered by
any translation $\tau$ of any segment $\sigma$ in $s$ contain words for which there is no evidence to modify them. In the absence of information, they will not be changed.

2.4 Step 4: pair translations of $\tau$ and $\tau'$ to form patching operators

After the initial matching occurs from the translations of $s$ to form $(\sigma, \tau)$ pairs, $(\sigma', \tau')$ pairs are created by translating mismatched sub-segments from $s'$. The alignment found between words in $s$ and words in $s'$ during fuzzy matching are used by an algorithm, analogous to that by Och and Ney (2000), to extract phrase pairs that project mismatched $\sigma$ sub-segments in $s$ into the corresponding sub-segments $\sigma'$ in $s'$. The $\sigma'$ sub-segments are sent to the MT system or other source of bilingual information to obtain their translations $\tau'$. The final result is a set of patching operators that contain translations that match the previously mismatched words in $s'$ and $s$.

In steps 1 through 3, we have already created the $(\sigma, \tau)$ pairs; now we translate $s'$ sub-segments to form $(\sigma', \tau')$ pairs. In our example, the $(\sigma', \tau')$ pairs translated by Apertium (Forcada et al., 2011) are:

<table>
<thead>
<tr>
<th>$\sigma'$</th>
<th>$\sigma$</th>
<th>positions</th>
<th>$\tau$</th>
<th>$\tau'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>the blue dog</td>
<td>the red dog</td>
<td>$\sigma'=1\text{--}3$, $\sigma=1\text{--}3$</td>
<td>el perro rojo</td>
<td>el perro azul</td>
</tr>
<tr>
<td>blue dog</td>
<td>red dog</td>
<td>$\sigma'=2\text{--}3$, $\sigma=2\text{--}3$</td>
<td>perro rojo</td>
<td>perro azul</td>
</tr>
<tr>
<td>blue</td>
<td>red</td>
<td>$\sigma'=2\text{--}2$, $\sigma=2\text{--}2$</td>
<td>rojo</td>
<td>azul</td>
</tr>
<tr>
<td>loud when</td>
<td>loud sometimes</td>
<td>$\sigma'=5\text{--}6$, $\sigma=5\text{--}7$</td>
<td>fuerte a veces cuando</td>
<td>fuerte cuando</td>
</tr>
</tbody>
</table>

In the example above, most of the $\tau'$ are aligned word by word to their corresponding $\tau$ in part because their source sub-segments ($\sigma'$ and $\sigma$) are also aligned word by word. Notice however that the last $\tau'$ ('fuerte cuando') does not align word by word to its corresponding $\tau$ ('fuerte a veces cuando') because it is a deletion case where the words 'a veces' should be deleted.

A patching operator consists of a $(\tau, \tau')$ pair and its positions in $t$. To obtain safer patching operators, we keep only those patching operators where there is overlap between $\tau$ and $\tau'$. On top of that, deletions are required to have at least two words (one on each side of the mismatched sub-segment) of overlapping. The fraction of words in $\tau'$ overlapping $\tau$ (and therefore $t$) may be a good indicator of the quality of the patching operator.

The resulting operators from our example with overlap underlined are:
2.5 Step 5: applying the patching operators

Once the applicable patching operators have been determined, the final step is to apply them to sub-segments in $t$ to create the final translation that is presented to the translator. It is entirely possible to have multiple combinations of patching operators that form multiple repaired segments $t'$. Here are some possible results of applying patching operators to $t$ from our patching example comparing them to the reference translation $t' – el perro azul ladra fuerte cuando llueve por la noche:

- $t_1' = el perro azul ladra fuerte cuando llueve por la noche - (correct, produced by #1 and #4 above)
- $t_2' = el perro azul ladra fuerte a veces cuando llueve por la noche - (incorrect, produced by #2 above)
- $t_3' = el perro rojo ladra fuerte a veces cuando llueve por la noche - (incorrect, produced by #4 above)
- $t_4' = el perro azul ladra fuerte cuando llueve por la noche - (correct, produced by #2 and #4 above)

Patching operators can deal with all three types of edit operations (substitution, insertion, and deletion). The patching examples shown above depict a substitution and a deletion example. Nonetheless, insertions can be handled using patching also. Insertions occur when $\tau'$ contains new words not in $\tau$ (most likely because its corresponding $\sigma'$ contained words that were not in the corresponding $\sigma$).

Note that deletions are always produced with some overlap, that is, in context. Context words around a word to be deleted along with their positions are necessary to determine if an operation is valid. The example above, for instance, creates a patching operator (#4) that deletes the word *sometimes*. We are able to use the context of the surrounding words *loud* and *when* to determine deletion. Context used for patching in this manner is different from other research: Hewavitharana et al. (2005), for example, apply all possible context-free deletions according to statistical word alignment, and later score the resulting segments to determine the best translation.

It is worthwhile to note that patching operators, as seen above, may not always be applicable because positions in $t$ that are modified by a patching operator may not be available for another patching operator when $\tau$ does not match the partially-patched sentence. That would
make the two patching operators incompatible. Our method generates all possible $t \cong s$ obtained by applying all possible sets of mutually compatible patching operators. In our experiments, $t \cong s$ are determined to be correct (or not) by comparing the $t \cong s$ translation to a previously-translated "gold" segment in a test set. Below, in the Experiments section, all of the patching operators applied are compared against their "gold" $t'$ counterparts. During real-world deployment of fuzzy-match repair, and in the absence of a reference translation, patching operators should be assessed before applying them in order to present only the best quality translations to CAT tool users.

### 2.6 Selecting the best $t \cong s$

The patching process can produce a large amount of patched segments $t \cong s$. Sometimes, various patches exist for the same $(\sigma, \sigma')$ mismatch. In our patching example, three different $(\tau, \tau')$ patching operators are produced that cover the same (blue, red) mismatch:

1. (el perro rojo, el perro azul)
2. (perro rojo, perro azul)
3. (rojo, azul)

In order to present the optimum translations to a CAT tool user, it would be beneficial to apply various restrictions to discard redundant or low quality patching operators like those above. The restrictions that we have tested in our experiments are:

- **Restriction 1 ($R_1$)** = Establish a minimum source-side $\sigma$ length in order to disallow patches that are too short.
- **Restriction 2 ($R_2$)** = Disallow patches that do not have context on both sides of the mismatched word to be translated.
- **Restriction 3 ($R_3$)** = Set $|\tau'| = |\tau| = 3$ to handle one-word substitutions with a minimum amount of context.

By applying the restrictions above to the corpus used in experimentation, we show that by using the patching method we are able to increase the translation accuracy of proposals from a translation memory via fuzzy-match repair. The amount of patching operators is limited by our restrictions; experiments display the effects of restriction on performance.

### 3 Experiments

#### 3.1 Experimental Settings

##### 3.1.1 Corpus

Our experiments were performed using an English–Spanish parallel corpus; note, however, that the fuzzy-match repair method would work with any language pair. We have used a data set extracted from the DGT translation memories, which was also used in a project on editing TL hints in a CAT system. The corpus provides an ideal number of segmented sentences both for input and testing as well as a translation memory.

The two main components that we used for experimentation are:

- **Test Set** - The input (in English) and output (in Spanish) containing 1500 source sentences to translate and their corresponding “gold” target sentences.

1[http://transducens.dlsi.ua.es/~mespla/resources/mtacat/02.40.10.40/](http://transducens.dlsi.ua.es/~mespla/resources/mtacat/02.40.10.40/)
3[http://www.dlsi.ua.es/~mespla/edithints.html](http://www.dlsi.ua.es/~mespla/edithints.html)
### Table 1: Segments and target words at different FMS thresholds

<table>
<thead>
<tr>
<th>FMS Threshold (%)</th>
<th># Matched SL Seg’s</th>
<th># Patched TL Seg’s</th>
<th># TL Words Edited</th>
</tr>
</thead>
<tbody>
<tr>
<td>[80, 85)</td>
<td>76</td>
<td>686</td>
<td>750</td>
</tr>
<tr>
<td>[85, 90)</td>
<td>123</td>
<td>239</td>
<td>285</td>
</tr>
<tr>
<td>[90, 95)</td>
<td>82</td>
<td>115</td>
<td>109</td>
</tr>
<tr>
<td>[95, 100)</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td><strong>Total:</strong></td>
<td><strong>282</strong></td>
<td><strong>1041</strong></td>
<td><strong>1145</strong></td>
</tr>
</tbody>
</table>

**TM** - The translation memory with 6000 translation units in English and Spanish.

Since our implementation is a proof of concept, and to avoid onerous computations, we chose to limit the amount of words in the source segments. Segments (s and s′) with more than 25 words or a FMS below 80 percent were discarded because they are harder to deal with in reasonable time due to the combinatorial explosion of patching operator sets.

The amount of mismatched sub-segments varies according to the FMS threshold used. Table 1 shows, for different FMS threshold intervals, the amount of matching (source language) segments, the amount of patched (target) segments and the amount of words in target segments that have been edited.

### 3.1.2 System Setup

Translations for the patching algorithm were performed using an MT system as a black box. In this case, segments were translated directly using the Apertium shallow-transfer MT system (Forcada et al., 2011). The word error rate (WER: see below) on the test set for this rule-based MT system is 56%; this value was computed by translating the source segments in the test set and using their target "gold" translations as references. It is important to note that MT errors are less likely to affect the quality of the patching operators because the τ’s obtained are always required to match a sub-segment in t.

### 3.2 Evaluation

To evaluate the performance of patching and the different types of restrictions (R₁ – R₃) that can be applied to patching operators, we computed the following metrics:

- The average WER between the "gold" translation (t′) and the patch translation for all repaired hypotheses (tₑ).
- The average WER between the "gold" translation and the best repaired hypothesis (this metric provides an indication of the best results we can achieve by patching)
- The average number of repaired hypotheses per segment

WER in our experiments is defined as the complement of the FMS defined in eq. (1):

\[
\text{WER}(t, t_e) = \left( \frac{\text{ED}(t, t_e)}{\max(|t|, |t_e|)} \right) \cdot 100\% \tag{2}
\]

WER tells us how patching performs and by computing the average WER for hypotheses with and without patching along with each individual restriction, we are able to gain better knowledge of restrictions that could be beneficial; that is, those reducing the number of hypotheses while retaining the best hypotheses as often as possible.
### Table 2: Evaluation of Restriction 1 through Restriction 3 with 6000 source segments

<table>
<thead>
<tr>
<th>Restriction(s)</th>
<th>Avg Amount of Hypotheses</th>
<th>Avg WER</th>
<th>Best WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Patching</td>
<td>0</td>
<td>21.37%</td>
<td>21.37%</td>
</tr>
<tr>
<td>No Restrictions</td>
<td>68.96</td>
<td>20.07%</td>
<td>17.42%</td>
</tr>
<tr>
<td>Restriction #1</td>
<td>14.20</td>
<td>20.74%</td>
<td>18.42%</td>
</tr>
<tr>
<td>Restriction #2</td>
<td>21.99</td>
<td>20.07%</td>
<td>17.43%</td>
</tr>
<tr>
<td>Restriction #3</td>
<td>23.14</td>
<td>19.78%</td>
<td>17.44%</td>
</tr>
<tr>
<td>Restrictions #1 - #3 combined</td>
<td>1.67</td>
<td>21.03%</td>
<td>19.62%</td>
</tr>
</tbody>
</table>

3.3 Patching Results

Patching restrictions that help improve system quality are those described in section 2.6: 
- $R_1$, restricting the σ length (restricted here to a minimum of 2 and a maximum of 5); 
- $R_2$, disallowing non-anchored patches, and 
- $R_3$, which sets $|\tau'| = |\tau| = 3$ to handle one-word repairs with minimal context.

Table 2 displays the results achieved for the restrictions and computations described in the Evaluation section. A separate execution is done during evaluation for the following cases: 1) evaluation without applying patching at all, 2) evaluation applying patching without any restrictions, 3) evaluation using each particular restriction, and 4) evaluation using all possible restrictions. A calculation is done that shows the average amount of hypotheses per segment for each setting along with the average WER and the best WER.

The patching technique seems to work better when applied to $\tau$ with less words as seen from the results above for $R_2$. In order to achieve quality patches, we use $(\sigma, \sigma')$ pairs that contain anchored words with overlap on both sides. For example, if the source input file contains a segment with the words “power input of 5 watts” and the translation memory source file contains words in a parallel sentence “power input of about 5 watts”, the system is more likely to correctly replace the “of about 5” by “of 5” due to the fact that there is overlap on both sides of the word “about”.

Table 2 shows a clear distinction between patching with restrictions and without them. Patching improves WER in all cases since the average WER for any restriction $R_i$ is lower than the average WER without patching. But, with respect to restrictions $R_1 - R_3$, only $R_3$ is able to improve the average quality of the hypotheses, thus reducing the average WER. In addition, $R_3$’s average amount of hypotheses is considerably lower (23.14). Restriction $R_2$, while not scoring as well as $R_3$, was able to maintain the same quality (average WER of 20.07%) as patching without restrictions with an average amount of hypotheses (21.99) even lower than $R_3$. Restriction $R_1$ is of less quality when compared to patching without restrictions. $R_1$’s best WER (18.42%) is worse, in absolute terms, than $R_2$ and $R_3$’s best WER.

When combining the restrictions ($R_1 - R_3$), we notice that both the average WER and Best WER perform considerably worse (a quality loss of near 1% or above). On the other hand, $R_2$ and $R_3$, retain the best WER much like their averages above with a very slight degradation of between .01% and .02%. That means that restrictions $R_2$ and $R_3$ should be considered the better restrictions for future use and; due to its lower performance, $R_1$ would probably not be used in the future.

It is worthwhile to note that another useful way to reduce redundant patches would be by deleting patching operator $(\tau_1, \tau_1')$ when there is another operator $(\tau_2, \tau_2')$ such that $\tau_1$ is a
substring of $\tau_2$ and $\tau'_1$ is a substring of $\tau'_2$ or vice-versa. It could be used as a way of reducing patches; but, it does not change the quality of patching as it only reduces patches that cover the same sub-segments. This type of filtering is left for future work and not covered in this paper.

4 Conclusion

We have presented a novel approach to fuzzy-match repair using any MT system or bilingual information source called patching. Patching focuses on a problem in the CAT environment: presenting accurate translation proposals to CAT tool users. In order to help CAT tool users translate faster, patching may be applied to repair and improve fuzzy-match proposals from a translation memory alone. By using any external source for gathering sub-segment translations, translation systems would be able to take advantage of patching by simply adding the patching library to an existing CAT tool and setting the external source of bilingual information.

In this work, we have applied a preliminary set of restrictions which have been shown to effectively reduce the number of repaired segments while keeping the best ones. We have shown that by applying patching to fuzzy matches from a translation memory, we can achieve better WERs as opposed to raw fuzzy matches. Future research will explore further restrictions and use the results to inspire the design of features relevant to learn quality estimators capable of ranking repaired fuzzy-matches so that only the most useful ones are shown to the professional translator using the CAT environment.

Acknowledgements: The authors thank the Spanish Ministry of Economy and Competitiveness for support through grant TIN2012-32615 and Rafael C. Carrasco for useful comments.

References


Enhancing Statistical Machine Translation with Bilingual Terminology in a CAT Environment

Mihael Arcan
Insight Centre for Data Analytics, National University of Ireland, Galway, Ireland

Marco Turchi
turchi@fbk.eu
FBK - Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy

Sara Tonelli
satonelli@fbk.eu
FBK - Fondazione Bruno Kessler, Via Sommarive 18, 38123 Trento, Italy

Paul Buitelaar
paul.buitelaar@insight-centre.com
Insight Centre for Data Analytics, National University of Ireland, Galway, Ireland

Abstract
In this paper, we address the problem of extracting and integrating bilingual terminology into a Statistical Machine Translation (SMT) system for a Computer Aided Translation (CAT) tool scenario. We develop a framework that, taking as input a small amount of parallel in-domain data, gathers domain-specific bilingual terms and injects them in an SMT system to enhance the translation productivity. Therefore, we investigate several strategies to extract and align bilingual terminology, and to embed it into the SMT. We compare two embedding methods that can be easily used at run-time without altering the normal activity of an SMT system: XML markup and the cache-based model. We tested our framework on two different domains showing improvements up to 15% BLEU score points.

1 Introduction
Recent studies (Federico et al., 2012; Läubli et al., 2013; Green et al., 2013) have shown significant productivity gains when human translators post-edit machine translation output rather than translating documents from scratch. This evidence has raised interest in the integration of machine translation systems within CAT software. In this context, an important open issue is how to support translators with domain-specific information when dealing with highly specific texts, i.e. manuals coming from different domains (information technology (IT), legal, agriculture, etc.). Translation tools such as Google Translate, Bing Translator or open source SMT systems such as Moses (Koehn et al., 2007) trained on generic data are the most common solutions, but they often result in unsatisfactory translations. A valuable alternative to support professional translators is represented by online terminology resources, e.g. IATE, which are continuously updated and can be easily queried. However, the manual use of these services can be very time demanding when working with a CAT tool. For these reasons, the automatic identification and integration of bilingual domain-specific terms into an SMT system is a crucial step towards increasing translation quality of high-specific texts in a CAT environment. This also reduces translators’ initial overload when dealing with different domains, because terminological lists are managed directly by the SMT system and no additional human intervention for retrieving domain-specific terminology is required.

In this paper, we propose a framework for extracting bilingual terms from parallel data and using them to enhance the performance of an SMT system embedded in a CAT tool. We focus on a real scenario, where a large translation project is split across different translators and each translator post-edits daily a limited amount of sentences provided by the SMT system. Our approach takes advantage of such post-edited data to gather bilingual domain-specific terms. The parallel data produced every day are then used to continuously improve a generic machine translation system by (i) automatically injecting the bilingual terms into the SMT system, and (ii) optimising the log-linear weights on this specific data.

Bilingual term extraction is performed in two steps. First, the source and the target sides of the data are processed by a keyword extractor to identify the most relevant terms in each language. Taking advantage of the parallel data, each monolingual term in the source language is paired with a term in the target language. We perform this step by comparing different techniques, showing that simple approaches based on word alignment and term translation are more robust and more efficient than the state-of-the-art method based on supervised classification (Aker et al., 2013).

As regards the integration of the bilingual terms in an SMT system, we cannot apply well-known approaches (Bouamor et al., 2012) adding the terms to training data or at the end of the phrase table, because in our CAT scenario we cannot stop the translation service and let translators wait for a long training time. For this reason, we investigate for the first time the integration of cache-based translation and language models (Bertoldi et al., 2013) in the context of terminology embedding comparing them with the XML markup technique. The cache-based model makes it possible to periodically add bilingual terms into an SMT system in real-time, without the need to stop it. In addition, we compare the cache-based models with a recently developed technique, namely the Realtime Adaptive Translation Systems with cdec (Denkowski et al., 2014), that, based on lexicalized synchronous context-free grammars, takes as input the whole source and post-edited sentences and automatically updates the models. The evaluation of our framework on two different domains (IT and medical) suggests that: (i) an SMT model enriched with the identified bilingual terms substantially improves translation quality in terms of BLEU score over a generic SMT system; (ii) strategies to integrate terminology need to take into consideration also the surrounding context of a translated term; (iii) in order to take advantage of the continuous appending of new information inside the SMT system a constant updating of the contribution of each component in the log-linear model is needed.

2 Bilingual Domain-Specific Terminology Generation

We propose a two-step approach to extract bilingual terminology for machine translation that requires only small amounts of parallel data (few hundred), as foreseen in a CAT scenario. The first step is the extraction of domain-specific terms from monolingual data (target and source sides of the parallel data), while the second is the creation of bilingual terminology starting from the monolingual ones. In order to obtain the best possible performance, we compare different approaches in both steps. At the monolingual level, we test the extraction using three unsupervised term extraction tools. For bilingual alignment, we compare different alignment strategies. The two steps are detailed in the following subsections.

2.1 Monolingual Terminology Extraction

In order to find the best performing approach to identify monolingual terms, we compare three available term extractors: the KX toolkit (Pianta and Tonelli, 2010), TWSC (Pinnis et al., 2012) and AlchemyAPI. Given our experimental scenario, where no or little training data are available, we chose three unsupervised terminology extractors supporting different languages.

4 http://www.alchemyapi.com/products/features/keyword-extraction/
KX is a terminology extractor, which combines frequency information and part-of-speech patterns of n-grams to identify the most relevant terms in a corpus. It is freely available for English and Italian and was the first-ranked unsupervised system in the Semeval2010 task on keyword extraction (Kim et al., 2010). TWSC follows an approach which is very similar to KX, integrating morpho-syntactic patterns with statistical features. One of the main differences w.r.t. KX is the implementation of different co-occurrence statistics to rank term candidates, and the treatment of nested terms. Nevertheless, we expect the performance of these two tools to be very similar. A third system considered is AlchemyAPI. This commercial tool employs sophisticated statistical algorithms and linguistic approaches to analyse textual content and extract topic keywords, but no further implementation details are given.

2.2 Bilingual Terminology Alignment

Once the lists of monolingual terms for the source and target language are automatically gathered, the alignment across languages is created. We propose and compare different strategies.

Given a source term and the parallel sentence pair in which it appears, a set of possible translations is found by either translating the term or by applying a word aligner. In both cases, we use a technique similar to the methodology proposed by (Ehrmann et al., 2011), where the translation system the word aligner are trained on the same data from which the bilingual terminology is extracted. The main idea is that the translation system should know how to translate a source term, since it has seen it in the training data; this reduces the number of untranslated terms. Moreover, this allows us to take advantage of monolingual term extractors and regular phrase extraction method, used to build the phrase table, to generate bilingual terminology.

Given a set of possible translations for each term, the correct translation is retrieved taking advantage of the parallelism between source and target sentences, whereby two methods are investigated: sentence lookup or term lookup. With the first, a target translation from the candidate list is accepted as correct if it matches a span in the target sentence. With the second, a translation is accepted if it has also been identified as a term in the target sentence by the monolingual term extractor. The term lookup method reduces the number of extracted bilingual terms, but guarantees a better quality of the alignments.

In our experiments, we compare our strategies with Term Aligner, a state-of-the-art bilingual alignment tool, based on the method proposed by Aker et al. (2013). In this method, the authors treat bilingual term alignment as a classification problem. An SVM binary classifier is trained on data derived from the multilingual thesaurus EuroVoc, using language dependent and independent features. The former ones are based on bilingual dictionaries created by the GIZA++ tool, while the latter use cognate-based features, e.g. the longest common subsequence ratio. The cognate features are binarized using a manually defined threshold. Since the original work focuses on term alignment in comparable corpora, we limit the tool to search for terms that appear in the same parallel sentence pair. Moreover, we use the same GIZA++ dictionaries built for identifying term translation.

3 Enhancing Terminology Translation

After the extraction of domain-specific bilingual terms, they need to be integrated into the workflow of the SMT system. We focus on a real scenario, where a large translation project is split into partitions with around 3,000 tokens, which represent the average workload of a professional translator in the post-editing task per day. Translating partition\textsubscript{n}, the decoder is supported by the extracted and aligned bilingual terminology from previous partitions (partition\textsubscript{1}, \ldots, partition\textsubscript{n-1}) using the XML markup or the cache-based models. To further improve the translation quality of partition\textsubscript{n}, the decoder accesses the log-linear weights from the previous partition, which were tuned beforehand with MERT (Bertoldi et al., 2009).
Given the extracted terms and the parallel sentences, we improve the translation capability of the SMT system by: (i) using the bilingual terms during the translation process and (ii) running an incremental tuning on different sets of parallel sentences coming from different working days.

3.1 Integration of Bilingual Terms into SMT

Since we place our work a CAT scenario, where an SMT system should continuously provide suggestions to the translator for each source sentence, we cannot integrate bilingual terms by retraining the whole model (Bouamor et al., 2012) or switching off the system and adding the terms at the end of the phrase table (Bouamor et al., 2011). Also the incremental training method introduced by Levenberg et al. (2010), which makes it possible to continuously add data without retraining the model, is not the best solution in our setting, because it tends to penalise terms with ambiguous translations favouring the most frequent and generic translations. For these reasons, we test two methods that can be easily used at run-time without altering the normal work of the SMT system and differentiate domain-specific from general translations: the widely-used XML markup and the cache-based model (Bertoldi et al., 2013).

XML Markup  With the XML markup approach, external knowledge is directly passed to the decoder by specifying the translation of specific spans of the source sentence. In case of multiple translations of the same source span, a score can be used to indicate the level of association between the source and target phrases.

Cache-Based Models In this work, we propose for the first time the use of the cache-based translation and language models (Bertoldi et al., 2013) for embedding bilingual terms into the SMT system. The main idea behind these models is to combine a large static global model with a small, but dynamic local model. This allows users to define and dynamically adapt domain-specific models that are combined during decoding with the global SMT models built on the training data. Differently from XML markup that only substitutes the annotated source strings with a given translation without considering the surrounding context for proper lexical choice, the cache-based model offers a better integration of the terms into the final translation.

The cache-based model relies on a local translation model (CBTM) and language model (CBLM). The first is implemented as an additional phrase table providing one score. All entries are associated with an ‘age’ (initially set to 1), corresponding to the time when they were actually inserted. Each new insertion causes an ageing of the existing phrase pairs and hence their re-scoring; in case of re-insertion of a phrase pair, the old value is set to the initial value. Phrase pairs in the model are scored based on the decaying function, whereby we test different rewarding and penalizing functions (hyperbola, power, exponential, cosine) as well as a constant function, where the ‘age’ is always set to 1. Similarly to the CBTM, the local language model is built to give preference to target terms found by the extraction tool. Each target term stored in CBLM is associated with a decaying function of the age of insertion into the model. Both models are used as additional features of the log-linear model in the SMT system.

3.2 Incremental Tuning

The continuous extraction and collection of bilingual terms changes the capability of the SMT to correctly translate new sentences and the contribution of each component in the log-linear model. For this reason, when a new partition of parallel sentences is available (partition$_n$), bilingual terms are first extracted. Then, before using them in the cache-based or XML markup module, the tuning step is performed using partition$_{n-1}$ as development set and taking advantage of all terms extracted from partition$_1$ to partition$_{n-2}$. When the new weights are computed, the bilingual terms extracted from partition$_{n-1}$ are added to the terms obtained
from all the previous partitions, and the new configuration of the SMT system is used to translate partition $n$. The aim of this procedure is to update the weights of each feature taking into consideration the new translation capability of the model. The initial configuration of the log-linear weights used by MERT at time $n-1$ is that obtained optimizing the system at time $n-2$. Once the new weights are computed, the old weights need to be overwritten. This is done by passing the new weights to Moses through XML tags for each incoming sentence, which required to extend Moses with this new option.

An issue with incremental tuning is the risk of over-fitting of the model on a small development set, when it differs from the test set. In our scenario, this is prevented by the fact that all the sets come from the same document, or from different documents on similar topic in the same project. Although it is important to tune an SMT system on a sufficiently large development set, reasonably good weights can be obtained even if such data are very few, as shown in Bertoldi and Federico (2009). In our framework, it is not possible to concatenate all the previous partitions to enlarge the development set, because the presence of already extracted bilingual terms in the cache-based models would artificially favour the cache-based components during the tuning.

### 4 Experimental Setting

In this Section, we propose a set of experiments aimed at showing the capability of our framework to extract high quality domain-specific bilingual terms from a small amount of parallel data and to integrate them in the translation task. The translation direction considered is from English to Italian. To identify the best monolingual term extraction tool as well as the most suitable bilingual alignment approach, we use freely available data, which were manually annotated to better evaluate all the intermediate steps of the experiment. Two datasets belonging to the IT domain, namely a portion of GNOME project data (4,3K tokens)\(^5\) and KDE Data (9,5K),\(^6\) are used for domain-specific term extraction.

The whole framework, including the machine translation part, is tested on a subset of the EMEA corpus (Tiedemann, 2009) for the medical domain (18K tokens) and an IT corpus (18K), extracted from a software user manual (Federico et al., 2014). Each corpus is split in partitions of around 3,000 tokens, i.e. the daily workload of a professional translator in post-editing, resulting in 6 partitions each.

For each translation task, we use the statistical translation toolkit Moses (Koehn et al., 2007), where the word alignments were built with the GIZA++ toolkit (Och and Ney, 2003). The IRSTLM toolkit (Federico et al., 2008) was used to build the 5-gram language model.

For a broader domain coverage of the generic SMT system, we merged parts of JRC-Acquis (Steinberger et al., 2006), Europarl (Koehn, 2005) and OpenSubtitles2013 (Tiedemann, 2009), obtaining a training corpus of 37M tokens and a development set of ~25K tokens. The generic SMT system used in all our experiments is trained on this merged general resource. The difference in size between the specific and the generic data is evident, i.e. approximately few thousands vs. more than 30 million tokens. For both domains, this reflects a real CAT scenario, where only a small quantity of domain-specific data is available.

**Manual Terminology Annotation** In order to evaluate the quality of the bilingual terms, we create a terminological gold standard for the IT domain. Two annotators with linguistic background were asked to mark all domain-specific terms in the monolingual GNOME and KDE corpora. Domain-specificity was defined as all (multi-)words that are typically used in the IT domain and that may have different translations in other domains. Then, the annotators had to manually create a bilingual pair if two domain-specific terms in a source and target sentence

---

\(^5\) https://l10n.gnome.org/  \(^6\) http://i18n.kde.org/
were found, one being the translation of the other. The average Cohen’s Kappa on GNOME and KDE data computed at token level was 0.66 for English and 0.53 for Italian, which corresponds to a substantial and moderate agreement following Landis and Koch (1977). This annotation effort resulted in the identification of 874 domain-specific bilingual terms in the two datasets.7

5 Evaluation

In this Section, we report the quality of monolingual term extraction and the bilingual alignment. For each domain we evaluate the performance obtained by applying different approaches to the integration of bilingual terms into an SMT system. Evaluation of the extracted monolingual and bilingual terms is performed on the manually annotated KDE and GNOME datasets by calculating precision, recall and f-measure. The BLEU metric (Papineni et al., 2002) is used to automatically evaluate the translation quality of the EMEA and the IT manual datasets.

5.1 Monolingual Term Extraction

Our first evaluation concerns monolingual term extraction from English and Italian documents provided by the KX, AlchemyAPI and TWSC extraction tools.

As shown in Table 1, KX tends to overgenerate when extracting English terms. It extracts the highest number of expressions, which results in a high recall, but low precision. On the other hand, TWSC extracts the least English terms. Based on F1, we observe that AlchemyAPI is the best performing tool when extracting English terms. On Italian data, TWSC achieves the best F1 score, while AlchemyAPI performs worst due to the lack of Italian resources within the Linked Open Data (LOD) cloud.8 KX shows a similar behaviour when extracting terms both from English and from Italian data, i.e. low precision and high recall. In summary, we select AlchemyAPI as the best performing term extractor for English and TWSC for Italian, to be used in the next phase.

5.2 Bilingual Term Alignment

In this step, we evaluate our strategies (i.e. Word Alignment and SMT n-best) to align monolingual terms and compare them against the performance of Term Aligner (see Section 2.2). We consider two different settings: in the first one, we use the two monolingual lists, which are automatically extracted by AlchemyAPI for English and TWSC for Italian. In the second one, instead, parallel terms are built starting from the monolingual terms, which were manually annotated to create the gold monolingual datasets.

Focusing on the translation projections, in the top part of the Table 2 (real situation with automatically extracted terms), we observe that the term lookup approaches, where the alignments are generated by word alignment and SMT n-best method, are too restrictive and output few bilingual terms, resulting in high precision but low recall. The sentence lookup strategies

---

### Table 1: Evaluation of monolingual term extraction for English and Italian

<table>
<thead>
<tr>
<th></th>
<th>GNOME - KDE (English)</th>
<th>GNOME - KDE (Italian)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>KX</td>
<td>AlchemyAPI</td>
</tr>
<tr>
<td># of Terms</td>
<td>1115</td>
<td>665</td>
</tr>
<tr>
<td>Precision</td>
<td>0.293</td>
<td>0.393</td>
</tr>
<tr>
<td>Recall</td>
<td>0.596</td>
<td>0.571</td>
</tr>
<tr>
<td>F1</td>
<td>0.393</td>
<td>0.466</td>
</tr>
</tbody>
</table>

---

7 The annotated data are made freely available to the research community under http://hlt.fbk.eu/technologies/bittercorpus
8 http://linkeddata.org/
Table 2: Bilingual term alignment using the automatically extracted monolingual terms and the gold standard

<table>
<thead>
<tr>
<th></th>
<th>Translation Projection</th>
<th>Term Aligner</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Word Alignment</td>
<td>SMT n-best</td>
</tr>
<tr>
<td>sent. lookup</td>
<td>term lookup</td>
<td>term lookup</td>
</tr>
<tr>
<td>Precision</td>
<td>0.027</td>
<td>0.192</td>
</tr>
<tr>
<td>Recall</td>
<td>0.270</td>
<td>0.101</td>
</tr>
<tr>
<td>F1</td>
<td>0.233</td>
<td>0.164</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Word Alignment</td>
<td>SMT n-best</td>
</tr>
<tr>
<td>sent. lookup</td>
<td>term lookup</td>
<td>term lookup</td>
</tr>
<tr>
<td>Precision</td>
<td>0.463</td>
<td>0.768</td>
</tr>
<tr>
<td>Recall</td>
<td>0.399</td>
<td>0.285</td>
</tr>
<tr>
<td>F1</td>
<td>0.425</td>
<td>0.415</td>
</tr>
</tbody>
</table>

Table 2: Bilingual term alignment using the automatically extracted monolingual terms and the gold standard

are more tolerant, identifying more bilingual terms and having a better recall. In terms of F1, the SMT n-best strategies have better scores compared to word alignment methods. This is due to the possibility to select a correct target term from the n-best translations and not only from the single option generated by word alignment. As for the Term Aligner tool, we run experiments with different cognate similarity thresholds from 0.1 to 1.0 with steps of 0.1, and a classifier trained on the EuroVoc data, as reported in the original paper by Aker et al. (2013). The best performance on term alignment is achieved with threshold of 0.5, and, in general, this method tends to align fewer bilingual terms but with high quality. Nevertheless, the alignment quality is substantially lower compared to the translation projection approaches. This can be deduced from the difference between the bilingual terms used to train the classifier and our test set.

When using monolingual terms provided by human annotators (bottom part of Table 2), we obtain significantly higher results compared to the real scenario described before. In this case, the SMT term-lookup method performs best. This implies that term lookup is more sensitive to the heterogeneity in automatically extracted data than the approach based on sentence lookup.

Term Aligner often obtains a precision close to 1, which is similar to the original results reported by Aker et al. (2013). This indicates that the method performs very good if it operates with high-quality data like our gold standard or the EuroVoc dataset. Nevertheless, it is sensitive to domain specificity and to the homogeneity of the terms to be aligned.

In summary, we compared several alignment approaches, i.e. Translation Projection with Word Alignment and SMT n-best method, both in combination with sentence and term lookup. Our evaluation included also Term Aligner with different thresholds. The SMT n-best approach always outperforms the others, whereby Term Aligner is negatively affected by heterogeneous data, showing the lowest performance with automatically extracted monolingual terms.

5.3 Translation Evaluation

After identifying the best tool for monolingual term extraction and the best approach for bilingual alignment, we carry out the final translation evaluation, based on the EMEA and IT manual datasets.

As described in Section 3, we split our data into several partitions and each of them is translated by: (i) a baseline SMT system that was built with the general resource, without embedding terminology; (ii) XML markup approach to embed the terminology paired with the
baseline SMT system; (iii) cache-based model, where the bilingual terminology was used to generate CBTM and CBLM in support of the general SMT system. The probability passed to the XML markup for each bilingual term is set according to the translation probability obtained by the SMT system used to project the source term onto the target language. Since a source term may have different translation candidates, the different translation probabilities give preference to more probable translations. Furthermore, XML markup cannot handle overlaps between dictionary entries. In our experiments, we found only 15 cases where the entries overlap, whereby we give preference to longer source terms.

For each set of partitions, the incremental tuning was run to update the log-linear weights. For a comparison, we also run MERT on each partition starting with flat weights (non-incremental tuning).

In Table 3, we report BLEU scores for each partition separately (columns “Part #”), as well as the evaluation on the whole corpus (column “Document level”). The approximate randomization approach Clark et al. (2011) is used to test whether differences among system performances are statistically significant at document level. Results in the table marked with * are statistically significantly better than the baseline with a p-value $< 0.05$.

Comparing the baseline XML markup and the cache-based methods, we notice that the translation performance of cache-based models always outperforms significantly all the other methods in both domains. This is also confirmed at partition level, with few exceptions for the initial partitions. The XML markup performs better than the baseline in both domains, but statistical significance is obtained only for the IT domain. Among different decay functions in the cache-based models, we report only the negative power decay function of the age, which achieves the best overall performance. This confirms the results described in Bertoldi et al. (2013) also when the approach is applied to a different context. To our surprise, the constant function did not outperform the reported decay function.

At document level, the incremental tuning always outperforms the results obtained starting MERT with flat weights. It is interesting to notice that the gap between the performance obtained by the incremental tuning and the standard approach generally increases partition after partition. This behaviour is more evident for the EMEA corpus, suggesting a more coherent distribution of sentences in the dataset. This favourable situation allows the incremental tuning
to better leverage the optimized weights of the previous partitions. Although the IT data show different levels of difficulty in the partitions (e.g. Partition 2 is easier to be translated than Partition 6), the incremental tuning is still able to smooth such differences and computes weights capable to produce better translations. The proposed framework has shown to be a valuable alternative to the well known XML markup method outperforming it in both domains.

Analysis of bilingual terms in test set  To better understand the performance of our framework, Table 4 reports additional statistics related to (i) the number of extracted terms used by the SMT system to translate the current partition (ii) the number of bilingual terms covered by the baseline phrase table, (iii) the percentage of unique terms that have the source side in the source part of the test set and (iv) the percentage of terms that have the source side in the source part of the test set and the target side in the reference part of the test set.

As expected, the number of terms increases after each partition giving a larger contribution to the SMT system. Analysing the percentage of extracted terms covered by the phrase table, we noticed that on average around 14% of the terms in the IT domain are known by the baseline system, while only 10% are covered for the medical domain. On one hand, this explains the lower performance on average of the baseline system translating the medical domain in comparison to the IT domain. On the other hand, it does not motivate the larger improvements for the IT compared to the EMEA domain, because less terms can contribute to enhance translation quality. Larger improvements in IT are also not supported by the larger number of source terms covered in the medical domain (ninth versus fourth row in Table 4), which indicates that more EMEA bilingual terms are used to cover source spans than the IT domain.

These quantities do not consider two important aspects. The first one is the impact of the target terms on the reference sentences. In the IT, we are able to extract more terms that have the correct translation in the reference test set (on average 85% for IT with peaks larger than 90% against 70% for EMAE). The second aspect is the level of repetitiveness of terms in each document. To estimate it, we compute the repetition rate (Tiedemann, 2010) that measures how often n-grams are repeated in the whole document. Although it is not limited to domain-specific terms, since it includes all possible n-grams, we consider it a good approximation. Both documents are quite repetitive (32.49 for IT and 12.94 for EMEA), but in the IT corpus repetitiveness is more than twice as high than in the EMAE one. These two aspects suggest that the IT corpus contains more domain-specific terms and we are able to provide translations that better fit the references. This last aspect is crucial for the XML markup that, by definition, is more sensitive to the quality of bilingual terms than the cache-based approach. This is confirmed in the EMEA experiments where the XML markup is not able to significantly outperform the
baseline, while the cache-based approach can produce more than 2 BLEU points improvements.

**Manual evaluation of translated sentences** In order to investigate to what extent the approaches differ from a translator’s point of view, we manually inspected the translations produced by the XML markup and cache-based approach. The quality of the two translation versions generally reflects the results reported in Table 3. The XML markup approach tends not to take into account the surrounding context of a translated string, while the cache-based one usually shows a better context-awareness. Specifically, it usually provides a better agreement between adjective and noun (which in Italian bear gender and number information). It also tends to provide more frequently the correct agreement between noun and verb, and even to translate English verbs in the progressive form as nouns, when appropriate. Instead, sentences translated with XML markup often contain gaps as well as agreement and reordering issues because not all terms are translated. We report an example where the source sentence is “Following are the steps for windows operating system.”. The XML markup output is “seguente sono i passaggi per finestre operanti data del sistema.”, while the cache-based translation “seguenti sono i passaggi per finestre sistema operativo.”. In the second version, the agreement between “seguenti” (“following”) and the verb is correct, while it is missing in the XML markup output. Besides, the cache-based model translated “operating system” as a multi-word (“sistema operativo”), while it is translated word by word in the XML markup version.

These differences are more evident in the medical domain, where the language is highly specific and noun phrases are often composed by complex noun chains (e.g. ‘an in vitro mammalian cell assay’, ‘increased lipid and uric acid values’), with implicit underlying dependencies. This is confirmed also by the results reported in Table 3, showing that translation quality is generally lower than for the IT domain.

### 6 Cache-Based Model vs. Online Adaptation Model with cdec

To complete our evaluation, we compare the XML markup and the cache-based approach with the Realtime Adaptive Translation Systems with cdec,9 (henceforth Realtime cdec) an online model adaptation system. Differently from the cache-based approach, it automatically extracts new translation rules from the whole source and post-edited sentences and adds them to the translation grammar. This system takes advantage of cdec (Dyer et al., 2010), a standalone decoder, aligner, and learning framework for SMT. cdec allows us to train word-based and phrase-based models, as well as models based on lexicalized synchronous content-free grammars (SCFG), which was used in our experiment. The adaptation of cdec to work in real time requires the use of Fast Align (Dyer et al., 2013) to perform on-the-fly word alignment between source and post-edited sentences. This makes possible the incremental addition of information to the translation models after a sentence is translated. Furthermore, Realtime cdec adapts the Bayesian language model using the hierarchical Pitman-Yor process approach, whereby MIRA (Chiang, 2012) is used to optimize the discriminative parameters of the decoder.

In our experiments we use the Realtime cdec similarly to the scenario described in Section 3.2. Each sentence pair (source, post-edition) from \( \text{partition}_{n-1} \) is added to the model and used by MIRA to optimise the weights. The initial weights employed by MIRA for the first sentence pair at time \( n - 1 \) are obtained after optimizing the system on the last sentence pair of \( \text{partition}_{n-2} \). When all the sentence pairs from \( \text{partition}_{n-1} \) are added, all the source sentences from \( \text{partition}_n \) are translated. It is worth to notice that this setting favours the Realtime cdec compared to the cache-based or XML markup method, because it adds the whole sentence and not only the bilingual terms.10

---

9 [http://www.cs.cmu.edu/~mdenkows/cdec-realtime.html](http://www.cs.cmu.edu/~mdenkows/cdec-realtime.html)

10 The cache-based model can also take advantage of non-terminological n-grams but it requires alignment between source and post-edited sentences, which is out of the scope of this paper.
Table 5: Comparison between the cache-based method with Moses and the Realtime Adaptive Translation Systems with cdec

<table>
<thead>
<tr>
<th>IT manual</th>
<th>Part 1</th>
<th>Part 2</th>
<th>Part 3</th>
<th>Part 4</th>
<th>Part 5</th>
<th>Part 6</th>
<th>Document level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache-based TM/LM</td>
<td>23.21</td>
<td>35.88</td>
<td>28.01</td>
<td>27.98</td>
<td>30.77</td>
<td>26.84</td>
<td>29.46*</td>
</tr>
<tr>
<td>Realtime cdec</td>
<td>14.25</td>
<td>14.36</td>
<td>27.56</td>
<td>35.85</td>
<td>31.21</td>
<td>39.04</td>
<td>27.90*</td>
</tr>
</tbody>
</table>

Table 5 illustrates the performance translating the IT manuals using our proposed approach and the Realtime cdec. Although both systems were tuned on the same development set, we observe that cdec/MIRA needs more parallel data to adjust the pre-tuned parameters when translating a new domain. Only after adding the sentence pairs from the first three partitions, Realtime cdec is able to outperform the cache-based approach and taking advantage of the content of the whole sentences from the next partitions to substantially improve over the cache-based model. On the contrary, the cache-based approach tuned with MERT shows to be less affected by the change of a domain and performs better with the first partitions at cost of lower performance in the last partitions resulting in a better BLEU score at document level.

Since Realtime cdec enhances its translation capability using the whole source and post-edited sentences, it is difficult to measure the impact of terminology. To overcome this problem, we evaluate only the correctness of the translated terms in the target sentences, and not the whole sentence itself. Therefore, we asked a linguist to manually evaluate the bilingual terms automatically extracted from the IT manuals. 403 out of 627 were marked as correct translation in the domain and we used them to check if our approach and the Realtime cdec are able to correctly translate these terms. On the whole document, we counted 1,538 occurrences of extracted terms in the target sentences generated by cdec. Although the cache-based model is not using the alignment information from the source and post-edited sentence, we counted 1,495 occurrences of terms in target sentences. For both methods, around 90% of these occurrences are correct translations in the references. Moreover, we measure an overlap of 85% of bilingual terms (appearing in source, target and reference sentences) between the cache-based method and Realtime cdec. These results show that both methods, Realtime cdec with the alignment information and our proposed framework embedding the extracted terminology, are able to correctly manage the translations of the domain-specific vocabulary.

7 Related Work

Our work is based on a framework that includes the monolingual extraction of domain-specific terms from a small parallel corpus, bilingual term alignment, and the integration of the bilingual terminology into an SMT system. In the past years, a number of techniques have been applied to the task of bilingual multi-word extraction from parallel or comparable corpora. Most of the work (Daille et al., 1994; Wu and Chang, 2003; Vintar and Fišer, 2008; Kim et al., 2009) focuses on identifying monolingual candidates using linguistic knowledge, statistical methods, or a combination of the two.

As for the bilingual alignment of terms, Aker et al. (2013) cast this task as a classification problem and use the EuroVoc thesaurus as training data. Their work mainly focuses on the quality of the extracted alignments, where the performance often reaches 100% precision. Our approach, however, shows a better performance due to the domain specificity of our dataset. The alignment algorithm proposed by Bouamor et al. (2012) is based on a vector space model. The entries in the vectors are co-occurrence statistics between the terms computed over the en-

---

\footnote{We performed experiments with different C-values of 0.1, 0.01 (default), 0.001, 0.0001, whereby we obtained best results using a C-value of 0.0001 for initial parameter tuning and 0.01 in the learning approach during translation.}
tire corpus. Furthermore, their embedding methods focus on concatenating the newly obtained bilingual data to the existing corpus or adding entries directly into the phrase table. The necessity of dealing with several domains implies the need to keep a large static translation model separate from specific parallel data, e.g. bilingual terminology. Thurman and Aleksić (2012) extract terms and lexicon entries from SMT phrase tables. In their approach they apply linguistic, lexicon and frequency filters to obtain good lexicon entries. Similarly, we also access the phrase table to build our bilingual terminology, whereby our filter relies on the term and sentence lookup approach.

Furthermore, there has been research done on the integration of domain-specific parallel data into SMT, e.g. dictionaries or bilingual terminology, either by retraining new and general parallel resources or adding new entries to the phrase table (Langlais, 2002; Ren et al., 2009; Haddow and Koehn, 2012; Finnis et al., 2012). Furthermore, Okita and Way (2010) investigate the effect of integrating bilingual terminology in the training step of an SMT system, and analyse in particular the performance of a word aligner sensitive to multi-word expressions and translation smoothing. As opposed to their approach, we do not have prior knowledge about the bilingual terminology, since we extract it on the fly based on the document to be translated. As a post-processing step, Itagaki and Aikawa (2008) propose a way to identify terminology translations from SMT output and automatically swap them with user-defined translations. Since the manual development of terminological resources is a time intensive and expensive task, our framework continuously builds bilingual terminology knowledge from the already translated sentences. In order to tackle term translation and the out-of-vocabulary issues, Arcan et al. (2012) used the multilingual web to build a parallel domain-specific corpus based on the vocabulary to be translated. Additionally, Arcan et al. (2014) extend their work focusing on disambiguated term extraction using the rich lexical and semantic knowledge of Wikipedia.

8 Conclusion

In this paper, we propose a framework to enhance translation quality by exploiting bilingual terms extracted from the parallel sentences daily produced by professional translators. The results show that an SMT model enriched with the identified bilingual terms substantially improves translation quality in terms of BLEU score over a generic baseline system. Furthermore, we investigate the integration of the extracted bilingual terms into the SMT system. For the first time we report on the usage of the cache-based model in the context of terminology embedding, whereby we compare results with the widely-used XML markup. The ability of the cache-based model to take into consideration the surrounding context of a translated term allows it to outperform the XML markup approach. In addition, we report a better performance in bilingual term alignment compared to the state-of-the-art Term Aligner.

In the future, we plan to integrate the proposed framework into a professional post-editing environment, measuring the translators’ productivity gain using automatically extracted terminology. Furthermore we plan to combine the strengths of the cache-based model treating a term as one translation unit and the Realtime cdec approach of embedding the incrementally extracted bilingual knowledge from the whole sentence into the translation system.

Acknowledgments

We would like to thank Dr. Ahmet Aker and Mārcis Pinnis for providing us with their newest software and the technical support for it. This publication has emanated from research supported in part by a research grant from Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289 and by the European Union supported project MateCat (ICT-2011.4.2-287688).
References


Clean Data for Training Statistical MT: 
The Case of MT Contamination

Michel Simard  
michel.simard@nrc.ca  
National Research Council Canada  
Multilingual Text Processing  
1200 Montreal Rd., Ottawa, Ontario K1A 0R6 Canada

Abstract  
Users of Statistical Machine Translation (SMT) sometimes turn to the Web to obtain data to train their systems. One problem with this approach is the potential for “MT contamination”: when large amounts of parallel data are collected automatically, there is a risk that a non-negligible portion consists of machine-translated text. Theoretically, using this kind of data to train SMT systems is likely to reinforce the errors committed by other systems, or even by an earlier version of the same system. In this paper, we study the effect of MT-contaminated training data on SMT quality, by performing controlled simulations under a wide range of conditions. Our experiments highlight situations in which MT contamination can be harmful, and assess the potential of decontamination techniques.

1 Introduction  
As Statistical Machine Translation (SMT) systems are becoming more widely used in industry, we see and hear a growing amount of advice on “best practices” for SMT, most notably regarding system training and data collection and preparation. One recurring theme is that of “clean” data: how training data for SMT systems (which often comes in the form of translation memories) should be exempt of any “dirt”, and how SMT users should go about “cleaning” their data. The most visible proponents of this clean-data approach are probably organizations such as TAUS\textsuperscript{1}, Asia Online\textsuperscript{2} and PangeaMT\textsuperscript{3}.

Multiple forms of “dirt” are said to affect SMT quality. Asia Online (2009) discuss a number of examples: formatting codes and other kinds of markup, such as translation memory metadata; untranslated segments (i.e. translation units in which either the target segment is empty, or a copy of the source segment); segments containing inconsistent or mangled character encoding. “Badly” translated segments also fall into this category, although it is not quite clear what it means for a segment to be “badly” translated. Some of the examples given by the authors of the Asia Online study suggest that the presence of translator insertions can be considered “bad”, as when, for example, the translator leaves a source-language term in parentheses for clarity. Non-literal translations are also frowned upon (e.g.: What is a project file? \rightarrow Présentation des fichiers projet), differing usage of punctuation in the source and target languages, etc. The notion of dirt can extend, in some cases, to out-of-domain data, or even in-domain data containing terminology that is inconsistent with the current task. One translator’s

\textsuperscript{1}\url{https://www.taus.net}  
\textsuperscript{2}\url{http://www.asiaonline.net}  
\textsuperscript{3}\url{http://pangeamt.com}
treasure is another translator’s trash.\textsuperscript{4}

The kinds of problems mentioned in Asia Online’s study are typical of SMT training data extracted from translation memories. Increasingly, however, SMT users are turning to the Web to complement their training sets (Resnik and Smith, 2003). There, other types of dirt can be expected to surface. One example is misaligned text: pairs of segments or documents in training data which are not translations of one another (Jiang et al., 2010; Goutte et al., 2012). In this case, the problem does not come from the data itself, but rather from the early processing steps by which the data is collected and organized for SMT training. While in typical translation memory usage, document- and segment-level alignments are validated manually, in Web-collected data, these steps need to be performed automatically. Goutte et al. (2012) estimate that some of the Web-collected data used in WMT evaluations\textsuperscript{5} contain as much as 13% misaligned pairs of segments. Their study nevertheless reveals that typical SMT systems are relatively immune to this sort of dirt, and that even when as much as 30% of the training data is misaligned, SMT quality remains essentially unaffected.

Another kind of dirt comes in the form of machine translated documents: in order to make their online content available in multiple languages, some organizations are known to use unedited MT output. For example, Microsoft is known to have used MT extensively for some of their online technical documentation (Aikawa et al., 2007). More recently, EBay has started using MT for localizing product descriptions (Wohlsen, 2014). When automatically harvesting large amounts of bilingual documents from the Web, there is a growing risk that some unknown proportion of these documents will actually be machine translated.

In general, the presence of machine-generated data in training data that is otherwise assumed to have been produced by humans is seen as a bad thing. Using this kind of data to build systems aimed at simulating human behavior is likely to reinforce the errors committed by other systems, or even earlier versions of the same system. This is what motivates efforts such as those of Google to “watermark” the output of their MT systems (Venugopal et al., 2011), thus making it possible for them to recognize the output of their own systems, and to exclude it from their training sets.

In practice, however, it is not clear that machine-generated translations are always harmful in MT training data. In fact, in some situations, MT has been used specifically to compensate for the lack of adequate bilingual training data. For example, Ueffing (2006) proposed a self-training procedure, in which machine-translated versions of the test data itself was used as an additional source of knowledge in a phrase-based SMT system. This approach, as well as a number of variants, have shown to improve MT quality in different contexts (Bertoldi and Federico, 2009; Schwenk and Senellart, 2009; Lambert et al., 2011). In a somewhat related manner, (Madnani, 2010) and (Dyer et al., 2011) address SMT overfitting issues by adding machine-generated reference translations to their SMT system’s tuning sets.

In the following pages, we focus on the effects of “accidental” MT contamination of MT training data: our goal is to assess to what extent this sort of contamination is harmful in SMT training, and what can be expected from cleaning procedures aimed specifically at eliminating this type of dirt. Our approach is to simulate MT contamination in otherwise clean data, and to measure the effect empirically on SMT systems trained with this data, using standard MT quality metrics. We describe the details of our experimental approach, the experimental data and the MT systems used in Section 2. The main results of our experiments are presented in Section 3, while the potential of data decontamination procedures is assessed in Section 4. Finally, we discuss the general implications of our findings in Section 5.

\textsuperscript{4}Weed is possibly a better metaphor than dirt for many of these issues with SMT training data: “a wild plant growing where it is not wanted and in competition with cultivated plants” (from the Oxford US English Dictionary).

\textsuperscript{5}http://statmt.org/wmt12/translation-task.html
2 Method

2.1 General Approach

The general objective of this study is to assess the impact of MT contamination in SMT training data. Our experimental approach is to inject machine-translated output into an otherwise “clean” training set, then measure the effect on the quality of SMT systems trained with this data. The question we are addressing is whether or not one should worry about MT contamination: How should one handle a given subset of the available training data, knowing that it may contain an unknown proportion of MT output. Therefore all our experiments compare the performance of “baseline” systems, trained exclusively with “clean” data, with that of “augmented” systems, produced by adding auxiliary training data contaminated with MT. We consider systems augmented with varying amounts of auxiliary data, and containing varying degrees of MT contamination.

Our experiments also consider varying baseline conditions, in terms of quantity of baseline training data, text specialization and language pair. Because augmenting datasets from Web documents is a strategy that is mostly used in small-data settings, it makes sense to focus on scenarios where the baseline systems are relatively “small”, trained with as little as 5000 segments in some cases, and not more than 200K sentence pairs. We consider data-collection efforts that aim at increasing training data by at least 50%, and as much as 8 times the size of the baseline dataset.

In the scenarios we wish to simulate, auxiliary data is collected from the Web. However, we wish to avoid biases resulting from differences between the baseline and auxiliary data’s domain or genre. Therefore, in each scenario, experiments are performed using a single, relatively homogeneous pool of bilingual data, for which we have human-quality translations, and process it in different ways so as to simulate the desired conditions. More specifically, we machine-translate this data so as to obtain two different translations for each source sentence in the training data: one human translation, and one machine translation. Different experimental conditions can then be created that correspond to different mixes of human and machine-generated translations.

The quality of the MT in the auxiliary training data should be an important factor in our study: intuitively, we expect “bad” training MT to have a more profound adverse effect on MT quality than “good” training MT. However, this is a factor that is particularly hard to control for. As mentioned earlier, the scenario we are trying to simulate is one where auxiliary data is harvested from the Web. Within this kind of data, MT may have been produced under widely varying conditions. One reasonable assumption is that most of it has been produced using a “generic” (non-specialized or non-domain-adapted) system of decent quality, such as Google Translate. The problem with using Google Translate for our experiments is that we have no control on its training data: if we are to perform experiments using existing bilingual resources (corpora routinely used in research), we have no way of verifying that some or all of this data is not already used by Google to train their system. There are at least two solutions to this problem. One is to select the experimental data in such a way that we can guarantee it has not been used by Google Translate. The second is to use an MT system other than Google Translate, for which we have full control on the training data. For the experiments described here, we opt for the latter strategy. We describe our experimental data and the MT systems used below.

2.2 Data

We describe here the three distinct data sets used in our experiments. More details can be found in Table 1.
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Language</th>
<th>Segments</th>
<th>Source Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>Europarl</td>
<td>English-French</td>
<td>2M</td>
<td>55.5M</td>
</tr>
<tr>
<td>EMEA</td>
<td>French-English</td>
<td>837K</td>
<td>12.7M</td>
</tr>
<tr>
<td>GALE</td>
<td>Chinese-English</td>
<td>547K</td>
<td>17.7M</td>
</tr>
</tbody>
</table>

Table 1: Experimental data

**Europarl** This is release v7 of the well-known Europarl corpus (Koehn, 2005), in English and French. We arbitrarily set the source language to be English. We set aside 2000 randomly picked pairs of segments for tuning purposes. The test set for this dataset is the Europarl test set from the WMT 2008 shared task (Callison-Burch et al., 2008).

**EMEA** This is release 0.3 of the EMEA dataset, from the OPUS Corpus (Tiedemann, 2009) in English and French. In this case, we chose French to be the source language. This is a parallel corpus made out of PDF documents from the European Medicines Agency. While highly technical, it is very repetitive by nature. The data is organized into individual documents. We respected these natural divisions when subsampling the data: we randomly picked 60 documents from the complete dataset, and assigned 30 for tuning and 30 for testing. We took at most 100 segments from each document, to avoid having a few documents overwhelm the tuning or test sets. This procedure yielded 2185 and 1832 tuning and test segments respectively.

**GALE** This is a collection of Chinese-English corpora from the DARPA GALE initiative, the larger part coming from the FBIS corpus. For tuning and test, we used the NIST 04 (1788 segments) and NIST 05 (1082 segments) Chinese-English test sets respectively. Four reference translations are provided for each of these sets.

### 2.3 MT Contamination

In our experimental setup, MT systems play two different roles: they are first used as “contaminators”, i.e. to produce machine-translated target-language versions of training segments; then, they are used as “machine learners” in the rest of the study.

To generate MT contamination in the GALE dataset, we used an older version of SYS-TRAN’s Chinese-English machine translation system (Yang et al., 2003). This is a rule-based system, customized for the domains of science and technology. It uses a rule-based word-segmenter and a bilingual lexicon with about 1.2M entries, containing words, expressions and rules.

For the English-French datasets (Europarl and EMEA), MT contamination was generated using SMT systems based on Portage (Larkin et al., 2010), the NRC’s phrase-based SMT technology. English-French and French-English systems were trained using a very large corpus of Canadian government data harvested from the Web (domain gc.ca), containing over 500M words in each language. Phrase extraction was done by aligning the corpus at the word level using both HMM and IBM2 models, using the union of phrases extracted from these separate alignments for the phrase table, with a maximum phrase length of 7 tokens. The following feature functions are used in the log-linear model: 5-gram language model with Kneser-Ney smoothing (Kneser and Ney, 1995); lexical estimates of the forward and backward probabilities obtained either by relative frequencies or using the method of (Zens and Ney, 2004); lexicalized distortion (Tillmann, 2004; Koehn et al., 2005); and word count. The parameters of the log-linear model were tuned by optimizing BLEU on the development set using the batch variant of MIRA (Cherry and Foster, 2012). Decoding uses the cube-pruning algorithm of (Huang and Chiang, 2007) with a 7-word distortion limit.

Table 2 shows the performance of these contaminator systems on the test sets used in our
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Test Segments</th>
<th>Tokens</th>
<th>BLEU</th>
<th>WER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Europarl</td>
<td>2000</td>
<td>59 936</td>
<td>26.05</td>
<td>65.66</td>
</tr>
<tr>
<td>EMEA</td>
<td>1832</td>
<td>32 601</td>
<td>31.31</td>
<td>56.90</td>
</tr>
<tr>
<td>GALE</td>
<td>1082</td>
<td>33 064</td>
<td>19.65</td>
<td>76.03</td>
</tr>
</tbody>
</table>

Table 2: MT Contaminator Performance

<table>
<thead>
<tr>
<th>Corpus</th>
<th>Baseline</th>
<th>+8× aux.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>segments</td>
<td>tokens</td>
</tr>
<tr>
<td>Europarl</td>
<td>222K</td>
<td>6.2M</td>
</tr>
<tr>
<td>EMEA</td>
<td>93K</td>
<td>1.5M</td>
</tr>
<tr>
<td>GALE</td>
<td>60.6K</td>
<td>2.0M</td>
</tr>
</tbody>
</table>

Table 3: Sizes of training datasets: “Baseline” contains clean (uncontaminated) data only; in addition, “+8× aux.” data also contains contaminated data totalling 8 times the size of the baseline data.

experiments. Using these MT systems, we built sets of training data containing varying degrees of MT contamination, ranging from 0.00 (uncontaminated – or clean – data) to 1.00 (data in which all the target language text is in fact MT output).

As much as possible, when generating contaminated training data, we tried to mimick a realistic situation, in which whole documents, rather than individual segments, are either clean or dirty. This is important, because MT systems, just like human translators, are known to translate terms relatively consistently within documents (Carpuat and Simard, 2012); and because all occurrences of a rare term tend to “bunch up” (Church and Gale, 1995), an MT system is likely to learn its translation from a small number of documents, sometimes a single one. In the case of the EMEA data, since we had access to the document structure, simulating this effect was straightforward. For GALE and Europarl data, we had to settle for an approximation: the original order of segments was preserved, and data was arbitrarily segmented into blocks of 50 contiguous sentences (in other words: we assumed 50-sentence documents).

Contamination was always produced by translating training data in the same direction as the system which it was used to train; for example, when building MT systems to translate from Chinese to English, we used training data containing Chinese source segments paired with their English machine translation. We speculate that this sort of contamination is more likely to have an adverse effect on MT performance than the opposite, and so our setup corresponds to a worst case scenario.6

3 Results

We built multiple MT systems for each dataset, starting with a baseline system trained exclusively with clean data, then gradually adding contaminated auxiliary data. Table 3 gives the sizes of the smallest and largest training sets for each test domain.

The “machine-learners” used in all experiments were also built using the Portage SMT toolkit, and a setup similar to that of the English-French contaminator MT systems (Section 2.3). The main difference is that here, we used 4-gram language models instead of 5-gram, which are more appropriate with smaller amounts of training data.

Figure 1 shows learning curves for the MT systems trained with these datasets. The size of the auxiliary data is expressed relative to the size of the baseline training data: 0 means “no

6Note that this is in line with Lambert et al. (2011)’s suggestion that, when using MT as complementary training data, it is better to use target-language corpora machine-translated into the source-language, rather than the opposite.
auxiliary data” (baseline conditions), 0.5 means “auxiliary data is half the size of baseline data”, etc.

With clean data (black curves), system performance increases more-or-less monotonically as more training data is used, as is usually the case with SMT systems. Depending on the text domain and the initial state of the system, this increase can be quite modest (Europarl) or surprisingly large (EMEA). But in all cases, the behavior is essentially the same. As the degree of MT contamination increases, however, the gains in performance diminish. This reduction is barely visible – and in some cases not statistically significant – with 5% contamination (red curves), but becomes more apparent as the level of contamination increases. In the case of Europarl and EMEA, when the auxiliary training data entirely consists of MT output (100% contamination level – light blue curves), the performance of the resulting system actually decreases with auxiliary data, regardless of the actual amount of data. For Europarl, the contamination threshold at which adding auxiliary training data becomes harmful seems to be somewhere around 0.5: at this level, the performance of the system remains stable as more data is added. For EMEA, this threshold appears to be above 0.5.

In the case of the GALE data, the performance of the system always improves, regardless of the degree of contamination. In this scenario, MT contamination was produced using a rule-based system (SYSTRAN). Complementarity effects between rule-based and statistical systems have been observed in the past, although in the very different context of automatic post-editing (Simard et al., 2007; Dugast et al., 2007), which could explain this difference in behavior. However, another important difference between the Europarl and EMEA scenarios on the one hand, and the GALE scenario on the other hand, is that in the latter, the performance of the baseline system (without auxiliary data; 14.02 BLEU) lies clearly below that of the MT contaminators themselves on the same data (19.65 BLEU; see Table 2). We conjecture that, as long as the quality of the translation in the training data is better than that of the system under training, performance can only increase.

To verify this hypothesis, we compare the learning behavior of systems augmented with 100% contaminated auxiliary data, under different baseline conditions. Figure 2 shows learning curves for systems trained with Europarl and EMEA data; each curve corresponds to a different baseline system.

In the case of Europarl data, the smallest baseline system was trained with only 12.5K clean sentence pairs: at 22.4 BLEU, its performance is below that of the MT contaminators (26.1 BLEU – dotted line). That system always benefits from more auxiliary data, eventually surpassing the MT contaminators. At the opposite end of the spectrum, a Europarl baseline system trained with 200K segments of clean data, with 29.5 BLEU, can only lose from the addition of 100% contaminated data. Intermediate systems behave in between these two extremes. An interesting case is the 50K baseline Europarl system, which initially loses 1 BLEU from doubling its training set with contaminated data, but eventually regains it, as more contaminated data is added. It is worth noting that all systems eventually perform better than the MT contaminators, which is likely explained by the presence of clean in-domain data in the training set (baseline system training data).

A somewhat similar pattern can be observed with EMEA systems. Interestingly in this case, with large amounts of auxiliary training data, all systems converge to almost identical performances, very close to that of the MT contaminator’s. In essence, this suggests that all of these systems eventually mimic the behavior of the MT contaminators. This result is in line with those of Dugast et al. (2008), who “relearn” a rule-based MT system with an SMT system by using the former system to generate training data for the latter.

\footnote{The GALE dataset is too small to produce reliable learning curves with baseline systems performing above the MT contaminator level.}
Figure 1: BLEU and WER scores of trained MT system as a function of auxiliary training data size (expressed as a factor of baseline data size); each curve corresponds to a different level of MT contamination in auxiliary data.
Figure 2: Learning Curves (BLEU score as a function of total amount of training data: baseline + auxiliary) for Europarl and EMEA domains, using 100% contaminated auxiliary data; each curve corresponds to a different amount of (clean) baseline training data. Dotted line is BLEU score of corresponding MT contaminator system on test data.

4 Decontaminating Training Data

Because “dirty” data takes many different forms, data-cleaning procedures are varied. In the case of extra-textual objects or encoding issues, it typically takes the form of in-segment filtering or case-by-case normalization; for misalignment, re-alignment may be necessary; for “bad” translations, re-translation is sometimes considered (Asia Online, 2009). In all cases, filtering out whole segments or even complete documents is usually the simplest solution. And in the case of MT contamination, filtering out dubious portions of the dataset is possibly the only reasonable option.

However, detecting MT is not an easy task, especially when it may be hiding inside large quantities of data, and if it comes from an arbitrary number of different MT systems. Somers et al. (2006) explore different ways of detecting inappropriate uses of online MT systems by language students in translation assignments, a task which is related to that of detecting MT. But their approach is based on comparing human production with MT output, an approach which would be difficult to apply in our case. As mentioned in the introduction, watermarking has also been considered for detecting MT-contaminated data (Venugopal et al., 2011). Unfortunately, this is not a general solution to MT detection because it only allows one producer of MT (e.g. Google) to recognize output from its own systems. In fact, Venugopal et al. suggest that no generally-applicable automatic method exists to distinguish between human- and machine-generated translations. Kurokawa et al. (2009) propose a method to determine whether a given version of a text is the original or a translation; a similar approach could arguably be used for our problem, given suitable training data. To our knowledge, this has not been done, and no such dataset is available.

But what if we knew how to detect MT? What if it was possible to reliably detect contaminated segments or documents, and filter them out of the auxiliary training data? To examine the potential of such methods, we trained systems with artificially decontaminated datasets. Figure 3 compares the behavior of these systems with those obtained with the corresponding contaminated datasets. Each curve plots the BLEU (WER) gain of a system for a given amount of auxiliary data, as a function of the degree of contamination: solid lines correspond to “raw” (contaminated) data, while dashed lines correspond to the same data after decontamination. Be-
cause these are “oracle decontaminations”, the results can be interpreted as an empirical upper bound on the MT quality that could be obtained from an actual “MT detector”.

As can be seen in these plots, when the contamination level of the auxiliary data is 0 (left edge of the graphs), filtering has no effect, and thus both systems produce identical results. When all auxiliary data is MT (right edge of the graphs), filtering eliminates all auxiliary data, and filtered systems display a gain of 0 (black horizontal line); in the case of EMEA and Europarl, the corresponding unfiltered systems display negative gains (i.e. losses), while the GALE systems retain positive gains, as observed earlier. In between these two extremes, the effect of decontamination is generally positive (although there are exceptions). Performance varies as a function of contamination level: the greater the contamination, the greater the effect of decontamination; and as a function of the size of the auxiliary dataset: the greater the amount of data, the greater the effect of decontamination. Yet, it is striking that in most “realistic” scenarios (MT contamination below 20%) the net effect of decontamination is negligible. This suggests that, if the goal is to improve the general quality of the output MT, it is probably not worth investing heavily in developing an MT detector for the specific purpose of cleaning up SMT training data.

5 Discussion

Should one worry about MT contamination when collecting auxiliary training data for SMT systems? It is difficult to answer this question based solely on the results of the experiments reported in the previous sections. One problem is that we have no idea what the actual level of MT contamination is on the Web. In practice, this is likely to depend on various application-dependent factors: where and when the data was collected, what domain and genre, and for which language pair. For some domains and languages, for example English-French legal texts, contamination may be expected to be negligible. For others, for example Russian-English product reviews and descriptions, it may be much higher. In the absence of general MT detection methods, the only way to reliably estimate the level of contamination is to analyse random samples of the collected data manually.

In our experiments, using fully machine translated (i.e. 100% contaminated) auxiliary training data most often degrades MT performance. This appears to contradict earlier results (Ueffing, 2006; Bertoldi and Federico, 2009; Schwenk and Senellart, 2009; Lambert et al., 2011), in which machine translated training data was successfully used to improve MT. The main difference with our work is that all these previous studies were concerned with domain-adaptation: the test and baseline training data were assumed to come from different domains, while the MT auxiliary data came from the same domain as the test data. In our case, all data – test, baseline and auxiliary training – come from the same domain. Again, our setup reflects a worst-case scenario, in which the auxiliary training data is not assumed to be closer to the test than the baseline training data, and our results must be interpreted accordingly.

Another important factor to consider when using potentially contaminated data is the intended application for the trained MT systems. To better understand this point, it is useful to examine some example translations. Table 4 compares example translations for two of our experimental systems on the EMEA data: the Baseline system, and the system trained with all available auxiliary training data, under the assumption that this was 100% contaminated (all translations are MT). Example 1 shows how natural “translation memory effects”, which can be normally observed when a domain is by nature very repetitive (as the EMEA), can be lost with contaminated data: a frequent sentence, for which the baseline system “knew” a correct translation, is now translated differently by the augmented system. To a certain extent, this phenomenon may be exacerbated in our study by our experimental methodology: in practice, finding a complete sentence verbatim in a Web-harvested corpus is probably quite unusual.
Figure 3: BLEU and WER gain relative to baseline, as a function of MT contamination level in auxiliary training data. Each pair of solid and dashed lines corresponds to a different amount of auxiliary training data (1, 4 and 8 times the size of the baseline dataset): solid lines correspond to contaminated data; dashed lines correspond to filtered (decontaminated) data.
1. for the full list of all side effects reported with Viagra, see the package leaflet.

2. subsequently, the dose should be adjusted in case by cases every 1 to 2 semaine(s) to maintain the average neutrophil count between $1.5 \times 10^9/l$ and $10 \times 10^9/l$.

Table 4: Translation examples from the EMEA test domain. Baseline translations are produced by the baseline MT system (see Table 3); + 100% aux. translations are produced by a system trained with all available auxiliary training data (8 times the baseline data), 100% contaminated.

<table>
<thead>
<tr>
<th>Source</th>
<th>Reference</th>
<th>Baseline</th>
<th>+ 100% aux.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>French</td>
<td>English</td>
<td>French</td>
</tr>
<tr>
<td>1. pour une description complète des effets indésirables observés sous Viagra, voir la notice.</td>
<td>for the full list of all side effects reported with Viagra, see the package leaflet.</td>
<td>for a full list of all side effects reported with Viagra, see the package leaflet.</td>
<td>for a complete description of adverse effects observed under viagra, see the notice.</td>
</tr>
<tr>
<td>2. par la suite, la dose doit être ajustée au cas par cas toutes les 1 à 2 semaines(s) pour maintenir le taux moyen de neutrophiles entre $1,5 \times 10^9/l$ et $10 \times 10^9/l$.</td>
<td>subsequently, the dose may be individually adjusted every 1 - 2 weeks to maintain the average neutrophil count between $1.5 \times 10^9/l$ and $10 \times 10^9/l$.</td>
<td>thereafter, the dose should be adjusted in case by case every 1 to 2 semaine(s) to maintain the average neutrophil count between $1.5 \times 10^9/l$ and $10 \times 10^9/l$.</td>
<td>subsequently, the dose must be adjusted on a case-by-case basis every 1-2 semaine(s) to maintain the average rate of neutrophils between 1.5 and 10 x 10^9/l.</td>
</tr>
</tbody>
</table>

Table 5: Translation examples from the Europarl test domain. See Table 4 for details.

<table>
<thead>
<tr>
<th>Source</th>
<th>Reference</th>
<th>Baseline</th>
<th>+ 100% aux.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>French</td>
<td>English</td>
<td>French</td>
</tr>
<tr>
<td>1. ensuite, il y a, là encore, c’est inévitable, mais néanmoins discutable, une part de arbitraire, on le comprend, dans les choix opérés par l’institut de florence.</td>
<td>ensuite, il y a une fois de plus et c’est inévitable, encore contestable, un élément de caractère aléatoire, ce qui est compréhensible, dans les décisions prises par l’institut de florence.</td>
<td>ensuite, il y a - et encore une fois, il est inévitable, mais discutable - un élément du caractère aléatoire, ce qui est compréhensible, dans les décisions prises par l’institut européen de florence.</td>
<td></td>
</tr>
<tr>
<td>2. je devrais être reconnaissants si le tribunal devrait émettre son opinion sur la réforme de la réglementation financière bientôt.</td>
<td>je serais reconnaissant la cour serait question son avis sur la réforme du règlement financier bien tôt.</td>
<td>je remercie la cour des comptes de bien vouloir émettre rapidement son avis sur la réforme du règlement financier.</td>
<td></td>
</tr>
</tbody>
</table>

But the same example also highlights a more frequent problem: the augmented system may “unlearn” official terminology and standard phraseology from the baseline corpus. For example, the term “side effect” which becomes “adverse effects”, or “package leaflet” which is now translated as “notice”. In Example 2, “average neutrophil count” becomes “average rate of neutrophils”. If the domain is very sensitive to terminology and terminological consistency, this may be a serious problem.

If, however, the emphasis is more on fluency and getting the meaning right, access to more data may turn out to be beneficial, possibly more so than the BLEU and WER scores would have us believe. Examples from the Europarl domain (Table 5) show situations where the translation from the augmented system, although not as close to the reference as the baseline, is actually more fluent and adequate. This seems to be due partly to a better coverage of the source vocabulary (e.g. randomness) and to better handling of idioms.
6 Conclusion

We have presented a study on the effect of MT contamination in training data on SMT performance. Our focus was on scenarios where an existing baseline MT system is augmented with Web-collected data, which may contain arbitrary quantities of machine-translated contents. Our experiments demonstrate that MT quality is systematically affected by contaminated training data; severely contaminated data can even decrease the quality of translations compared to the baseline. In all cases of mild contamination, however, the adverse effects are usually quite small, unless very large amounts of data are involved, or in the case of highly technical and/or repetitive material.

Automatic methods to reliably distinguish between human and machine translations would potentially be useful for decontaminating Web-collected data. However, it is uncertain that such methods can be deployed in practice. In the presence of potentially contaminated data, an alternative approach might be to identify which part of the collected training data is more likely to be useful. For example, one could easily identify pairs of segments that contain previously unseen vocabulary. Similarly, one could filter out segments or phrase-pairs that contain domain-specific terminology, or unusual or inconsistent translations of this terminology.

Yet another approach with less-reliable training data is to turn to standard domain-adaptation techniques. For example, auxiliary data could be used to generate distinct phrase tables and language models, whose relative weight in the trained systems would be determined empirically based on performance on suitably chosen tuning sets (Foster and Kuhn, 2007). This is something we plan to integrate into our experimental framework in the near future.

When considering using bilingual data automatically harvested from the Web as training data for MT systems, it is important to take the final application into consideration: whether the resulting MT is intended for post-editing terminology-sensitive material, or for gisting and knowledge acquisition. In the end, the best advice is probably to sample the data for quality, and more importantly to monitor the quality of the resulting MT systems, both by appropriate use of standard benchmarks and metrics, including human evaluation.

7 Acknowledgements

We wish to thank Jean Senellart and the people at SYSTRAN for providing the Chinese-to-English translations for the GALE corpus.

References


Bilingual phrase-to-phrase alignment for arbitrarily-small datasets

Kevin Flanagan 667644@swansea.ac.uk
Department of Languages, Translation and Communication, Swansea University, Swansea SA2 8PP, U.K.

Abstract

This paper presents a novel system for sub-sentential alignment of bilingual sentence pairs, however few, using readily-available machine-readable bilingual dictionaries. Performance is evaluated against an existing gold-standard parallel corpus where word alignments are annotated, showing results that are a considerable improvement on a comparable system and on GIZA++ performance for the same corpus. Since naïve application of the system for N languages would require N(N - 1) dictionaries, it is also evaluated using a pivot language, where only 2(N - 1) dictionaries would be required, with surprisingly similar performance. The system is proposed as an alternative to statistical methods, for use with very small corpora or for ‘on-the-fly’ alignment.

1. Introduction

The process of extracting phrase pairs from parallel corpora is relevant in several contexts, in particular when inducing a translation model with a Statistical Machine Translation (SMT) system, such as that described by Koehn, Och et al. (2003). Phrase pairs are derived from alignments between parts of a bilingual sentence pair, based on word-alignment probabilities. A related procedure is alignment of parts of parse trees for a bilingual sentence pair, to create models for syntax-based machine translation (MT) (Tinsley, Zhechev et al. 2007, Lavie, Parlikar et al. 2008). What these tasks have in common is a requirement for initial word-level alignment information to be established, typically using a tool such as GIZA++ (Och and Ney 2003). In turn, this requires the corpus to be large to achieve useful results, and above a certain minimum size to produce any results at all.

A different context where alignment below sentence level can be of relevance is when recalling content from Translation Memory (TM) systems, whether in order to propose translations for sub-sentential matches (Simard 2003, Planas 2005) or to identify where to edit inexact sentence-level matches (Kranias and Samiotou 2004, Esplà Gomis, Sánchez Martínez et al. 2011). While alignments of that kind can be achieved using statistical methods like those of GIZA++, the size requirements of those methods restrict their usefulness to cases where TMs are sufficiently large. Pre-training GIZA++ on a separate, large corpus does not give very good results, as shown by Esplà Gomis, Sánchez Martínez et al. (2012).

The system presented here aligns parts of bilingual sentence pairs in isolation, without any training step, so has no minimum size requirement. It is intended to be of particular relevance to TM applications, but may also be used to generate sub-sentential alignments in other contexts where they are required and where the data available is limited. It produces hierarchically-arranged pairs of word spans representing the alignments between sentence parts, including alignment of individual words where possible. Alignments between longer spans can be rendered as word alignments for evaluation or use in other contexts. The system is in
no way intended to be a wholesale replacement for an effective and established tool like GIZA++; rather, it provides an alternative in scenarios where GIZA++ use is not practical.

The remainder of this paper is organized as follows. Section 2 describes related work and Section 3 presents the alignment algorithm. Section 4 provides details on evaluation of the system against a parallel corpus with manually-annotated word alignments, while Sections 5 and 6 draw conclusions and discuss further research.

2. Related work

While a great deal of research has been carried out into word alignment of suitably-large parallel corpora – notably the influential ‘IBM Models’ described by Brown, Pietra et al. (1993) – fewer approaches suitable for smaller datasets have been described. A system for aligning regularized syntactic structures described by Grishman (1994) makes use of bilingual dictionaries to generate initial candidate alignments which are then used in tandem with the syntactic structures to extract subtree correspondences. Although the approach is applied to a small dataset (73 Spanish-English sentence pairs), it differs from the system described here in requiring syntactic structure information to guide the alignment process. Conversely, the system described by Planas (2005) applies flat rather than tree-structured sentence analysis, assigning part-of-speech (POS) categories to sentence sections, then using those categories to attempt alignment rather than any lexically-based translation resource. This only provides sub-sentential alignment “as long as the languages processed are parallel enough” (Planas 2005: 5), and requires an analyzer for each language concerned.

The word alignment technique described by Mandreoli, Martoglia et al. (2003) uses neither grammatical nor bilingual dictionary information, but instead applies a number of heuristic approaches to establishing potential points of alignment between sentence tokens – identical punctuation, numbers and proper nouns; similarity of lexical tokens based on Longest Common Subsequence (LCS) – each of which is scored to reflect other factors (e.g. longer LCS scores higher, distant relative token position within sentence scores lower), such that lower-scoring points are discarded before interpolating to produce a final alignment. English-Italian examples are given, where pairings such as ‘electric/elettrico’ and ‘collect/collezione’ illustrate the LCS approach described. Nevertheless, given that it is not unusual for information sequencing to change in translation and that some language pairs exhibit little lexical similarity, this alignment technique seems likely to provide poor results for other languages or text types, and a relatively low accuracy level may be acknowledged where the authors note that “the goal of the word aligner is not to find the rigorous matching between each of the words, but to be able to determine, with good approximation, what target segment a given source segment corresponds to” (Mandreoli, Martoglia et al. 2003: 4).

A system requiring more resources is described by Macken (2010), where sentence pair tokens are first lemmatized and assigned POS tags, by external tools. The sentence pair is then sub-sententially aligned in a two-step process. The first step splits the sentences into ‘chunks’ using a rule-based chunker, while bilingual dictionaries are used to establish tentative lexical correspondences. Pairs of chunks having a high proportion of POS and lexical correspondence are designated as ‘anchor’ chunks. The second step then continues chunk alignment around these anchors, using similar methods combined with some additional heuristics. Performance is evaluated against a set of gold-standard alignments created by annotators for the project, using a variety of text types. Performance statistics are provided with several bilingual dictionaries, of which some are statistically induced – though not using the entirety of the parallel corpora to be aligned – but others are derived from existing bilingual dictionary data. Alignment performance against the gold-standard data indicates that the approach produces useful results. While this system could in principle be used with small datasets, it would re-
quire a lemmatizer, POS tagger and rule-based chunker for each language, as well as suitable
dictionary data.

An interesting approach to aligning nodes in a bilingual phrase-structure parse tree pair is
described by Tinsley, Zhechev et al. (2007). For a parallel corpus, word alignment probabili-
ties are first automatically induced using the Moses toolkit (Koehn, Hoang et al. 2007). For
each sentence pair, hypotheses are constructed, each aligning a node in one parse tree with a
node in the other parse tree. The hypotheses are scored using the word alignment probabili-
ties, and a ‘greedy’ algorithm used to select and retain the most probable, while excluding
those that then contradict well-formedness criteria in relation to those retained. Under their
span1 strategy, scoring of hypotheses with a node dominating a single terminal is deferred
until other hypotheses have been processed. This guides selection in cases such as “where
source terminal x most likely translates to target terminal y but there is more than one occur-
rence of both x and y in a single sentence pair” (Tinsley, Zhechev et al. 2007: 4), and cases
where there exist two different target terminals with induced word alignment probabilities for
a source terminal x, and x is more correctly aligned to the less-probable of the two. Inducing
word alignment probabilities from the corpus to be aligned is not a suitable approach for arbi-
trarily-small datasets, but a similar use of hypothesis evaluation may be, if another source of
word alignment information is used.

In Esplà Gomis, Sánchez Martínez et al. (2012), the term source of bilingual information
(SBI) is used to denote the different resources that can potentially be exploited by their word
alignment method, such as bilingual dictionaries, translation memories, and in particular MT.
Each sentence in a pair to be aligned is split into all possible sub-segments up to a given
length L in words, and the available SBIs are queried for translations of each sub-segment.
Where the translation of the sub-segment is found in the translated sentence, a tentative
alignment is established between the sub-segment and the occurrence of the translation. Once
all such alignments have been established, an alignment score for each word pair is calculated
using a formula to measure the alignment pressure exerted by the translations found, and final
alignments are selected that maximize these scores. Performance is evaluated using three dif-
f erent MT systems in combination as SBIs to align sentence pairs from an existing gold-
standard word-aligned dataset of 400 sentence pairs. Results are measured against the gold-
standard alignments and specifically compared with results using GIZA++ to align the same
data, measured against the same gold-standard alignments, in two different cases, with a
GIZA++ baseline trained only on the test dataset, and with GIZA++ pre-trained on a separate,
much larger parallel corpus. Precision and recall figures show the system gives better results
than pre-trained GIZA++, and results comparable with the GIZA++ baseline, though the au-
thors show that GIZA++ performance deteriorates as the dataset size reduces, while the per-
formance of their system is not in principle affected by having as little as a single sentence
pair, to align ‘on the fly’. Overall, performance of the system appears promising, though they
note that “the weakness of our method is the recall, which may be improved by combining
other SBIs” (Esplà Gomis, Sánchez Martínez et al. 2012: 98). Nevertheless, this system is
specifically intended for use with arbitrarily small datasets and provides a useful comparison
for the system presented in this paper.

3. Alignment algorithm

The algorithm presented here produces hierarchically-arranged pairs of word spans, where
each pairing represents an alignment between the source and target words spanned. For a giv-
en sentence, spans can enclose smaller spans, but may not partially overlap. Sentences are
first tokenized. In the processing that follows, punctuation tokens are first ignored\(^1\), while all other tokens are considered ‘words’. The algorithm then operates in four phases. Firstly, tentative seed alignments are established between words in the source and target sentences, using whatever bilingual dictionary resources are available. Secondly, each sentence is divided into all possible word spans of length two or more, and alignment scores are calculated for a subset of source and target span pairings. A ‘greedy’ selection process then records the highest-scoring pairing as aligned and eliminates pairings involving spans that overlap the recorded spans and seed alignments that contradict the recorded span alignment. (Remaining pairings affected by seed alignment removal are then rescored.) The selection process continues until no further pairings can be recorded. Thirdly, seed alignments whose source and target words occur within recorded span pairings are also recorded as aligned span pairings, as are matching punctuation tokens. Finally, further aligned span pairings are deduced using ‘remainder’ logic, then any redundant span pairings (aligned spans where the words in both spans are all contained within shorter aligned spans) are removed. The following sections describe these phases in more detail. The system as evaluated in section 4 implements each of these phases, using only the external resources described in section 4.2.

### 3.1. Generation of seed alignments

To establish tentative word correspondences for use as seed alignments, a variety of external data resources can be used, where each resource is queried using a given word in either sentence in order to retrieve any available translations for that word. Machine-readable bilingual dictionaries are an obvious example, as are lists of lexical probabilities generated with GIZA++ or similar from a separate, larger parallel corpus, while domain-specific terminology databases can also be queried to provide translations in this way, as can MT systems. Where tools are available, the lemmatized or stemmed forms of words can also be used for querying, which may increase the likelihood of retrieving translations.

For a given query word, any retrieved translations are compared with the words in the actual sentence translation. Where a match is found, a seed alignment is recorded between the query word and the translation word(s), and given a probability based on a number of factors, including provenance (fixed values for terminology databases or MT systems; retrieved values if found in lexical probability lists) and whether the query word and/or matching translation word(s) are lemmatized or stemmed forms. If two or more queries for a given word retrieve translations matching the same translation word(s) – such as when there is agreement between separate external resources, or query results for the lemmatized form match those for the original form – only the highest-probability seed alignment is retained. While queries consist of single words, retrieved translations may often consist of multiple words (e.g. translations of French ‘compenser’ into English may include ‘make up for’). Seed alignment information retains the 1-to-n relationship between those words for use when calculating span alignment score, as described below. In addition to external resources, a heuristic is used to establish further seed alignments. Where a word in one sentence exactly matches a word in the translated sentence (possible proper noun or other non-translated item), a lower-probability seed alignment is recorded between them.

### 3.2. Span pairing selection

Calculating scores for all possible span pairings of all possible spans in two sentences \(S\) and \(T\) is a problem of polynomial complexity. Since 1-word spans are excluded, for a sentence \(S\) of

\(^1\) Initial experimentation found that, with this approach, poorer results were achieved when punctuation tokens were used to generate seed alignments. Comparative results and example cases are omitted here for brevity.
length \( m \) words, there are \( m(m - 1)/2 \) spans to consider, so when aligning with a sentence \( T \) of \( n \) words, there are \( (m(m - 1))(n(n - 1))/4 \) pairings available. To reduce running time, scores are not calculated for pairings considered ‘invalid’ based on the relative span lengths and sentence lengths. For example, with \( S \) of length 15 words and \( T \) of length 17 words, an alignment between a 2-word span in \( S \) and a 12-word span in \( T \) will have very low score using the formulae below. The result of an intuition-based function of span and sentence lengths is used to build a subset of all possible pairings that excludes ‘invalid’ cases.

Each remaining case consists of a span \( s \) in \( S \) and \( t \) in \( T \). In a similar way to Tinsley, Zhechev et al. (2007: 4), the following strings are computed:

\[
\begin{align*}
    s_t &= s_{i_1} \ldots s_{i_k} \\
    t_t &= t_{j_1} \ldots t_{j_y}
\end{align*}
\]

where \( s_{i_1} \ldots s_{i_k} \) and \( t_{j_1} \ldots t_{j_y} \) denote the spans \( s \) and \( t \) respectively, and \( S_1 \ldots S_m \) and \( T_1 \ldots T_n \) denote the set of words in \( S \) and \( T \) respectively. The score \( \gamma \) for a given span pair \((s, t)\) is computed according to (2).

\[
\gamma(s, t) = \alpha(s_{i_1}, t_{j_1}) \cdot \alpha(s_{i_k}, t_{j_y})
\]

Individual string-correspondence scores \( \alpha(x, y) \) are computed using a selection of the seed alignments between \( x \) and \( y \) to create a set of seed alignments \( A \) as described below. Having established \( A \) for strings \( x \) and \( y \), and defining the set of words in \( x \) having seed alignments in \( A \) as \( A_x \), and the set of words in \( y \) having seed alignments in \( A \) as \( A_y \), the score \( \alpha(x, y) \) is calculated as given in (3).

\[
\alpha(x, y) = \frac{2 \cdot \sum_{k=1}^{n} P(A_k)}{2 \cdot |A| + |\{x_i : A_x \ni x_i\}| + |\{y_j : A_y \ni y_j\}|}
\]

The process of selecting the seed alignments between \( x \) and \( y \) for \( A \) merits some explanation. Consider the following sentence pair:

\[
\text{EN: He made good use of the afternoon to make up for lost time by drawing a map.} \\
\text{FR: Il a profité de l’après-midi pour rattraper le temps perdu en faisant un plan.}
\]

Suppose the seed alignments shown in Table 1 have been generated (lemmatized forms in parentheses), and for ease of illustration, all have a probability value of 1.0. The score to attribute to a given string pair should be a function of the number of seed alignments between the pair and the number of words concerned. So, the string pair (“lost time”, “temps perdu”) should score more highly than (“lost time by”, “temps perdu”), since neither of the seeds for ‘by’ (rows 12 and 13 in Table 1) match a word in “temps perdu”. A simple calculation method would be to attribute the seed alignment probability to each word in the strings covered by a seed alignment, sum those probabilities, then divide by the total number of words, in this simplified example giving 1.0 for (“lost time”, “temps perdu”) and 0.8 for (“lost time by”, “temps perdu”). However, in this example it is desirable for (“make up for lost time by”, “rattraper le temps perdu en”) to score higher than (“make up for lost time”, “rattraper le temps perdu en faisant”). With that simple calculation method, however, those pairs would score 0.72 and 0.81, since ‘make’ has a seed alignment with both ‘rattraper’ (row 18 in Table 1) and ‘faisant’ (row 2 in Table 1). To avoid this distortion, when selecting seed alignments to be used for computing string-correspondence scores, a seed alignment \( a \) may only be added to the set if no seed alignment in the set aligns any words also aligned by \( a \). This is referred to herein as the uniqueness requirement when selecting seed alignments with which to score a string correspondence.
A further consideration is applied for 1-to-n seed alignments. In this example, it is desirable for (“He made good use”, “Il a profité”) to score higher than (“the afternoon to make up for”, “profité de l’après-midi pour”). With that simple calculation method, those pairs would score 0.714285 and 0.72 respectively, as there is a seed alignment between ‘make’ and ‘profiter’, and no weight is given to the component words of the translation of ‘profiter’ being found together. To address this, the seed alignment probability attributed to each translated word matching a 1-to-n query result is divided by the number of words in that result, then per (2), the divisor is reduced when scoring spans containing multiple words matching the same 1-to-n seed alignment. This is referred to herein as the grouping adjustment when selecting seed alignments with which to score a string correspondence.

The set $A$ of seed alignments selected to score string pair $(x,y)$ is then assembled by gathering all the seed alignments that exist between $x$ and $y$, applying the grouping adjustment then selecting the highest-probability seed alignments available that meet the uniqueness requirement.

Once scores for the span pairs have been calculated, zero-scoring pairs are discarded, then the ‘greedy’ selection procedure continues per Algorithm 1 selection.

**Algorithm 1 selection**

```plaintext
while span pairs remain in the list
    if there is a single non-pending span pair with the highest score then
        confirm the span pair
    else if there are tied non-pending highest-scoring span pairs whose spans do not overlap then
        confirm those span pairs
    else if there are tied non-pending highest-scoring span pairs whose overlaps meet intersection criteria then
        confirm the intersection span pair(s)
    else
        flag the tied highest-scoring span pairs as pending
        flag all other span pairs involving any highest-scoring spans as pending
        if all span pairs are flagged as pending then
            remove all span pairs
        end if
    end if
end while
```
Algorithm 2 confirm

for each span pair provided
  record the span pair and remove from list
  remove all overlapping span pairs
  discard all seed alignments contradicting it
  rescore affected span pairs
  remove all zero-scoring span pairs
end for
reset all pending flags

In a similar way to Tinsley, Zhechev et al. (2007: 3), where there are tied highest-scoring span pairs, they are left 'pending' while lower-scoring span pairs are examined. However, tied highest-scoring span pairs that have no overlaps with other span pairs having the same score are recorded immediately, since it is desirable to record the highest-scoring pairs wherever possible. Furthermore, intersection criteria are applied when there are tied highest-scoring span pairs. This is much more likely to happen when using external resources where – unlike lexical probabilities generated by GIZA++ – there is no specific probability value associated with query results, and so seed alignment probabilities are assigned based on the provenance of the results, and therefore have relatively uniform values. For the sentence pair at (4), if the seed alignments generated from whatever resources cause the span pairs (“made good use”, “profité de l’après-midi”) and (“to make up for lost”, “l’après-midi pour rattraper”) to have the same score, nothing can be inferred from those two pairs, which in any event are not good-quality alignments. However, if the pairs (“He made good use of the afternoon”, “Il a profité de l’après-midi”) and (“the afternoon to make up for”, “l’après-midi pour rattraper”) have the same score, it is undesirable to flag these two good-quality alignments as pending in order to examine lower-scoring alignments which are a priori less likely to be good-quality. In this case, unlike the preceding low-quality span pairs, there is a level of agreement between the two pairings, in that both English and French spans share intersecting words. These tied span pairs are then considered to meet the intersection criteria, and the intersection span pair (“l’après-midi”, “the afternoon”) is recorded immediately, on the basis that words appearing in both highest-scoring span pairs are most likely to be aligned. While not the same procedure, this technique recalls the similarity template learning heuristic applied in Cicekli and Güvenir (2001) to translation examples with common sequences.

3.3. Seeding alignment confirmation

When no span pairs remain to be recorded as alignments, seed alignments are examined. In increasing order of combined word length (since the shorter the spans, the less chance of ambiguous seed alignments), recorded span pairs are compared with seed alignments. Where a span pair contains a query word that generated only one seed alignment within that span pair, a further span pair is added, aligning the query word that generated the seed alignment with the resulting translation word(s).

3.4. Span pair deduction/reduction

Following span pair alignment as above, it may be possible to deduce further alignments. For example, if the span pair (“lost time by drawing a map”, “temps perdu en faisant un plan”) has been aligned, and the hierarchy contains child span pairs (“lost time”, “temps perdu”) and (“a map”, “un plan”), then a new span pair (“en faisant”, “by drawing”) is recorded as aligned. This process is repeated until no further deductions can be made. Thereafter, redundant alignments in the hierarchy are removed. In this case, the aforementioned parent span pair
 (“lost time by drawing a map”, “temps perdu en faisant un plan”) is removed, since it is completely expressed by contiguous child span pairs.

4. Evaluation

The aligner is evaluated here by comparing the alignments produced by the algorithm using a given set of resources against a manually-aligned gold standard, first using bilingual dictionaries providing direct translations between source and target languages, then using dictionaries that provide those translations via a pivot language. In each case, GIZA++ is also used to align the corpus, and results are again compared with the gold standard.

4.1. Gold-standard data

The main gold-standard data used for evaluation with direct-translation dictionaries was the English-Spanish word-aligned data from the ‘tagged EPPS corpus’ distributed for TC-STAR 2006 evaluation (Lambert, De Gispert et al. 2005), consisting of 400 sentence pairs drawn from the Europarl corpus (Koehn 2005). In that data, only alignments between individual tokens can be recorded, and what might be considered alignments between spans of words are represented by recording a word alignment between every word in one of the spans and every word in the other. The start of this sentence pair provides an example:

EN: Here in Parliament, we have […]
ES: En esta Asamblea hemos […]

Between the words ‘Here in Parliament’ and ‘En esta Asamblea’, there are a total of nine alignments annotated, linking each of the three words with each of the three translation words. The data also distinguishes between word alignments that are ‘sure’ and those that are ‘possible’. Of those nine alignments, there are two that are annotated as ‘sure’, aligning (‘in’, ‘En’) and (‘Parliament’, ‘Asamblea’). The system under evaluation also generates alignments between spans of words, but records them as distinct spans rather than multiply-linked word alignments. In order to produce alignment data from the system that could be compared more readily with the gold-standard data, the alignment data produced was first subject to an automated conversion. For each aligned span pair, each word not subject to a child span alignment was aligned to each such word in the other span.

For the purposes of evaluating aligner performance for a different language pair with direct-translation dictionaries, 20 English sentences were taken from the tagged EPPS corpus and paired with their German translations in the Europarl corpus from which the English sentences were originally drawn. These English-German sentences pairs were then manually word-aligned by a single annotator using the same principles as applied for the English-Spanish tagged EPPS corpus, to create a small English-German manually-annotated gold-standard corpus against which to test the system.

In order to have gold-standard data against which to evaluate the system operating with pivot-language dictionaries, 20 Spanish sentences were also taken from the tagged EPPS corpus, and paired with their French translations in the Europarl corpus from which the Spanish sentences were originally drawn. These French-Spanish sentence pairs were also then manually word-aligned by a single annotator using the same principles as applied for the English-Spanish tagged EPPS corpus, to create a small French-Spanish manually-annotated gold-standard corpus against which to test the system.
4.2. External resources

For this evaluation, the external resources used were lemmatizers for English, Spanish and French, and a number of machine-readable bilingual dictionaries: French-English (20,403 entries), English-French (21,498 entries), German-English (127,879 entries), English-German (121,380), Spanish-English (17,243 entries) and English-Spanish (20,820 entries). Lemmatizers used were: for English, Morfologik 1.6<sup>2</sup>; for Spanish, Freeling 3.0 (Carreras, Chao et al. 2004); for French, a purpose-built lemmatizer using the data from the Morphalou project (Romary, Salmon-Alt et al. 2004); for German, a purpose-built lemmatizer using the data distributed with the Morphy analysis tool (Lezius, 2000). Dictionaries used were all from the XML Dictionary Exchange Format project<sup>3</sup>. (The files exhibited some corruption and omission, repaired manually.)

4.3. Metrics

Precision and recall were computed versus the gold-standard corpora for the alignments produced both by the system presented here and by GIZA++. These were then combined to obtain the F-measure. These three metrics were computed in two ways, for only the ‘sure’ alignments in the gold standard, and for all alignments in the gold standard.

4.4. GIZA++

Alignments for the test corpora were also produced using GIZA++, running it in both directions (source to target and target to source) then combining both sets of alignments using the grow-diag-final-and heuristic (Koehn, Och et al. 2003).

4.5. Results

Table 2 shows the results obtained aligning the English-Spanish gold-standard corpus of 400 sentence pairs using the system described above with direct-translation dictionaries, compared with the results reported for the same corpus using the system described by Esplà Gomis, Sánchez Martínez et al. (2012) and with alignment of the same corpus using GIZA++.

<table>
<thead>
<tr>
<th>Alignment type</th>
<th>Dictionary-based aligner</th>
<th>GIZA++</th>
<th>Esplà Gomis et al</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F</td>
</tr>
<tr>
<td>‘sure’ only</td>
<td>76.2%</td>
<td>61.3%</td>
<td>66.6%</td>
</tr>
<tr>
<td>all</td>
<td>81.4%</td>
<td>46.0%</td>
<td>57.6%</td>
</tr>
</tbody>
</table>

Table 2: Precision (P), recall (R) and F-measure (F) produced for ‘sure’ alignments, and separately for all alignments, when aligning the EN-ES gold-standard corpus of 400 sentence pairs.

The results show that the system described in this paper produces significantly higher precision than GIZA++, with slightly lower recall for ‘sure’ alignments and more noticeably lower recall for ‘sure’ and ‘possible’ alignments taken together. This raises interesting questions about which metric and which alignment type is of more importance for a given application. For use in TM as described by Simard (2003) and Planas (2005), alignment quality will have a direct bearing on translation suggestions recalled from the TM. In that context, high precision is arguably of some importance, since the lower the precision, the more ‘noise’ there is likely to be in the results, undesirably distracting the translator. The same consideration applies to the similarity coefficient threshold used to recall TM ‘fuzzy matches’, where “users
are generally advised not to set the similarity coefficient too low, to avoid being swamped by
dissimilar and irrelevant examples” (Macklovitch and Russell 2000: 4). For similar reasons,
matches from ‘sure’ alignments are more likely to be of immediate use than ‘possible’ alignments. The results also show that the system described here produces higher recall and signifi-
cantly higher precision than that achieved by Esplà Gomis, Sánchez Martínez et al. (2012)
using the same corpus.

Nevertheless, an alignment system using bilingual dictionaries may be of less use in a
TM or other translation context if for use with N languages, N(N - 1) bilingual dictionaries are
required. This could be reduced to 2(N - 1) dictionaries if the system can be used with a pivot
language. Table 3 shows the results from using the system to align the small French-Spanish
corpus described above in this way, specifically, by ‘chaining’ translations from the French-
English dictionary to the English-Spanish dictionary to act as a French-Spanish dictionary,
and similarly combining the Spanish-English and English-French dictionaries to act as a
Spanish-French dictionary.

<table>
<thead>
<tr>
<th>Alignment type</th>
<th>Dictionary-based aligner</th>
<th>GIZA++</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>‘sure’ only</td>
<td>72.0%</td>
<td>67.2%</td>
</tr>
<tr>
<td>All</td>
<td>81.0%</td>
<td>58.2%</td>
</tr>
</tbody>
</table>

Table 3: Precision (P), recall (R) and F-measure (F) produced for ‘sure’ alignments, and separately for
all alignments, when aligning the small FR-ES gold-standard corpus.

It may appear surprising that recall is significantly higher than with the English-Spanish
corpus. Close reading of the results suggest this is because the French and Spanish sentences
are often lexically more similar to each other than either is to the corresponding English sen-
tence, and have more closely-corresponding word order, as with the following example:

FR : Il nous faut deux mois au minimum pour faire ce travail, avec le maximum de
célérité et de sérieux requis.

(5) ES : Necesitamos dos meses como mínimo para hacer ese trabajo, con la máxima ce-
leridad y seriedad requerida.
EN: We needed at least two months to do this work with the required care, even at
maximum speed.

As a result, the dictionary-based seed alignments that fuel the alignment process are more
likely to be confirmed for French-Spanish than for English-Spanish. (Closely-corresponding
word order also results in fewer ‘possible’ alignments in the gold standard.) However, overall
precision is reduced when using a pivot language, typically for sentence pairs with less lexical
correspondence, since the ‘chained’ translation suggestions for a query word can be more
numerous and more distant semantically from the query word. For example, querying a
French-English dictionary for a French word may result in three English translations, then
querying an English-Spanish dictionary for each of those English words may result in nine
Spanish translations in total, making spurious seed alignments more likely. Even so, precision
and recall are both considerably higher than results achieved with GIZA++ for the larger
gold-standard English-Spanish corpus shown in Table 2, although results for that language
combination are not directly comparable with those for the small French-Spanish corpus, and
naturally much higher than the GIZA++ results on this much smaller corpus, shown alongside
in Table 3.

Results from using the system to align the small English-German corpus are shown in Table
4. Although direct-translation dictionaries were used, as for the larger English-Spanish cor-
pus, precision is noticeably lower, while recall is noticeably higher. Performance of the align-
er is conditioned by how many seed alignments are created in the first phase of the process,
that is, the seed density for the sentence pair to align. Where seed density is low, alignments produced are less fine-grained (there are more long spans where individual word correspondences can not be identified), producing more multiply-linked word alignments when converted as described above. Seed density statistics for the corpora used are shown in Table 5, expressed as the percentage of total words in both languages for which a seed alignment was established by dictionary translation or heuristic.

<table>
<thead>
<tr>
<th>Alignment type</th>
<th>Dictionary-based aligner</th>
<th>GIZA++</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
</tr>
<tr>
<td>‘sure’ only</td>
<td>72.1%</td>
<td>63.1%</td>
</tr>
<tr>
<td>All</td>
<td>78.2%</td>
<td>53.3%</td>
</tr>
</tbody>
</table>

Table 4: Precision (P), recall (R) and F-measure (F) produced for 'sure' alignments, and separately for all alignments, when aligning the small EN-DE gold-standard corpus.

Although the German-English and English-German dictionaries are far larger than the other dictionaries used, manual inspection of aligned sentences indicates that they contain relatively few synonyms. In a German sentence containing ‘Unterfangen’, the only translation retrieved is ‘undertaking’, while the corresponding English word is ‘enterprise’, for which the only translation retrieved is ‘Unternehmung’, and therefore no seed alignment is established between those two words. Seed density would be improved by using synonyms during seed alignment generation, even if only for one of the languages concerned. Where span alignments are produced that have low seed density, they have a corresponding low alignment score. For the TM-related applications considered above, this would allow these less-reliable alignments to be rejected when recalling translation suggestions.

5. Conclusions

This paper presented a system producing phrase-to-phrase alignment for arbitrarily-small datasets, whose output can also be expressed as word alignments. The system has the advantages of being able to make use of readily-available machine-readable bilingual dictionaries, requiring no training step, and allowing domain-specific resources such as terminology databases to be easily exploited to assist in alignment accuracy for specialized text types. Evaluation of the system against a gold standard showed precision and recall were considerably better than achieved using the state-of-the-art GIZA++ word-alignment tool when aligning a relatively small dataset (400 sentence pairs). Results from alignment in a pivot-language scenario – albeit on a small set of sentence pairs – indicated that, for N languages, it would be feasible to require only 2(N - 1) dictionaries rather than N(N - 1).

6. Further work

Application of the alignment system to corpora consisting of other language pairs (German-French, German-Spanish, Welsh-English) is currently underway, both using direct-translation dictionaries and pivot-language dictionaries, for further intrinsic evaluation of results against gold-standard data. The algorithm is also to be integrated into a TM system to provide sub-sentential recall, allowing for extrinsic evaluation of performance. This system will optionally word-align TM data (when of sufficient size) using GIZA++, allowing results using the two aligners to be compared, and for GIZA++ to be the default aligner for large TMs. Testing is
also planned to measure the effect of using terminology databases as an external resource when aligning specialized texts with the system described. In that regard, an enhancement to the seed-alignment-generation process is to be developed, to allow for tentative alignments to be established by querying not only with single words, but with short spans of words.

References


A Probabilistic Feature-Based Fill-up for SMT

Jian Zhang  
zhangj@computing.dcu.ie  
Liangyou Li  
liangyouli@computing.dcu.ie  
Andy Way  
away@computing.dcu.ie  
Qun Liu  
qliu@computing.dcu.ie

The CNGL Centre for Global Intelligent Content, School of Computing, Dublin City University, Ireland

Abstract

In this paper, we describe an effective translation model combination approach based on the estimation of a probabilistic Support Vector Machine (SVM). We collect domain knowledge from both in-domain and general-domain corpora inspired by a commonly used data selection algorithm, which we then use as features for the SVM training. Drawing on previous work on binary-featured phrase table fill-up (Nakov, 2008; Bisazza et al., 2011), we substitute the binary feature in the original work with our probabilistic domain-likeness feature. Later, we design two experiments to evaluate the proposed probabilistic feature-based approach on the French-to-English language pair using data provided at WMT 07, WMT13 and IWLST11 translation tasks. Our experiments demonstrate that translation performance can gain significant improvements of up to +0.36 and +0.82 BLEU scores by using our probabilistic feature-based translation model fill-up approach compared with the binary featured fill-up approach in both experiments.

1 Introduction

Like many machine-learning problems, Statistical Machine Translation (SMT) is a data-dependent learning approach. The prerequisite is large amounts of training data in order to generate statistical models. In general, the training data has to be sentence-aligned and bilingual. Some heuristic approaches are often used when deconstructing the training data into phrase-level representations, and the statistical models are computed based on the phrase probability distributions. The generated models are then combined in a log-linear model (Och and Ney, 2002). A basic SMT system may consist of a translation model and a language model, where the translation model provides a target-language translation \( e \) for a source-language sentence \( f \), and the language model ensures the fluency of the target-language translation \( e \).

One challenge which rises above others in SMT is that the translation performance decreases when there are dissimilarities between the training and the testing environments. This type of challenge is often defined as “domain adaptation” in previous work. The underlying reasons that caused domain adaptation challenge are many, but the obvious one is that SMT system training is a complicated data-dependent processing pipeline. It often involves many efforts from various steps, for example, the phrase pair extraction step needs to be consistent with the preceding word alignment step, with one assumption made being that the context of the extracted phrases is irrelevant. In addition, the training sentences are only implicitly visible and become unnecessary once the phrase table is built. In this paper, we try to address the problem...
of phrase-table extraction in a phrase-based SMT training environment, and propose a probabilistic feature-based translation model fill-up approach by creating an inheritance relationship between the extracted phrase pairs and the corresponding bilingual sentence pairs.

Domain adaptation for SMT is a well studied research field. Recently, many new ideas have been introduced, mainly regarding the data adaptation and model adaptation. Most work on data adaptation for SMT focuses on making efficient use of the training data. Liu et al. (2007) use information-retrieval techniques on a transductive-learning framework to increase the count of important in-domain training instances, which results in phrase-pair weights being favourable to the development set. Bicici and Yuret (2011) employ a feature decay algorithm which can be used in both active learning and transductive learning settings. The decay algorithm is used to increase the variety of the training set by devaluing features that have already been seen from a training set. In recent studies, a cross-entropy difference method has seen increasing interest for the problem of SMT data selection (Moore and Lewis, 2010; Axelrod et al., 2011). The training dataset is ranked using cross-entropy difference from some language models trained on in-domain or general-domain sentences. Then a threshold is set to select the pseudo in-domain sentences. The intuition is to find sentences as close to the target domain and as far from the average of the general-domain as possible. Later, Mansour et al. (2011) argue that “An LM does not capture the connections between the source and target words, and scores the sentences independently”, and linearly interpolate IBM model 1 (Brown et al., 1993) into the cross-entropy difference framework. The translation performance is improved on both Arabic-to-English and English-to-French translation tasks compared with the standalone cross-entropy difference approach.

Applying adaptation techniques to the statistical models, especially to the translation model, is another popular approach used in domain adaptation for SMT. Some research follows the path of adding in new features into the phrase table. Chen et al. (2013) add vector similarity into the phrase table and use it as a tuning- and decoding-time feature. The similarity is computed by comparing the vectorized representation of phrase pairs extracted from the development set and the training set. Eidelman et al. (2012) achieve translation performance improvement by including a lexical weight topic feature into the translation model. The topic model used in their work is built based on the source side of the training sentences. There is also work which focuses on translation model combination. Foster and Kuhn (2007) and Koehn and Schroeder (2007) combine the translation models in a log-linear model at tuning and decoding time. Sennrich (2012) proposes an approach to interpolate the translation models based on perplexity minimization. Haddow and Koehn (2012) focus on the extracting and scoring steps when building a phrase table for SMT. One of the conclusions is that while out-of-domain data can improve the translation coverage for rare words, it may be harmful for common in-domain words. This suggests that the translations which contain a lot of in-domain evidence should be kept.

2 Related Work

The translation model fill-up approach was introduced into SMT by Nakov (2008). In his work, the phrase tables are merged by keeping all the phrase pairs unchanged from the in-domain phrase table, and only adding in the phrase pairs from the general-domain phrase tables that are not contained in the in-domain phrase table, as in (1):

\[
\text{Fill-up}(PT) = \{PT_{\text{in}}\} \cup \{PT_{\text{out}} - PT_{\text{in}}\}
\]

where \(PT_{\text{in}}\) and \(PT_{\text{out}}\) are the in-domain and general-domain phrase table, respectively, and \(\{PT_{\text{out}} - PT_{\text{in}}\}\) is the relative complement of \(PT_{\text{out}}\) in \(PT_{\text{in}}\), with the original SMT translation model features from each merging phrase tables preserved. Furthermore, a new feature
value (1 or 0.5) is allocated to each phrase pair in the merged phrase table to indicate its provenance.

Bisazza et al. (2011) modify the feature value of Nakov (2008) by interpreting it differently. A scaling factor, such as 1 \((=\exp(0))\) and 2.718 \((=\exp(1))\), is used to define the provenance of each phrase pair in the phrase table. The fill-up model (Bisazza et al., 2011) \(T_F\) is defined as in (2):

\[
\forall (\tilde{f}, \tilde{e}) \in T_1 \cup T_2 : \\
\phi(\tilde{f}, \tilde{e}) = \begin{cases} 
(\phi_1(\tilde{f}, \tilde{e}), \exp(0)) & \text{if } (\tilde{f}, \tilde{e}) \in T_1 \\
(\phi_2(\tilde{f}, \tilde{e}), \exp(1)) & \text{otherwise}
\end{cases}
\]

(2)

Bisazza et al. (2011) also extend the fill-up approach into the SMT reordering model and provide a study of pruning options. The experiments show that the fill-up approach is not only able to produce comparable translation performance with log-linear combinations of translation models, but is also an approach which increases the efficiency of minimum error rate training.

3 Probabilistic Feature-based Fill-up

In this paper, we follow the previous studies (Nakov, 2008; Bisazza et al., 2011), and propose a probabilistic feature-based translation model fill-up approach for SMT. The assumption we make for our approach is that the domain information of a training sentence pair is inheritable by the extracted phrase pairs, and such an assumption is often valid in the traditional data selection research for SMT training. Data selection is often applied when in-domain training data is small and expensive to collect, but where a large amount of general-domain training data is nonetheless available. However, Haddow and Koehn (2012) point out that it might be heavy-handed if a 1-0 cutoff is used for SMT data selection, as the general-domain data can still have a contribution to the translation system. We believe that a probabilistic feature-based fill-up approach can be factored in as a soft-handed data-selection approach. Like Bisazza et al. (2011), we extend the original fill-up algorithm (Nakov, 2008), but instead of assigning firmness provenance feature values to the phrase table, we train a machine-learning algorithm to give a probability measurement with respect to the domain information to each training sentence pair. Then we use the assumption that the domain information of a training sentence pair is inheritable by the extracted phrase pairs to make such a domain-likeness feature applicable to the phrase table. The probability scale ensures the domain-likeness feature is elastic, but also under control at tuning and decoding time.

One concern is that a phrase pair in a translation table can be extracted from a number of different training sentence pairs depending on the alignment applied and the extraction heuristic used. Accordingly, those training sentence pairs will be estimated to different domain-likeness feature values by the machine-learning algorithm used. We define the following three simple heuristics to address this issue:

- **Min**: the feature value uses the minimum domain-likeness estimations from the extracted sentence pairs. The motivation for this is if a phrase pair is extracted from a sentence pair which has a lot of evidence to be excluded from the target domain, such a phrase pair should not be classified as in-domain even if other strong in-domain indicators are present.

- **Arithmetic Mean**: use the arithmetic mean of all the domain-likeness estimations. There is no bias to any sentence pair since each will still be able to contribute the final feature value.

- **Geometric Mean**: use the geometric mean value to describe the central tendency of all domain-likeness estimations.
In the rest of this paper, we describe the machine-learning algorithm used to assign the domain-likeness value in the merged phrase table, and then we introduce the feature set used to train the said learning algorithm in Section 4. Then we describe our experiments to evaluate our probabilistic feature-based translation model fill-up approach and make comparisons with the previous fill-up studies using the basic settings\(^1\) in Section 5. Later in the paper, we make comparisons between the proposed approach with previous work on data selection (Axelrod et al., 2011) in Section 6, and provide our observations regarding the probabilistic domain-likeness feature distribution on the merged phrase table in Section 7. Finally, we give our conclusion together with avenue for future work in Section 8.

4 Support Vector Machines

4.1 SVM Algorithm

SVM is a well-known machine-learning algorithm often applied to classification or regression tasks. In classification, SVM maps a testing instance into a hyperplane which optimally separates the training data, and then outputs the predicted class label of the testing instance belongs to, the (soft margin) objective function is defined as (Cortes and Vapnik, 1995; Chang and Lin, 2011) in (3):

\[
\min_{w,b,\xi} \frac{1}{2}w^Tw + C\sum_{i=1}^{l} \xi_i \\
\text{s.t. } y_i(w^T\phi(x_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, y \in \{1, -1\}
\]

where \(w\) is the weight vector, \(C\) is a tunable trade-off parameter indicating a punishment for misclassified decisions, \(l\) is the number of training instances, \(\xi_i\) is known as the slack variable, and \(\phi\) is the kernel function mapping training instances into a high-dimensional space.

The underlying reason for using a kernel function in SVM is that the training instances in some situations are linearly non-separable and we need to improve the separability by projecting them into a high-dimensional space. In our experiments we use the Radial Basis Function (RBF) kernel for SVM training and predicting, defined as in (4):

\[
\exp\{-\gamma|u - v|^2\}
\]

The gamma parameter \(\gamma\) is a tunable variable which adjusts the width of RBF.

As SVM predicts class labels only, Chang and Lin (2011) extend the approach proposed by Wu et al. (2004) to give a probability estimation for every prediction. In our work, we use the predicted probability to indicate the domain-likeness estimation.

4.2 SVM Feature Set

It is worth recalling that our probabilistic feature-based fill-up approach is based on the assumption that the domain information of a training sentence pair can be inherited by the extracted phrase pairs, and such an assumption is often applied in SMT data selection algorithms for domain adaptation. In our case, if we are able to assign a probabilistic domain-likeness value to each training sentence, then to include them as a new decoding feature into the fill-up phrase table is effortless. Thus, we can transfer our objective into assigning the domain-likeness estimation to the SMT training sentences.

The cross-entropy, which is defined as in (5):

\[
H(p_{LM}) = -\frac{1}{n} \sum_{i=1}^{n} \log p_{LM}(w_i|w_1, \ldots, w_{i-1})
\]

\(^1\)The fill-up (Bisazza et al., 2011) provides several pruning options. There is also a cascaded fill-up method applicable for more than one general-domain phrase model. We do not make comparisons for these cases.
has been used as a strong domain indicator in much adaptation research (Klakow, 2000; Gao et al., 2002; Moore and Lewis, 2010; Axelrod et al., 2011). In equation (3), \( n \) is the number of words \( w \) in a sentence. However, in our work, we use the transformation of cross-entropy, known as perplexity, which is defined as in (6),

\[
Perplexity = 2^{H(p_{LM})}
\]  

We take inspiration from the previous works in Axelrod et al. (2011), and design three sets of SVM training features for each SMT training sentence pair

- **Source Domain Features**: the domain evidence shown from the source side of the training data. We use the perplexity value computed from the in- and general-domain language models in this feature set.

- **Target Domain Features**: the domain evidence shown from the target side of the training data. We use the perplexity value computed from the in- and general-domain language models in this feature set.

- **Domain Distance Features**: a feature set similar to the language model data-selection approach in Axelrod et al. (2011). We use both the source-side perplexity difference and the target-side perplexity difference in this feature set.

5 Experiment

5.1 Corpora

The experiments in this paper use data from WMT07, WMT13 and IWLST11 translation tasks. We choose our experiments on the French-to-English language pair. We first perform some standard data cleaning steps, including tokenization, punctuation normalization, replacement of special characters, lower casing and long sentence removal (\(<0\) or \(>80\)), resulting in the preprocessed data summarized in Table 1. We use scripts provided within Moses 1.0 translation system framework (Koehn et al., 2007)\(^2\) for all cleaning steps.

<table>
<thead>
<tr>
<th>Corpus</th>
<th>Train</th>
<th>Tune</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>News Commentary (nc_2007)</td>
<td>42,884</td>
<td>1,064 (nc-devtest200)</td>
<td>2,007 (news-test2007)</td>
</tr>
<tr>
<td>Europarl (ep_2007)</td>
<td>1,257,436</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>TED (ted_11)</td>
<td>106,642</td>
<td>934 (dev2010)</td>
<td>1,664 (tst2010)</td>
</tr>
<tr>
<td>news-commentary-v9 (nc_v9)</td>
<td>181,274</td>
<td>n/a</td>
<td>n/a</td>
</tr>
</tbody>
</table>

Table 1: SMT training corpus statistics

There are two fill-up experiments designed to evaluate our approach, defined as \(\text{prob-fill-up heuristic(in-domain,general-domain)}\), such as \(\text{prob-fill-up heuristic(nc_2007,ep_2007)}\) and \(\text{prob-fill-up heuristic(ted_11,nc_v9)}\), where \(\text{heuristic}\) refers to the heuristics stated in Section 3 of this paper. The experimental design is to assess our approach in both of the following situations: (i) general-domain dataset being significant larger than the in-domain data, and (ii) the two datasets being similar in size, as seen in Table 1.

\(^2\)http://www.statmt.org/moses/
5.2 SVM training

We use the R language package \textit{e1071} (Dimitriadou et al., 2009)\(^3\) to train the SVM algorithm, where the \textit{e1071} in R language is an interface to the libsvm (version 2.6) (Chang and Lin, 2011) implementation. SVM training is a supervised learning process so having labeled training data available is essential. The label is either in-domain or general-domain for the SVM training instance in our case.

A set of high-quality training data for tasks like classification is a luxury in machine-learning, and such datasets often cannot be obtained automatically. The in-domain labeled SVM training data can be obtained directly from the SMT training set, but the general-domain data is mixed with in- and out-of-domain instances. One solution is to rank the general-domain instances with respect to the known in-domain information, and then mark the most distant partition instances as the opposite of the in-domain class for SVM training. Such a solution can create a clear boundary in the SMT training set, but there is a danger of causing the SVM training data to be of low variance and high bias. The reason for this is that a similar amount of SVM training instances from both labeled classes are suggested to be used in order to set up a fair training condition. However, in the domain-adaptation context, where only a small amount of in-domain instances and a large amount of general-domain instances are available, we are restricted to selecting only a limited number of SVM training instances. The size limitation and the ranking selection used may lead the SVM training instances to be of low variance and high bias. In addition, we also have the prior knowledge of the predicting instances available before the SVM is trained, but it is unfortunate that such knowledge is ignored. In fact, the SVM in our case prefers to be trained on the two classes of instances that represents the average of the general-domain dataset and the in-domain dataset. Then the probability prediction produced by such an SVM can indicate the distance of a predicting instance from those two classes. Thus, we simply randomly select \(M\) number of general-domain and in-domain sentences as SVM training instances in our experiments.

To extract features for the selected SVM training data, we randomly select an equal number (size \(N\)) of sentences from the in- and the general-domain dataset and train an \(n\)-gram language model, where \(n = \{2 \ldots 5\}\), then extract the perplexity features for each \(n\) setting. The language model training at this step uses the same restrictions as in Moore and Lewis (2010), where a token is treated as an instance of \(<\text{UNK}>\) unless it appears at least twice at the in-domain training dataset. We keep \(T\) number of SVM training sentences to tune the parameters in equations (3) and (4). We test the accuracy of the trained SVM using the corresponding SMT development data. The data used for SVM training, language model training and SVM tuning are summarized in Table 2. The SVM-tuned parameters are presented in Table 3. We use the open source IRSTLM toolkit (Federico et al., 2008) for language model training and KenLM (Heafield, 2011) to compute the sentence perplexity.

<table>
<thead>
<tr>
<th>Experiment</th>
<th>(M)</th>
<th>(N)</th>
<th>(T)</th>
</tr>
</thead>
<tbody>
<tr>
<td>prob-fill-up(nc, 2007, ep, 2007)</td>
<td>42,884</td>
<td>40,000</td>
<td>2,884</td>
</tr>
<tr>
<td>prob-fill-up(usted, 11, nc, yr)</td>
<td>50,000</td>
<td>45,000</td>
<td>5,000</td>
</tr>
</tbody>
</table>

Table 2: SVM data statistics, where \(M, N\) and \(T\) are the data sizes (in sentences) used for training, tuning and testing, respectively.

\(^3\)http://www.csie.ntu.edu.tw/~cjlin/libsvm
### 5.3 Translation System Training

All SMT systems in our experiments are trained using the phrase-based SMT with Moses 1.0 framework. The reordering model is not included in our translation system since we are interested only in measuring the system effects coming from translation models. We use the word aligner MGIZA++ (Gao and Vogel, 2008) for word alignment in both translation directions, and then symmetrize the word alignment models using the heuristic of grow-diag-final-and. We use all five default Moses 1.0 translation model features. The translation systems are tuned with minimum error rate training (Och, 2003) using case-insensitive BLEU (Papineni et al., 2002) as the optimization measure. A 5-gram language model is trained with the open source IRSTLM toolkit using all the available target sentences in each of the fill-up experiment scenarios. We use the Moses default language model toolkit KenLM at the tuning and decoding time.

### 5.4 Results

We set our baseline systems to be the fill-up system of Bisazza et al. (2011) (fill-up(experiment)), which has been integrated within the Moses 1.0 framework. Tables 4 and 5 report our results using case-insensitive BLEU on the corresponding test sets. We use † to indicate where the probabilistic feature-based fill-up approach systems (prob-fill-up_heuristic(experiment)) achieve significant improvement (Koehn, 2004) compared with the baseline systems at the level $p = 0.01$ level with 1000 iterations.

![Table 3: SVM-tuned parameters values $C$ and $\gamma$, where $C$ is the trade-off parameter in equation (3), and $\gamma$ adjusts the width of RBF in equation (4).](image)

<table>
<thead>
<tr>
<th>Experiment</th>
<th>$C$</th>
<th>$\gamma$</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>prob-fill-up(nc, 2007, ep, 2007)</td>
<td>16</td>
<td>0.125</td>
<td>0.8139</td>
</tr>
<tr>
<td>prob-fill-up(ted, 11, nc, 9)</td>
<td>2</td>
<td>0.03125</td>
<td>0.8565</td>
</tr>
</tbody>
</table>

Table 4: prob-fill-up_heuristic(nc, 2007, ep, 2007) experiment BLEU scores on testing data, the significance testing at the level $p = 0.01$ level with 1000 iterations.

The result of the prob-fill-up_heuristic(nc, 2007, ep, 2007) experiment in Table 4 shows that the probabilistic feature-based fill-up systems using three heuristics for domain-likeness calculation can improve the translation performance over the baseline system. The system using the central tendency heuristic for the domain-likeness estimation outperforms the other, obtaining 0.36 absolute BLEU score and 1.3% relative improvement over the baseline system, and $p = 0.01$ significant improvement.

In our second experiment as seen in Table 5, the geometric mean calculation produces a strong BLEU score, +0.39 (1.3% relative) higher in contrast with the baseline system. However, the arithmetic mean calculation achieves the best result in this experiment with a 31.64 BLEU score (2.66% relative) on the test set. Both of the above two systems in our last experiment...
Table 5: \textit{prob-fill-up heuristic(\textit{ted\_11,nc\_v9})} experiment BLEU scores on testing data, the significance testing at the level $p = 0.01$ level with 1000 iterations.

qualify as statistically significant improvements over the baseline system at $p = 0.01$ level. The \textit{prob-fill-up Min(\textit{ted\_11,nc\_v9})} system underperforms the baseline system by about 0.1 absolute BLEU score difference.

Overall, our approach is able to significantly improve upon the baseline translation performance in both of the designed testing scenarios.

6 Data selection

In this section, we compare our probabilistic feature-based fill-up approach with the data selection approach proposed in Axelrod et al. (2011). In general, data selection is one of the standard approaches used in SMT training when out-of-domain or general-domain data is available. It is often required to train many SMT systems in order to find the most appropriate proportion of general-domain data to include and obtain the best performance from it. In this experiment, we first rank the general-domain corpus according to the sum of in- and out-of-domain perplexity difference normalized by the corresponding sentence length, defined as in (7), with the ranking in reverse order:

$$\text{PPL - DIFF} = \frac{[\text{PPL}_{\text{src}}(S) - \text{PPL}_{\text{O-src}}(S)]}{\text{length}(S)} + \frac{[\text{PPL}_{\text{tgt}}(T) - \text{PPL}_{\text{O-tgt}}(T)]}{\text{length}(T)} \tag{7}$$

where $S$ and $T$ are the source and target sentences, respectively. The language models described in Section 5.2 are used to compute perplexities. The top $p$ proportion of the ranked general-domain corpus is selected, and concatenated with the in-domain corpus. The concatenation is then used to train the data selection systems. We employ the same experimental settings described in Section 5.3 for this experiment, with the word alignments computed in advance using the combination of all in- and general-domain data. The tuning and test datasets described in Table 1 are also taken in order to compare with the experiment results described in Section 5.4.

Figures 1 and 2 illustrate the effects of the selection proportion on the BLEU score of SMT systems. As we might expect, additional general-domain training instances can benefit SMT performance, with 20\% of \textit{ep\_2007} and 65\% of \textit{nc\_v9} selection, obtaining 27.28 and 31.73 BLEU scores, respectively. In addition, it is harmful to include a large proportion of general-domain data, which can overtake the in-domain data and cause target-domain bias. In contrast, the proposed probabilistic feature-based fill-up approach is able to efficiently use all of the general-domain data, achieving significantly better translation results (Table 4) on the \textit{(nc\_2007,ep\_2007)} dataset and comparable translation results (Table 5) on the \textit{(ted\_11,nc\_v9)} dataset.
Figure 1: BLEU scores with different $p$ proportion of data selection on ($nc_{2007},ep_{2007}$) dataset.

Figure 2: BLEU scores with different $p$ proportion of data selection on ($ted_{II},nc_{v9}$) dataset.
7 Domain-likeness Distribution

In this section, we study the distribution of the domain-likeness feature added into the final merged phrase table. The main difference between our approach with the previous fill-up methods is the interpretation of the additional features employed. A learned probabilistic domain-likeness feature is used by our approach, while a binary provenance indicator is applied in previous work. It is easy to establish that the in-domain part of the produced phrase tables is identical in our and previous work, and that the total number of phrase entries is also the same. Thus, we mainly focus on the general-domain phrase entries in this section. We take the \textit{prob-fill-up heuristic}(\texttt{ted\_11,nc\_v9}) experiment in the previous section as the case study.

The \textit{prob-fill-up heuristic}(\texttt{ted\_11,nc\_v9}) experiment merges \texttt{pt(ted\_11)} and \texttt{pt(nc\_v9)} phrase tables. 5,790,068 in-domain phrase entries from \texttt{pt(ted\_11)} are kept, and 12,915,649 general-domain phrase entries from \texttt{pt(nc\_v9)} are used to fill-up. 236,779 of the phrase entries from \texttt{pt(nc\_v9)} conflict with the phrase entries in \texttt{pt(ted\_11)}, and are neglected in the final produced phrase table. The final merged phrase table contains 18,468,938 phrase entries in the \textit{prob-fill-up heuristic}(\texttt{ted\_11,nc\_v9}) experiment, where the standalone phrase table using the concatenated \texttt{ted\_11} and \texttt{nc\_v9} corpus produces 18,339,548 phrase pairs.

Table 6: Filtered \textit{prob-fill-up heuristic}(\texttt{ted\_11,nc\_v9}) phrase table entry counts with intervals of 0.05 according to SVM-assigned domain-likeness feature value.

<table>
<thead>
<tr>
<th>Interval Group</th>
<th># of phrases (Min)</th>
<th># of phrases (Arithmetic Mean)</th>
<th># of phrases (Geometric Mean)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.95 - 1.00</td>
<td>1,301,571</td>
<td>1,301,803</td>
<td>1,301,820</td>
</tr>
<tr>
<td>0.90 - 0.95</td>
<td>29,085</td>
<td>29,197</td>
<td>29,209</td>
</tr>
<tr>
<td>0.85 - 0.90</td>
<td>20,117</td>
<td>20,229</td>
<td>20,254</td>
</tr>
<tr>
<td>0.80 - 0.85</td>
<td>16,272</td>
<td>16,335</td>
<td>16,366</td>
</tr>
<tr>
<td>0.75 - 0.80</td>
<td>15,565</td>
<td>15,625</td>
<td>15,675</td>
</tr>
<tr>
<td>0.70 - 0.75</td>
<td>14,041</td>
<td>14,164</td>
<td>14,352</td>
</tr>
<tr>
<td>0.65 - 0.70</td>
<td>12,816</td>
<td>12,966</td>
<td>13,747</td>
</tr>
<tr>
<td>0.60 - 0.65</td>
<td>12,635</td>
<td>12,889</td>
<td>13,595</td>
</tr>
<tr>
<td>0.55 - 0.60</td>
<td>12,536</td>
<td>12,938</td>
<td>13,759</td>
</tr>
<tr>
<td>0.50 - 0.55</td>
<td>11,562</td>
<td>13,121</td>
<td>21,299</td>
</tr>
<tr>
<td>0.45 - 0.50</td>
<td>14,673</td>
<td>15,930</td>
<td>33,106</td>
</tr>
<tr>
<td>0.40 - 0.45</td>
<td>13,596</td>
<td>15,539</td>
<td>20,530</td>
</tr>
<tr>
<td>0.35 - 0.40</td>
<td>16,060</td>
<td>43,168</td>
<td>22,923</td>
</tr>
<tr>
<td>0.30 - 0.35</td>
<td>17,022</td>
<td>26,438</td>
<td>34,956</td>
</tr>
<tr>
<td>0.25 - 0.30</td>
<td>20,564</td>
<td>29,720</td>
<td>34,802</td>
</tr>
<tr>
<td>0.20 - 0.25</td>
<td>24,397</td>
<td>47,674</td>
<td>43,848</td>
</tr>
<tr>
<td>0.15 - 0.20</td>
<td>31,233</td>
<td>56,000</td>
<td>55,217</td>
</tr>
<tr>
<td>0.10 - 0.15</td>
<td>45,590</td>
<td>81,080</td>
<td>79,150</td>
</tr>
<tr>
<td>0.05 - 0.10</td>
<td>88,412</td>
<td>146,956</td>
<td>140,063</td>
</tr>
<tr>
<td>0.00 - 0.05</td>
<td>5,916,294</td>
<td>5,722,269</td>
<td>5,709,370</td>
</tr>
</tbody>
</table>

To demonstrate the distribution of the phrase pairs in the merged phrase table, we first group the phrase entries in the merged phrase tables (filtered using the corresponding test set) with intervals of 0.05 according to the domain-likeness feature value. We can observe in Table 6 that the SVM predictions fall mostly into the 0.00 - 0.05 or 0.95 - 1 intervals. We think that the prediction follows the natural composition of the general-domain dataset, so the composition can be described as consisting of some of the target unrelated sentences, some of the
mixed domain sentences and some of the in-domain sentences. The range between 0.05 \sim 0.95 also draws our attention. All three heuristic functions create similar numbers of phrase entries for each interval group at the upper bound range: 0.70 \sim 1.00. This may be evidence that there is only 0.92 BLEU score difference between the best- and worst-performed probabilistic feature-based fill-up systems in Table 5 since the upper bound range is the closest to the target translation domain. Later, the \textit{Geometric\_Mean} system acts more aggressively and there is a dramatic increase in the quality of phrase pairs at the intervals of 0.45 \sim 0.50. We think that this interval is the most uncertain region in the general-domain dataset given the knowledge inferred by the corresponding heuristic functions. A similar increase also can be found in the \textit{Arithmetic\_Mean} system at the intervals of 0.35 \sim 0.40, but the increasing curve is sharper compared with the growth in \textit{Geometric\_Mean}. The lower bound range in Table 6 is in a very mixed situation.

The graph in Figure 3 compares for the interval grouped range between 0.10 to 1.00, the percentage of phrase entries contributing to the overall phrase table. It shows that the general-domain training sentences can provide different levels of utility, and can be beneficial (in the case of probability feature value >0.5) or harmful (in the case of probability feature value <0.5) to the merged phrase table. Haddow and Koehn (2012) also found that general-domain training data can benefit the translation table most when it is just allowed to add entries, but also that the scores from the general-domain may be harmful to translation quality. Previous work tries to address this question by defining a fairness feature value to all phrase pairs extracted from the general-domain training sentences. However, such a fairness feature value may cause the potential in-domain phrase entries to be treated unjustly. Using a probabilistic feature value representing domain-likeness can distinguish between the extracted phrase pairs and also provides a soft-handed approach for phrase-table merging.

Figure 3: The distribution of \textit{Min}, \textit{Arithmetic\_Mean} and \textit{Geometric\_Mean} phrase pairs contribution comparison: X-axis represents the range from 0.10 to 1.00. Y-axis represents the percentage of phrase entries to the overall testing data filtered phrase table.
8 Conclusion and Future Work

In this paper, we addressed the inaccurate assumption introduced at the phrase extraction step for phrase-based SMT training. We extended the fill-up phrase-table merging approach by assigning a domain-likeness probabilistic feature. We described the rationale behind our probabilistic feature-based fill-up approach and explained our intuitions regarding the SVM feature set. We also designed two experimental scenarios, showing that our fill-up approach is a soft-handed dynamic approach and can significantly improve translation performance in both experiments compared to previous fill-up studies. However, the approach shown in this paper is still preliminary and can be extended further. We have not carried out experiments regarding any implication between the SVM performance and the SMT translation performance; our SVM features are purely inspired by the previous data selection studies and can also be more elegant. In future work, we would like to carry out such studies. We would also like to experiment on a reordering model fill-up and introduce more domain-oriented SVM training features. The proposed probabilistic feature-based fill-up approach can also be viewed as a domain adaptation approach, where bilingual in-domain training sentences are unavailable, but where a large amount of general-domain bilingual training sentences is easy to obtain. We can train the SVM algorithm to assign the domain-likeness feature using the source and the target monolingual in- and general-domain data to the general-domain only phrase table. Thus the general-domain-only phrase table can gain some domain knowledge at decoding time.

9 Acknowledgements

This research is supported by the Science Foundation Ireland (Grant 12/CE/I2267) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Dublin City University.

References


Document-level Re-ranking with Soft Lexical and Semantic Features for Statistical Machine Translation

Chenchen Ding  
tei@mibel.cs.tsukuba.ac.jp  
Department of Computer Science, University of Tsukuba  
1-1-1 Tennodai, Tsukuba, Ibaraki, 305-8573, Japan

Masao Utiyama  
mutiyama@nict.go.jp  
National Institute of Information and Communications Technology  
3-5 Hikaridai, Seikacho, Sorakugun, Kyoto, 619-0289, Japan

Eiichiro Sumita  
eiichiro.sumita@nict.go.jp

Abstract

We introduce two document-level features to polish baseline sentence-level translations generated by a state-of-the-art statistical machine translation (SMT) system. One feature uses the word-embedding technique to model the relation between a sentence and its context on the target side; the other feature is a crisp document-level token-type ratio of target-side translations for source-side words to model the lexical consistency in translation. The weights of introduced features are tuned to optimize the sentence- and document-level metrics simultaneously on the basis of Pareto optimality. Experimental results on two different schemes with different corpora illustrate that the proposed approach can efficiently and stably integrate document-level information into a sentence-level SMT system. The best improvements were approximately 0.5 BLEU on test sets with statistical significance.

1 Introduction

State-of-the-art statistical machine translation (SMT) systems (Koehn et al., 2007) have achieved good performance for many translations, such as French-to-English translation. The success can be attributed to the statistical model used in translation and the huge data for model training. However, the well-developed techniques of SMT are mainly focused on the sentence-level translation, i.e., the models are trained on the parallel corpus of sentence pairs, and the translation is conducted sentence-by-sentence. Because in practice sentences are usually contained in a document and surrounded by context, recent research has begun to focus on enhancing SMT systems with the addition of document-level information.

As to the features of document-level translation, a frequently discussed issue is lexical consistency in translation: i.e., words tend to be translated consistently in a document (Carpuat, 2009; Carpuat and Simard, 2012). There are also detailed discussions around the consistency of different parts of speech (Guillou, 2013; Meyer and Webber, 2013). On the basis of lexical consistency theory, many researchers focus on increasing the lexical consistency in translation (Tiedemann, 2010; Xiao et al., 2011; Ture et al., 2012). Beyond lexical consistency, there are attempts at using lexical cohesion in translation (Ben et al., 2013; Xiong et al., 2013a,b), which considers the semantic relation between words. Rather than the lexical features, the topic of
documents is also taken as a feature in some recent research (Gong et al., 2011; Eidelman et al., 2012; Xiong and Zhang, 2013; Hasler et al., 2014).

Among the different features, lexical consistency is the simplest feature because it only considers the lexical words themselves. In contrast, lexical cohesion involves more semantic information, such as hypernyms and hyponyms, usually requiring a word-net. Approaches that use the document topic as a feature usually require training data, such as a document-level parallel corpus, in the training or decoding phases.

In this paper, we propose an approach that considers both the lexical consistency and semantic relation on the document level. The approach first uses an off-the-shelf SMT system to conduct the sentence-level translation, where both the training and decoding are on the sentence level. Then we introduce two document-level features, one using the word-embedding technique to model the semantic relation of context on the target side and the other using a token-type ratio to model the consistency in translation. With the two document-level features, we conduct a further decoding on the document level to get a better combination of sentence-level translation within a document. As to the weights of the introduced features, we utilize a multi-objective learning approach based on the Pareto optimality (Duh et al., 2012) to simultaneously optimize the sentence-level and document-level metrics. The proposed approach requires no word-net or document-level parallel corpus for model training. Instead, it requires a vector list of the target-side vocabulary by word embedding and a small development set of parallel document pairs to tune the weights of document-level features.

The remainder of the paper is organized as follows. In Section 2, we mention the related work around using document-level information in translation. In Section 3, we describe our proposed approach. Section 4 presents experimental results, and Section 5 is the discussion, where we compare the proposed approach with a consistency verification approach (Xiao et al., 2011). Section 6 contains the conclusions and future work.

2 Related Work

For the approaches focusing on the lexical consistency, an early attempt is the work of Tiedemann (2010), where decaying cache models for both language and translation models are used for SMT. The cache models give the SMT system a preference for recently used words and translation rules. The approach succeeded for an out-of-domain test set but failed for an in-domain test set. Tiedemann (2010) mentions that the cache model may be “risky”. In Xiao et al. (2011), a re-decoding approach for a baseline SMT system is proposed to ensure lexical consistency in translation, with quite detailed manual analysis of the experiment results. The approach also has an improved BLEU score, which the authors mention as a bonus. Ture et al. (2012) used a force-decoding approach for an SCFG-based translation system with several Okapi BM25 term weights. These works are based on the “one translation per corpus” constraint discussed in Carpuat (2009). On the other hand, the report in Carpuat and Simard (2012) asserted “SMT translates document remarkably consistently, even without document knowledges.” In our opinion, this is a complex issue that may depend on the data used or even the language pair in the translation task.

As to the approaches using lexical cohesion, Ben et al. (2013) and Xiong et al. (2013a) use semantic relations in a word-net to identify bilingual hypernym and hyponym relations in translation. In Xiong et al. (2013b), a thesaurus is used to construct the source-side lexical chain. These approaches step forward into the field of the semantic; thus, they require the help of particular linguistic resources.

Document-level topic-based approaches also exist (Gong et al., 2011; Eidelman et al., 2012; Xiong and Zhang, 2013; Hasler et al., 2014), which introduce extra topic models into the translation process to improve the word selection for specific topics. Usually, the topic model
is statistical and needs to be trained on monolingual or bilingual document-level data. Along with the feature of lexical cohesion, the topic is a sophisticated feature that must be supported by extra resources.

Many approaches using document-level features require to modify the decoder of a baseline system to adapt to their features in decoding. Research mainly focusing on the decoding and tuning algorithm, such as the series work of Hardmeier et al. (2012) and Stymne et al. (2013), extends the traditional sentence-based SMT system to be able to collaborate with document-level features.

As to our approach, the features used can model the lexical consistency as well as semantic relation at a certain level while not being as rigid as the features/operations on the very lexical level that many previous approaches use. We assume that these features, combined with the multi-objective tuning, will provide a robust and stable way to take advantage of document-level information in an SMT system.

3 Proposed Approach

3.1 Overview

The proposed approach is essentially a re-ranking process in a document-level decoding (Fig. 1). We first use an off-the-shelf baseline SMT system to translate a document sentence-by-sentence, obtaining the m-best translation candidates for each sentence. The baseline SMT system can be trained and tuned in a standard way with sentence-level parallel data. Then, we conduct a decoding on the document level to find good combinations among the m-best candidate sentences. The search is realized in a cube-pruning way (Chiang, 2007). Here, we use good to mean that the combinations are good for both sentence- and document-level metrics under the Pareto optimality (Duh et al., 2012). As far as we know, this is the first attempt to apply document-level re-ranking in an SMT system.
3.2 Notation
In the following description, we use \( D \) to denote an input document on the source-side language composed of \( n \) sentences, which are \( \{s_1, s_2, \ldots, s_n\} \). The reference translation of \( D \) on the target-side language is denoted by \( D^r \), composed of \( \{r_1, r_2, \ldots, r_n\} \), where sentence \( r_k \) (1 \( \leq k \leq n \)) is the reference translation of \( s_k \). Each sentence \( s_k \) (1 \( \leq k \leq n \)) in \( D \) has an \( m \)-best translation candidate composed of \( \{t_{k1}^1, t_{k1}^2, \ldots, t_{km}^k\} \). Over the total \( nm \)-best candidate lists, we search for a combination \( C = \{t_{c11}, t_{c22}, \ldots, t_{cnn}\} \) (1 \( \leq c_k \leq m, 1 \leq k \leq n \)) for optimization.

Generally, a test set contains multiple \( l \)-documents of \( \{D_1, D_2, \ldots, D_l\} \); hence, correspondingly we search \( \{C_1, C_2, \ldots, C_l\} \).

3.3 Optimization Function
For optimization, we use two objective metrics: a sentence-level metric (\( S_{\text{eval}} \)) and a document-level metric (\( D_{\text{eval}} \)), as follows.

\[
S_{\text{eval}}(\{C_1, \ldots, C_l\}, \{D_{1r}^1, \ldots, D_{lr}^l\}) = \text{BLEU}(\{C_1, \ldots, C_l\}, \{D_{1r}^1, \ldots, D_{lr}^l\}) \tag{1}
\]

Using the BLEU score for the \( S_{\text{eval}} \), a candidate translation will be evaluated with its reference translation, sentence by sentence, disregarding document-level information (Fig. 2).

\[
D_{\text{eval}}(\{C_1, \ldots, C_l\}, \{D_{1r}^1, \ldots, D_{lr}^l\}) = \frac{\text{avg}}{1 \leq i \leq l} \{\text{avg diff (context of } t_{ki} \in C_i, r_k \in D_{ir}^i) \} \tag{2}
\]

For the \( D_{\text{eval}} \), we evaluate the context of a candidate translation with its reference translation (Fig. 3). In Exp. (2), \text{avg} represents average and the function \text{diff} represents the difference between sentences. We will mention the details of the function \text{diff} and the context in the description of introduced features’ calculation, because they are essentially identical.

Because the context is composed of candidate translation of other sentences within a document, any candidate translation will be evaluated according to two aspects: the similarity between its own reference and itself (\( S_{\text{eval}} \)), and the relation with the other references where it becomes a context (\( D_{\text{eval}} \)). The former evaluation is preformed in a strict n-gram matching method to control the local translation of every word, whereas the latter is quite sketchy to reveal more sentence cross-relation in the document.

3.4 Document-Level Features
As a baseline, SMT system has already been tuned to optimize the \( S_{\text{eval}} \) (i.e., BLEU), we introduce two features, \( f_{\text{doc}}^i \) and \( f_{\text{doc}}^{st} \), to represent the performance against \( D_{\text{eval}} \) as follows.
Figure 3: In \( D_{eval} \), every translation \( t \) will be compared with reference \( r \) of the other translations, where \( t \) becomes context within a document. Here, the context of \( t_k \) is \( t_{k-1} \) and \( t_{k+1} \).

\[
f_{doc}^k(C) = \text{avg}_{1 \leq k \leq \text{length}(C)} \text{diff}(\text{context of } t_k \in C, t_k \in C)
\]

Exp. (3) is similar to Exp. (2), with the \( r_k \in D_r \) substituted for \( t_k \in C \) (i.e., in Fig. 3, the lower rank and the upper rank are identical). The feature \( f_{doc}^k \) reveals the difference of a candidate translation with its context, which is also composed of candidate translations. Specifically, we take the context of \( t_k \) as:

\[
\{t_{\max(0,k-x)}, \cdots t_{k-1}, t_{k+1}, \cdots t_{\min(\text{length}(C),k+x)}\}
\]

Here \( x \) is a window size. Further, for the \( \text{diff}(\cdot, \cdot) \) function, we want it to be flexible to reveal more sentence cross-relation as the strict lexical-based evaluation will be controlled by the \( S_{eval} \). Therefore, we use the word-embedding technique to transform the lexical information into vector representation and use the distance between vectors as the \( \text{diff} \) function. Specifically, we define the \( \text{diff}(\cdot, \cdot) \) as:

\[
\text{diff}(\mu, \nu) = \log \|\mu - \nu\|
\]

where \( \mu \) and \( \nu \) are two vectors and \( \|\cdot\| \) is the Euclidean norm. To get the vector of a sentence or a set of sentences, we use the bag-of-word approach to get the average vector of all the word vectors contained by the sentence(s).

The \( f_{doc}^k \) feature concerns only to the target-side translation candidates. If candidate translations are similar to their context on average in a document, the feature will be small and if they are not, the feature will be large. On the other hand, we also need a feature to reveal the consistency in translation that can connect the source side and target side. So we use the feature \( f_{doc}^{st} \) to reveal the consistency in translation.

\[
f_{doc}^{st}(D, C) = \text{avg}_v \left\{ \log \frac{\sum_{w \in \{v, w \} \in (D, C)} \text{count}(v, w)}{|\{w \} \in \{v, w \} \in (D, C)}\right\}
\]

Here, \( v \) is a word on source side and \( w \) is a word on target-side; \( (v, w) \) is a translated word pair and \( \text{count}(\cdot) \) is a count function. Exp. (6) is essentially an average of token-type log-ratio over a source-side word \( v \) (Fig. 4): i.e., for \( v \), we count the total times it has been translated by \( \sum_w \text{count}(v, w) \) and count how many types of target-side words it has been translated to by \( |\{w \} \in \{v, w \} \in (D, C)}\). If source-side words are consistently translated to one or a few certain target-side words on average, the feature will grow large; if not, the feature will be small.

Besides the two document-level features we introduced, we also take the score generated by the baseline SMT system as a sentence-level feature \( f_{snt} \). Then we use an interpolation of
the features as the score for a \( \{C_1, C_2, \cdots, C_l\} \) search as follows.

\[
\text{score}(\{D_1, \cdots, D_l\}, \{C_1, \cdots, C_l\}) = \\
\sum_{1 \leq i \leq l} \{\lambda_{\text{snt}} f_{\text{snt}}(D_i, C_i) + \lambda_{\text{doc}}^{f_{\text{doc}}} f_{\text{doc}}^{f_{\text{doc}}}(D_i, C_i) + \lambda_{\text{doc}}^{f_{\text{st}}} f_{\text{st}}^{f_{\text{st}}}(D_i, C_i)\} \\
(|\lambda_{\text{snt}}| + |\lambda_{\text{doc}}^{f_{\text{doc}}}| + |\lambda_{\text{doc}}^{f_{\text{st}}}| = 1)
\]

(7)

3.5 Decoding and Tuning

The algorithm we used in decoding is basically a cube-pruning algorithm (Chiang, 2007) to merge the \( m \)-best list of translation candidates together over an entire document. Within the process of merging, the \( f_{\text{doc}}^{f_{\text{doc}}} \) and \( f_{\text{st}}^{f_{\text{st}}} \) are calculated. The merging needs to be conducted on the entire document because the \( f_{\text{st}}^{f_{\text{st}}} \) can only be calculated for a given combination of sentence candidates over the entire document.\(^1\)

Because the number of lists is equal to the number of sentences in a document, which usually becomes several tens or over a hundred, the original cube-pruning approach will not work well because its steps only forward to the next one candidate in each list from the present frontier, which prevents the search from generating enough combinations when there are too many lists. For example, consider a case in which we search 100 different combinations of sentence candidates over a document composed of 100 sentences; on average, we only touch the 2-best candidate (the one immediately below the top one) of each sentence. To avoid the problem, we use a wider margin \( B \) for each list in search rather than only +1 in the original algorithm.\(^2\) The time complexity of the search for a document will be \( O(N^2BT) \), where \( N \) is the number of sentences in a document; \( B \) is the width of the margin; and the \( T \) is the search times. For a document with \( N \) sentences, in each search, \( N \cdot B \) combinations will be generated for \( f_{\text{doc}}^{f_{\text{doc}}} \) and \( f_{\text{doc}}^{f_{\text{doc}}} \) calculation. The two feature calculations are linear to the number of sentences in a document, i.e. \( O(N) \). Thus, we have the described time complexity.

We apply the decoding algorithm on a development set of document pairs to tune the weights \( \lambda_{\text{snt}}, \lambda_{\text{doc}}^{f_{\text{doc}}}, \) and \( \lambda_{\text{doc}}^{f_{\text{st}}} \). According to Duh et al. (2012), the tuning algorithm is a multi-objective learning algorithm under the Pareto optimality. The method of Duh et al. (2012) is originally used for simultaneously tuning parameter weights to optimize different sentence-level translation measures. It has been shown that multi-objective tuning shows more robustness

---

\(^1\) Note that we can set a window size for \( f_{\text{doc}}^{f_{\text{doc}}} \)

\(^2\) Specifically, the change is regarding line 11 of Fig. 6 in Chiang (2007). This line is executed multiple times in our search, with a more large enumerating margin for each list.
than traditional single-objective tuning. In our approach, we tune the weights under the Pareto optimality of \( \{ S_{\text{eval}}, D_{\text{eval}} \} \) as follows:

\[
\begin{align*}
\text{argmax}_{\lambda_{\text{snt}}, \lambda_{\text{doc}}, \lambda'_{\text{doc}}} \{ S_{\text{eval}}, +D_{\text{eval}} \} & \quad (\lambda_{\text{doc}} > 0) \\
\text{argmax}_{\lambda_{\text{snt}}, \lambda_{\text{doc}}, \lambda'_{\text{doc}}} \{ S_{\text{eval}}, -D_{\text{eval}} \} & \quad (\lambda_{\text{doc}} < 0)
\end{align*}
\]

We maximize the \( S_{\text{eval}} \) (BLEU) because it is a measure for which the higher is the better. However, we are not sure in the case of \( D_{\text{eval}} \). As mentioned, the \( D_{\text{eval}} \) and feature \( f_{\text{doc}} \) essentially have the same interpretation, so we make the sign of \( D_{\text{eval}} \) dependent on the sign of \( \lambda_{\text{doc}} \), to make the optimization meaningful. When \( \lambda_{\text{doc}} > 0 \), i.e., the distance between a sentence and its context is to be encouraged, we maximize the \( D_{\text{eval}} \); if the opposite, we minimize the \( D_{\text{eval}} \) (i.e., maximize the \(-D_{\text{eval}}\)).

The multi-objective tuning will generate a Pareto frontier of multiple sets of weights rather than a single deterministic weight setting. The difference between the linear combination and Pareto optimality in multi-objective tuning has been discussed and compared in Duh et al. (2012). Generally, the Pareto optimality strategy is to optimize first agnostically and a posteriori let the designer choose among a set of weights. This philosophy is also reasonable in our approach, which is a post-process applied in a baseline SMT system to introduce document-level information. In practice, if document-level information is no available, our approach degenerates to the baseline system (i.e., \( \lambda_{\text{doc}} = 0 \)); otherwise, the approach produces several sets of \( \{ \lambda_{\text{doc}}, \lambda'_{\text{doc}} \} \), which suggests that we should pay attention to the document-level features.

4 Experiment

4.1 Data and Settings

We tested the proposed approach on French-to-English translation because this translation task has been handled well by state-of-the-art SMT systems. We used two different schemes. One is on the WIT\(^3\) corpus of TED talks\(^4\) (Cettolo et al., 2012), which contains a small training set with document-level parallel development set and test set. The other scheme is a relatively more realistic setting: using the Europarl corpus (Koehn, 2005) for model training and an in-domain development set for the weight tuning in the baseline SMT system. Then we selected document pairs from the Common Crawl (CC) Corpus\(^5\) of WMT2013 for document-level development and test set. The CC corpus has a lot of noise, with many document pairs only several sentences long – too short for our purposes. Thus, we selected relatively high-quality document pairs, with moderate lengths of 40–60 sentences to compose the baseline. The data used in the two schemes and the detailed information are listed in Tables 1 and 2, respectively.

In experiments, as previously described, a baseline SMT system was built from sentence-level parallel data (the \( \text{train} \) row in Tables 1 and 2) and tuned on sentence-level development set (the \( \text{dev.} \ (\text{snt.}) \) row). We used the phrase-based statistical machine translation (PB SMT) system of Moses\(^6\) (Koehn et al., 2007) as the baseline SMT system. In model training, we used the \( \text{grow-diag-final-and} \) to symmetrize the output of GIZA++\(^7\) (Och and Ney, 2003). The \( \text{max-phrase-length} \) was set to 7 and the reordering model was \( \text{msd-bidirectional-fe} \). The language

---

3We always set \( \lambda_{\text{snt}} \) to be positive. \( \lambda_{\text{doc}} \) and \( \lambda'_{\text{doc}} \) can be either positive or negative.

4https://wit3.fbk.eu/

5http://www.statmt.org/wmt13/translation-task.html

6http://www.statmt.org/moses/

7https://code.google.com/p/giza-pp/
Table 1: Data used in experiment.

<table>
<thead>
<tr>
<th>scheme-1</th>
<th>scheme-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>train WIT</td>
<td>Europarl</td>
</tr>
<tr>
<td>dev. (snt.) WIT</td>
<td>WMT dev2006</td>
</tr>
<tr>
<td>dev. (doc.) WIT</td>
<td>CC</td>
</tr>
<tr>
<td>test WIT</td>
<td>CC</td>
</tr>
</tbody>
</table>

Table 2: Number of sentence and document pairs of corpora. The dev. (snt.) and dev. (doc) of scheme-1 are an identical set.

<table>
<thead>
<tr>
<th>scheme-1</th>
<th>scheme-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>train 0.14M snt.</td>
<td>1.99M snt.</td>
</tr>
<tr>
<td>dev. (snt.) 934 snt.</td>
<td>2,000 snt.</td>
</tr>
<tr>
<td>dev. (doc.) 8 doc. / 934 snt.</td>
<td>14 doc. / 600 snt.</td>
</tr>
<tr>
<td>test 11 doc. / 1,664 snt.</td>
<td>55 doc. / 2,500 snt.</td>
</tr>
</tbody>
</table>

model was an interpolated 5-gram model with modified Kneser-Ney discounting, trained by SRILM\(^8\) (Stolcke, 2002), on each scheme’s training data. In sentence-level decoding, the \textit{table-limit} was 20; the \textit{stack size} was 200; and the \textit{distortion-limit} was 6, all of which followed the default settings of Moses’ decoder. The feature weights of the baseline PB SMT system were tuned by MERT (Och, 2003) to optimize the sentence-level development set BLEU (Papineni et al., 2002). The settings in tuning and translating on sentence-level were identical.

For the document-level decoding of the proposed approach, we used the baseline system to generate a 1000-best translation candidate list for each sentence in a document. Each translation candidate was attached with the word alignment information in sentence-level decoding for the \(f_{doc}^t\) calculation. Duplicate candidates in a 1000-best list were merged to one candidate taking the highest score\(^9\) of the baseline SMT system. For the \(f_{doc}^t\) calculation, we used a high-quality English word embedding used in the SENNA\(^{10}\) toolkit (Collobert et al., 2011).\(^11\) The word embedding is over a vocabulary of 130,000 words, with 50-dimension vectors.

In the document-level decoding algorithm, we set the \textit{margin} in cube-pruning to \([-10, 10]\) to enlarge the search space. The search generated 100 document-level candidates for re-ranking. In the \(f_{doc}^t\) feature and the \(D_{eval}\) calculation, we set \textit{window-size} to 2. That is, the context was defined as the two preceding and two succeeding sentences.

For weight tuning on the document level, the multi-objective tuning can be combined with any tuning algorithm, such as MERT (Och, 2003), MIRA (Chiang, 2012), or PRO (Hopkins and May, 2011). Our approach contains only two free weights, \(\lambda_{doc}^t\) and \(\lambda_{doc}^t\); thus, we used a \textit{greedy search} for them in \((-1.0, 1.0)\), with step of 0.1, to avoid any possible search errors in the tuning phase.

We took the \textit{consistency verification} approach (Xiao et al., 2011) as the comparison approach in our experiments. Similar to our approach, this approach takes advantage of the \textit{m}-best translation candidates and uses a further decoding step to polish the baseline sentence-level

\(^8\)http://www.speech.sri.com/projects/srilm/

\(^9\)As well as the word alignment of the highest-scoring candidate.

\(^10\)http://ml.nec-labs.com/senna/

\(^11\)We also tried other vectors of word embedding, such as using \texttt{word2vec} (https://code.google.com/p(word2vec/) or \texttt{nplm} (http://nlg.isi.edu/software/nplm/) to train vectors on a data dump of Wikipedia. However, different vectors did not affect the performance much so we just used the pre-trained vectors of SENNA.
translation (the re-decoding in Xiao et al. (2011)). Specifically, the approach first generates a list of ambiguous words on the source side. Then it collects possible translations of those ambiguous words from \( m \)-best translation candidates and selects one standard translation for each ambiguous word. Finally, the translation model (i.e. phrase table) was filtered to ensure that it contains only the standard translation for ambiguous words. With the filtered translation model, a re-decoding is conducted.

In our experiment, we followed the instructions of Xiao et al. (2011), using 5-best list, a scaling factor \( \alpha \) of 0.01,\(^{12}\) and the M1 method, which leads to a better performance. A problem is that experiments conducted in Xiao et al. (2011) were on corpora of news, and they used a term database to select the source-side ambiguous word. Because we do not have such a resource and our experimental schemes have more variations, we selected the source-side ambiguous word by \( tf-idf \) score and took the top-\( k \) \( tf-idf \) words. We varied the \( k \) in the experiment.\(^{13}\)

### 4.2 Results

In Table 3, we show the test set BLEU of the baseline SMT system and the effect of the consistency verification method. For scheme-1, the baseline achieve a test set BLEU of over 30, despite the scanty training data. In Cettolo et al. (2012), the performance on the same dataset of English-to-French is reported, which also had a test set BLEU of over 30. Because French and English have relatively similar vocabulary and syntax, we consider the baseline of scheme-1 reasonable. For scheme-2, the baseline’s test set BLEU is also near to 30, as we intend to build a high baseline.\(^{14}\) When we test the consistency verification method, we observe that it works on scheme-1 but not on scheme-2, and the performance worsens as when the number of verified words increases. We attribute the phenomenon to the rigidity of the consistency verification method. As mentioned, the data used in Xiao et al. (2011) are bound to the news field. Although the topics vary among the documents, a substantial consistency in special-term translation is required in the news field, and Xiao et al. (2011) did use a database of terms. However, the textual data used in our experiment are more casual and variable, especially in scheme-2. Consequently, the consistency verification method does not perform well in scheme-2.

In Tables 4 and 5, we show the experimental results of the proposed approach in scheme-1 and scheme-2, respectively. Different sets of weights on the frontier of Pareto optimality are listed,\(^{15}\) with their corresponding \( S_{eval} \) and \( D_{eval} \) on development set and \( S_{eval} \) on test set (i.e., test set BLEU). The first rows, \( \lambda_{doc} = 0 \) and \( \lambda_{doc} = 0 \) are the performance of the baseline SMT system for scheme-1 and scheme-2. We conduct a statistical significance test via the bootstrap method (Koehn, 2004) using bleu-kit\(^{16}\). For each row, + and − mean the result is better or worse than the baseline, respectively: a single mark means the difference is on the level of \( p < 0.05 \) and a double mark means on the \( p < 0.01 \) level. For the overall performance, in scheme-1, the change of test set BLEU is in the range of \([-0.01, +0.48]\) points compared to the baseline; in scheme-2, the range of change is in \([-0.26, +0.56]\). Because the Pareto frontier offers multiple weights rather than a deterministic, the change on test set BLEU we report here is a range rather than a deterministic value.

---

\(^{12}\)Xiao et al. (2011) said a proper \( \alpha \) is in \([0.005, 0.1]\).

\(^{13}\)The \( k \) is used to generate a list of source-side words for verification. The list also contains unambiguous words; hence, the types of verified source-side words are less than \( k \) in our experiment.

\(^{14}\)The test set used in Hasler et al. (2014), which is chosen from the same CC data, has a baseline test set BLEU near 20.

\(^{15}\)In search, we filter out the weights that noticeably worsen the development set \( S_{eval} \) (BLEU), which are worse than −0.5 point than the baseline.

\(^{16}\)http://www.nlp.mibel.cs.tsukuba.ac.jp/bleu_kit/
Table 3: Test set BLEU of baseline SMT system and consistency verification (Xiao et al., 2011) of $k = 10, 20, .50$.

<table>
<thead>
<tr>
<th></th>
<th>scheme-1</th>
<th>scheme-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>31.10</td>
<td>29.39</td>
</tr>
<tr>
<td>top-10</td>
<td>31.17</td>
<td>28.46</td>
</tr>
<tr>
<td>top-20</td>
<td>31.08</td>
<td>28.21</td>
</tr>
<tr>
<td>top-50</td>
<td>30.22</td>
<td>27.13</td>
</tr>
</tbody>
</table>

Table 4: $S_{eval}$ and $D_{eval}$ on development set and $S_{eval}$ on test set (test set BLEU) in scheme-1. Different $\lambda^t_{doc}$ and $\lambda^s_{doc}$ are generated by multi-objective tuning.

<table>
<thead>
<tr>
<th>$\lambda^t_{doc}$</th>
<th>$\lambda^s_{doc}$</th>
<th>dev. $S_{eval}$</th>
<th>dev. $D_{eval}$</th>
<th>test BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>28.09</td>
<td>.2223</td>
<td>31.10</td>
</tr>
<tr>
<td>+0.2</td>
<td>-0.2</td>
<td>27.76</td>
<td>.2238</td>
<td>31.50++</td>
</tr>
<tr>
<td>+0.2</td>
<td>-0.5</td>
<td>27.70</td>
<td>.2242</td>
<td>31.54++</td>
</tr>
<tr>
<td>+0.3</td>
<td>-0.3</td>
<td>27.69</td>
<td>.2243</td>
<td>31.50++</td>
</tr>
<tr>
<td>+0.3</td>
<td>-0.4</td>
<td>27.64</td>
<td>.2250</td>
<td>31.58++</td>
</tr>
<tr>
<td>+0.4</td>
<td>0.0</td>
<td>27.84</td>
<td>.2238</td>
<td>31.34++</td>
</tr>
<tr>
<td>+0.4</td>
<td>-0.1</td>
<td>27.70</td>
<td>.2242</td>
<td>31.41++</td>
</tr>
<tr>
<td>+0.5</td>
<td>-0.1</td>
<td>27.60</td>
<td>.2255</td>
<td>31.51++</td>
</tr>
</tbody>
</table>

Table 5: $S_{eval}$ and $D_{eval}$ on development set and $S_{eval}$ on test set (test set BLEU) in scheme-2. Different $\lambda^t_{doc}$ and $\lambda^s_{doc}$ are generated by multi-objective tuning.

<table>
<thead>
<tr>
<th>$\lambda^t_{doc}$</th>
<th>$\lambda^s_{doc}$</th>
<th>dev. $S_{eval}$</th>
<th>dev. $D_{eval}$</th>
<th>test BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>28.34</td>
<td>.1583</td>
<td>29.39</td>
</tr>
<tr>
<td>-0.1</td>
<td>-0.8</td>
<td>28.79</td>
<td>.1566</td>
<td>29.95++</td>
</tr>
<tr>
<td>-0.2</td>
<td>-0.6</td>
<td>28.73</td>
<td>.1565</td>
<td>29.89++</td>
</tr>
<tr>
<td>-0.3</td>
<td>-0.5</td>
<td>28.62</td>
<td>.1562</td>
<td>29.85++</td>
</tr>
<tr>
<td>-0.3</td>
<td>-0.6</td>
<td>28.38</td>
<td>.1549</td>
<td>29.71++</td>
</tr>
<tr>
<td>-0.4</td>
<td>-0.4</td>
<td>28.51</td>
<td>.1557</td>
<td>29.80++</td>
</tr>
<tr>
<td>-0.5</td>
<td>-0.4</td>
<td>28.34</td>
<td>.1539</td>
<td>29.45</td>
</tr>
<tr>
<td>-0.6</td>
<td>-0.1</td>
<td>28.44</td>
<td>.1556</td>
<td>29.76++</td>
</tr>
<tr>
<td>-0.6</td>
<td>-0.3</td>
<td>28.28</td>
<td>.1534</td>
<td>29.39</td>
</tr>
<tr>
<td>-0.7</td>
<td>0.0</td>
<td>28.43</td>
<td>.1553</td>
<td>29.67++</td>
</tr>
<tr>
<td>-0.7</td>
<td>-0.2</td>
<td>28.19</td>
<td>.1534</td>
<td>29.24</td>
</tr>
<tr>
<td>-0.9</td>
<td>0.0</td>
<td>28.11</td>
<td>.1532</td>
<td>29.13−</td>
</tr>
</tbody>
</table>

5 Discussion

From the experimental results, we observe that the proposed approach can generate better results by introducing document-level features on different datasets. In Table 4, we observe that the development set $S_{eval}$ in scheme-1 is actually decreased by different weights while the test set BLEU increases. This is because we use an identical development set for sentence-level tuning in the baseline SMT system and for the purposes of document-level tuning. Apparently, the tuning in the baseline system tends to over-fit the development set, and the proposed approach
Table 6: Change on the test set BLEU of sentence-level (only changed translations are counted).

<table>
<thead>
<tr>
<th></th>
<th>unchanged</th>
<th>increased</th>
<th>decreased</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheme-1</td>
<td>13.58%</td>
<td>12.32%</td>
<td>9.13%</td>
<td>35.03%</td>
</tr>
<tr>
<td>scheme-2</td>
<td>9.48%</td>
<td>13.16%</td>
<td>7.24%</td>
<td>29.88%</td>
</tr>
</tbody>
</table>

Table 7: Change on the test set BLEU of document-level.

<table>
<thead>
<tr>
<th></th>
<th>unchanged</th>
<th>increased</th>
<th>decreased</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>scheme-1</td>
<td>0 doc.</td>
<td>8 doc.</td>
<td>3 doc.</td>
<td>11 doc.</td>
</tr>
<tr>
<td>scheme-2</td>
<td>0 doc.</td>
<td>49 doc.</td>
<td>6 doc.</td>
<td>55 doc.</td>
</tr>
</tbody>
</table>

can release it by the $D_{eval}$. In Table 5, for scheme-2, we can observe both the development set $S_{eval}$ and test set BLEU increase with most of the weights. In this scheme, the training data (including sentence-level tuning data) are quite different from the document-level development and test set. Hence, the efficiency of the document-level features are more obvious.

Compared with the consistency verification method, our approach uses no precise lexical features, which we rely on the baseline system to address. As a result, the proposed approach can avoid the rigidity of consistency verification and be adaptable to variant datasets.

For a further investigation, judging by their best performance, we calculate the test set BLEU for each sentence and for each document in the two schemes. The sentence-by-sentence evaluation is shown in Table 6. We find that approximately one third of the sentences have been changed from the baseline and only approximately one eighth of the sentences see an improved BLEU. Figs. 5 and 6 depict the difference of BLEU of changed sentences. The document-by-document evaluation is shown in Table 7 and depicted in Figs. 7 and 8. We can observe that most documents in each scheme have an improvement of the BLEU score. The phenomenon suggests that document-level information does disturb the performance of special sentences (as the "risky" stated in Tiedemann (2010)) because the baseline SMT system has already done a good job. However, treating the document as an evaluation unit can lead to better performance.

We show a translation example in Table 8. In the example, the French word voyage is selected to be verified and its translation is fixed to be journey. This is not a wrong translation, although more variations are usually required. On the other hand, the French word ville is translated to town in the baseline and untouched by the consistency verification method, whereas the proposed approach can select a more correct translation of city. We can see the proposed approach to be a more flexible approach than consistency verification.

6 Conclusions and Future Work

In this paper, we introduced two document-level features to improve a sentence-level baseline SMT system. In the proposed approach, we integrated document-level information to the sentence-level translation using multi-objective tuning under both a sentence- and document-level measure. Experimental results demonstrated that the approach works on different datasets and experimental schemes.

We plan to explore more document-level features and improve the search algorithm in future. We are considering applying the linear programming method in Koshikawa et al. (2010) to our document-level decoding.
Table 8: Translation example of baseline, consistency verification, and our proposed approach.

<table>
<thead>
<tr>
<th>Input</th>
<th>Reference</th>
<th>Baseline</th>
<th>Consistency Verification</th>
<th>Proposed Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>le voyage doit débuter et se terminer dans le même pays, mais pas forcément dans la même ville.</td>
<td>· travel must begin and end in the same country, but not necessarily in the same city.</td>
<td>the trip must begin and end in the same country, but not necessary in the same town.</td>
<td>the journey must begin and end in the same country, but not necessary in the same town.</td>
<td>the trip must begin and end in the same country, but not necessary in the same city.</td>
</tr>
</tbody>
</table>

Figure 5: Change on test set BLEU of sentence-level translation, best output of scheme-1, sorted by the difference (only changed translations are illustrated).

Figure 6: Change on test set BLEU of sentence-level translation, best output of scheme-2, sorted by the difference (only changed translations are illustrated).

Figure 7: Change on test set BLEU of document-level translation, best output of scheme-1, sorted by the difference.

Figure 8: Change on test set BLEU of document-level translation, best output of scheme-2, sorted by the difference.

References


A Comparison of Mixture and Vector Space Techniques for Translation Model Adaptation

Boxing Chen
Boxing.Chen@nrc-cnrc.gc.ca
Roland Kuhn
Roland.Kuhn@nrc-cnrc.gc.ca
George Foster
George.Foster@nrc-cnrc.gc.ca
National Research Council Canada, Ottawa, Canada

Abstract
In this paper, we propose two extensions to the vector space model (VSM) adaptation technique (Chen et al., 2013b) for statistical machine translation (SMT), both of which result in significant improvements. We also systematically compare the VSM techniques to three mixture model adaptation techniques: linear mixture, log-linear mixture (Foster and Kuhn, 2007), and provenance features (Chiang et al., 2011). Experiments on NIST Chinese-to-English and Arabic-to-English tasks show that all methods achieve significant improvement over a competitive non-adaptive baseline. Except for the original VSM adaptation method, all methods yield improvements in the +1.7-2.0 BLEU range. Combining them gives further significant improvements of up to +2.6-3.3 BLEU over the baseline.

1 Introduction
The translation of a source-language expression to a target language might differ across genres, topics, national origins, and dialects, or the author’s or publication’s style. The word “domain” is often used to indicate a particular combination of all these factors (Chen et al., 2013b). Statistical machine translation (SMT) systems are trained on bilingual parallel and monolingual data. The training data vary across domains, and translations across domains are unreliable. Therefore, we can often get better performance by adapting the SMT system to the test domain.

Domain adaptation (DA) techniques for SMT systems have been widely studied. Approaches that have been tried for SMT model adaptation include self-training, data selection, data weighting, context-based DA, and topic-based DA, etc. We will review these techniques in the next Section. Among all these approaches, data weighting has received most attention, it assigns each data item a weight according to its closeness to the in-domain data. Mixture model and vector space model adaptation (Chen et al., 2013b) are two data weighting techniques. Both of them assume the existence of $N$ bilingual sub-corpora as training data, and have weights tuned on an in-domain development corpus (dev).

Mixture model adaptation assigns weights at corpus-level; sub-models trained on different domain data sets are combined linearly or log-linearly (Foster and Kuhn, 2007). Provenance features (Chiang et al., 2011) compute a separate set of lexical weights for each sub-corpus, and then log-linearly combined. While vector space model adaptation (Chen et al., 2013b) assigns weights at phrase-level; it weights each phrase pair in the training data with the similarity score between the phrase pair and the in-domain dev data.

In this paper, we first propose two extensions to the original vector space model adaptation, thereby invent two new data weighting adaptation methods based on the vector space model. The first one is grouped VSM adaptation, which classifies the in-domain data to several groups,
such as general-domain, domain-specific phrase pair, then assigns several weights to the phrase pair with the similarity scores between the phrase pair and the dev data sub-sets. The second one is called distributional VSM adaptation, which directly uses the phrase pair’s distribution across sub-corpora as decoding features. The two methods both significantly improve the SMT performance over the original VSM adaptation.

The second contribution of this paper is a systematic comparison of two groups of DA methods under the most common scenario for SMT: that the training material is heterogeneous, and an in-domain development set is available. The first group of adaptation technique is mixture model, includes linear mixtures, log-linear mixtures, provenance features; the other group of adaptation technique is vector space model, includes original vector space model (VSM), and the newly proposed grouped VSM and distributional adaptation. The mixture model and vector space adaptation techniques both rely on information about the sub-corpora from which the data originate. However, a key difference is that vector space model methods capture each phrase pair’s distribution across sub-corpora, while in mixture models, a phrase pair’s distribution is the prevalence of the pair within each subcorpus. Given this difference, it is interesting to have a systematic comparison between them.

Another small but nice contribution is that inspired from (Chiang et al., 2011) and (George Foster and Kuhn, 2013), we use a simple form of smoothing for log-linear mixture adaptation, which significantly improves performance over the non-smoothed log-linear mixtures.

Experiments on NIST Chinese-to-English and Arabic-to-English tasks show that 1) each technique yields significant improvement over a competitive but non-adaptive baseline; 2) the largest improvements are for five of the six methods (those other than the original VSM), with improvements for these five all in the range +1.7-2.0 BLEU; 3) combining some of these techniques yields further significant improvement. The best combination yields improvements of +2.6-3.3 BLEU over the baseline.

2 Reviewing of SMT adaptation techniques

Most approaches to DA can be classified into one of five categories: self-training, data selection, data weighting, context-based DA, and topic-based DA. Recently, several new approaches have also been studied.

With self-training (Ueffing and Ney, 2007; Chen et al., 2008; Schwenk, 2008; Bertoldi and Federico, 2009), an MT system trained on general domain data is used to translate in-domain monolingual data. The resulting target sentences or bilingual sentence pairs are then used as additional training data.

Data selection approaches (Zhao et al., 2004; Lü et al., 2007; Moore and Lewis, 2010; Axelrod et al., 2011; Duh et al., 2013) search for monolingual or bilingual parallel sentences that are similar to the in-domain data according to some criterion, then add them to the training data.

Data weighting approaches can be seen as a generalization of data selection: instead of making a binary include vs. not-include decision about a given sentence or sentence pair, one weights each data item according to its closeness to the in-domain data. This can be applied at corpus, sentence, or phrase level. At corpus level, linear and log-linear mixture combine sub-models trained on different domain data sets linearly or log-linearly (Foster and Kuhn, 2007). Log-linear mixture DA can employ the same discriminative tuning algorithm used to combine the log-linear high-level components of a typical SMT system (the translation model, language model, reordering model, etc.), but linear mixture DA seems to work better. Thus, (Foster et al., 2010; Senrich, 2012; Chen et al., 2013a; George Foster and Kuhn, 2013) studied linear mixture adaptation. (Koehn and Schroeder, 2007), instead, combined the sub-models via
alternative paths. Provenance features (Chiang et al., 2011) compute a separate set of lexical weights for each sub-corpus, then all these lexical weights are combined log-linearly. (Razmara et al., 2012) used ensemble decoding to mix multiple translation models. (Senrich et al., 2013; Cui et al., 2013) extended the mixture model method to multi-domain DA.

At sentence level, (Matsoukas et al., 2009) used a rich feature set to compute weights for each sentence in the training data. A sentence from a corpus whose domain is close to that of the in-domain dev set would receive a high weight. At a finer level of granularity, (Foster et al., 2010) used a rich feature set to compute phrase pair weights. Vector space model adaptation (Chen et al., 2013b) is another phrase-level data weighting approach; it weights each phrase pair with the similarity score between the in-domain dev data and each phrase pair in the training data based on vector space model in which vectors (for the entire dev data, or for each phrase pair) represent domain profiles.

The cache-based method (Tiedemann, 2010; Gong et al., 2011) is a form of context-based adaptation technique. In the tradition of (Kuhn and De Mori, 1990) it uses a cache to consider the local or document-level context when choosing translations. (Carpuat et al., 2013) employed word sense disambiguation, using local context to distinguish the translations for different domains.

Work on topic-based DA includes (Tam et al., 2007), where latent semantic analysis (LSA) models topics for SMT adaptation, (Eidelman et al., 2012; Hewavitharana et al., 2013) which employs a latent Dirichlet allocation (LDA) topic model, and (Eva Hasler and Koehn, 2012), which employs hidden topic Markov models (HTMMs), adding a sentence topic distribution as an SMT system feature.

Other DA approaches include mining translations from comparable data to translate OOVs and capturing new senses in new domains (Daume III and Jagarlamudi, 2011; Irvine et al., 2013). (Dou and Knight, 2012; Zhang and Zong, 2013) learned bilingual lexica or phrase tables from in-domain monolingual data with a decipherment method, then incorporated them into the SMT system.

3 Mixture model adaptation

There are several adaptation scenarios for SMT, of which the most common is 1) The training material is heterogeneous, with some parts of it that are not too far from the test domain; 2) A bilingual dev set drawn from the test domain is available. A common approach to DA for this scenario is: 1) split the training data into sub-corpora by domain; 2) train sub-models or features on each sub-corpus; 3) weight these sub-models or features via machine learning or SMT tuning algorithms.

In the rest of the paper, we assume that the training data are split into $N$ sub-corpora. We apply various adaptation techniques to a phrase-based SMT system whose translation model incorporates forward and backward phrase translation probabilities and forward and backward lexical weights. Linear and log-linear mixture techniques are applied to phrase translation probabilities, provenance features to the lexical weights.

3.1 Linear mixture model

Given a set of phrase tables, each trained on one of the $N$ sub-corpora and with “forward” and “backward” probabilities for phrase pairs, these sub-models can be combined linearly. Let us consider the “backward” probability $p(s|t)$ of source-language phrase $s$ being generated by target-language phrase $t$ (for all the adaptation techniques, each technique is applied symmetrically in “backward” and “forward” directions). For a set of $p_i(s|t)$, each trained on a sub-corpus $d_i$, the mixture model is
\[
p(s|t) = \sum_{i=1}^{N} \alpha_i p_i(s|t)
\]  

(1)

To set weights \(\alpha_i\), we first extract a set of phrase pairs from an in-domain dev set using standard techniques. This yields a joint distribution \(\hat{p}\), which we use to define a maximum likelihood objective as in Equation 2. The weights can then be learned efficiently using the EM algorithm. This estimation approach was first proposed in (Foster et al., 2010).

\[
\hat{\alpha} = \arg \max_{\alpha} \sum_{s,t} \hat{p}_i(s, t) \log \sum_{i=1}^{N} \alpha_i p_i(s|t)
\]  

(2)

A problem with this approach is that large phrase tables with good phrase pair coverage get assigned high weights by EM. When such tables are out-of-domain, this will cause translation performance to suffer: the in-domain translations for which they got credit under EM will be replaced by out-of-domain alternatives to which they assign higher probability. To correct this bias in favour of large corpora, (George Foster and Kuhn, 2013) proposed sub-sampling phrase tables. Thus, before running EM, we randomly select roughly the same number of phrase pairs covered by the in-domain dev set from each phrase table to learn the weights. While, the actual interpolated phrase table uses the full data.

### 3.2 Log-linear mixture model

Like the linear mixture model, the log-linear mixture model is made up of sub-models trained on each of the sub-corpora. However, it combines the sub-models multiplicatively by directly using those sub-models as features in the SMT log-linear framework. Thus, one is tuning directly to the desired objective function (e.g., BLEU) instead of to likelihood, as EM does.

The standard implementation of log-linear mixtures has a serious disadvantage compared to linear mixtures: it performs badly when there are many sub-corpora, and the sub-models are not smoothed. Within a log-linear combination, all sub-models must agree on successful hypotheses. Linear mixtures implement a kind of “or”, while log-linear models implement a kind of “and”: if \(p_i(s|t)\) is zero because phrase pair \((s, t)\) did not occur in sub-corpus \(d_i\), \(p(s|t)\) will be estimated as zero. One can assign and tune small positive probabilities for the \(p_i(s|t)\) for missing phrase pairs, but this tends not to work very well. “Small” models thus need to be down-weighted heavily to avoid excessive vetoing; whatever useful information they might possess is also discarded (George Foster and Kuhn, 2013). Linear mixture models are far more forgiving of phrase pairs that are missing from some of the sub-corpora. Inspired from (Chiang et al., 2011) and (George Foster and Kuhn, 2013), to address this problem, we propose the following smoothing scheme:

\[
\hat{p}_i(s|t) = (1 - \lambda)p_i(s|t) + \lambda \hat{p}_a(s|t)
\]  

(3)

where \(\hat{p}_a(s|t)\) is trained on all training data, and \(\lambda\) is tuned on held-out data. The sub-model weights \(\hat{p}_i(s|t)\) are then learned under the standard SMT log-linear framework. Experiments (see 5.3) show that this simple smoothing greatly improves the performance of log-linear mixture adaptation.

---

1We use the models trained on the whole training data to align the dev set. This can be done with mgiza (http://www.kyloo.net/software/doku.php/mgiza)
3.3 Provenance features

Provenance features (Chiang et al., 2011) are applied to lexical weights. There are slight variations in computing lexical weights (Foster et al., 2006), they all use forward and backward word translation probabilities $T(s|t)$ and $T(t|s)$ estimated from the word-aligned parallel text. The conditional probability for word pair $(s, t)$ in a translation table is computed as below:

$$T(s|t) = \frac{\text{count}(s, t)}{\sum_s \text{count}(s, t)}.$$  \hspace{1cm} (4)

We adopt the approach proposed in (Zens and Ney, 2004) to compute the lexical weights. It assumes that all source words are conditionally independent:

$$p_{lw}(s|t) = \prod_{i=1}^{n} p(s_i|t)$$  \hspace{1cm} (5)

and adopts a “noisy-or” combination, so that

$$p(s_i|t) = 1 - \prod_{j=1}^{m} (1 - T(s_i|t_j))$$  \hspace{1cm} (6)

where $n$ and $m$ are number of source and target words in the phrase pair $(s, t)$ respectively.

To compute the provenance features, we first estimate the word translation tables $T(s|t)$ and $T(t|s)$ trained on the $N$ sub-corpora. However, many word pairs are unseen for the word translation table of a given sub-corpus. Following (Chiang et al., 2011), we smooth the translation tables:

$$\hat{T}_i(s|t) = (1 - \lambda)T_i(s|t) + \lambda T_a(s|t)$$  \hspace{1cm} (7)

where $T_a(s|t)$ is the word translation table trained on all training data. $T_i(s|t)$ is the conditional probability for word pair $(s, t)$ in a translation table extracted from the $i$th sub-corpus. After we obtain the smoothed word translation lexicons for each sub-corpora, we compute the lexical weights using Equation 5, therefore, we obtain $2 \times N$ provenance features.

4 Extensions to vector space model (VSM) adaptation

The original vector space model (VSM) adaptation was proposed in (Chen et al., 2013b), two new variants are proposed in this paper.

4.1 Original VSM adaptation

The version of VSM in (Chen et al., 2013b) compares the domain vector profile of the in-domain dev set with a profile for the phrase pairs extracted from the training data. The similarity score for these two vectors is used as a decoding feature. This VSM variant will be called “original” VSM.

The domain vector for phrase-pair $(s, t)$ is a vector where each entry reflects the contribution of a particular sub-corpus to the phrase pair. It is defined as a vector of standard $tf(s, t) \cdot idf(s, t)$ weights, where $tf(s, t)$ is the raw joint count $c_i(s, t)$ in the corpus $d_i$ normalized by dividing by the maximum raw count of any phrase pair extracted in the corpus $d_i$.\footnote{We tried normalizing with the total count or not normalizing: normalizing with the maximum count works better in practice.} Let
\[ tf_i (s, t) = \frac{c_i (s, t)}{\max \{ c_i (s_j, t_k), (s_j, t_k) \in d_i \}}. \] (8)

The \( \text{idf} (s, t) \) is the inverse document frequency. We use the standard formula:

\[ \text{idf} (s, t) = \log \left( \frac{N}{\text{df}(s, t)} + C \right), \] (9)

where \( \text{df}(s, t) \) is the number of sub-corpora that \( (s, t) \) appears in, and \( C \) is an empirically determined smoothing term.

To calculate the domain similarity score between a phrase pair and the in-domain data, we (following (Chen et al., 2013b)) compute the Bhattacharya coefficient (BC) (Bhattacharyya, 1943). To map the BC score onto a range from 0 to 1, we normalize each weight in the vector by dividing it by the sum of the weights. Thus, we get the probability of a phrase pair in the \( i \)th sub-corpus:

\[ p_i (s, t) = \frac{tf_i (s, t) \cdot \text{idf}(s, t)}{\sum_{j=1}^{N} tf_{j} (s, t) \cdot \text{idf}(s, t)} \] (10)

\[ p_i (s, t) = \frac{tf_i (s, t)}{\sum_{j=1}^{N} tf_{j} (s, t)} \] (11)

For the in-domain dev set, we first run word alignment and phrase extraction in the usual way, then sum the distribution of each phrase pair \( (s_j, t_k) \) extracted from the dev across sub-corpora to represent its domain information. The \( i \)th component of the dev domain vector is thus

\[ w_i (\text{dev}) = \sum_{j=0}^{J} \sum_{k=0}^{K} c_d (s_j, t_k) tf_i (s_j, t_k) \cdot \text{idf}(s_j, t_k) \] (12)

\( J, K \) are the total number of source/target phrases extracted from the dev data respectively. \( c_d (s_j, t_k) \) is the joint count of phrase pair \( (s_j, t_k) \) found in the dev set. \( p_i (\text{dev}) \) is normalized similarly.

\[ p_i (\text{dev}) = \frac{w_i (\text{dev})}{\sum_{j=1}^{N} w_j (\text{dev})} \] (13)

The Bhattacharya coefficient (BC) is defined as:

\[ BC(\text{dev}; s, t) = \sum_{i=0}^{N} \sqrt{p_i (\text{dev}) \cdot p_i (s, t)} \] (14)

4.2 Grouped VSM

The original VSM adaptation computes the in-domain domain vector with a set of phrase pairs extracted from the in-domain dev set. However, even a highly domain-specific dev set contains both domain-specific phrase pairs and general-domain phrase pairs. For example, in “I have two computers, one is a Dell laptop, another one is a HP desktop”, the phrases “Dell laptop”, and “HP desktop” are from the computer domain, while the phrases “I have”, “two”, “one is”, “another one is” are general. Therefore, we devised a new form of VSM which groups the in-domain phrase pairs into subsets, which are then used for VSM adaptation. That is we partition the phrase-pairs of the dev set into \( K \) subsets, then derive a domain vector and similarity score for each, resulting in \( K \) similarity scores as decoding features.
After initial attempts using \(K\)-medoid and \(K\)-mean clustering to group the phrase pairs, which we abandoned because these two algorithms were computationally expensive and yielded poor DA, we implemented a simple, cheap grouping algorithm. This algorithm can classify the phrase pairs into 2 or \(N+1\) subsets, where \(N\) is the number of sub-corpora. We first create \(N+1\) pseudo-domain vectors, each with \(N\) entries: the first \(N\) vectors form an \(N \times N\) identity matrix, and the last vector represents the uniform probability vector - each of its entries is \(1/N\) as in Equation 15. Then, we calculate the similarity score for the domain profile vector for each phrase pair in the dev set with each of the \(N+1\) pseudo-vectors. The phrase pairs are then grouped into \(N+1\) subsets according to their biggest similarity score. Thus, each phrase pair in the dev set is assigned to the domain for which it has the largest component, except for general phrase pairs, which are assigned to domain \(N+1\).

\[
[1, 0, ..., 0], [0, 1, ..., 0], ..., [0, ..., 1, ..., 0], [0, 0, ..., 1], \text{ and } [\frac{1}{N}, \frac{1}{N}, ..., \frac{1}{N}].
\]  

(15)

\[
lab(s,t) = \arg\max_{i=1,...,N+1} BC(pvec_i; s, t)
\]

(16)

where \(lab(s,t)\) is the subset label of phrase pair \((s,t)\), and \(pvec_i\) is the \(i\)th pseudo-vector in Equation 15.

If we want to split the dev phrase pairs into only two groups - domain-specific and general-domain - we merge the first \(N\) subsets into a domain-specific set, while the last subset of phrase pairs is the general-domain set. To group the phrase pairs into \(N \) subsets, i.e. if we don’t want to have a general-domain sub-set, we remove the last, uniform-domain vector from the procedure.

After grouping the phrase-pairs of the dev set into \(K\) subsets, we then compute a domain vector and similarity score for each using the algorithm in the section 4.1. Then, the \(K\) similarity scores are used as decoding features.

### 4.3 Distributional VSM

This is the second new version of VSM. Instead of computing a similarity score between the domain vectors of the phrase pair and the dev set, we propose to use the entries of the phrase pair’s domain vector directly as decoder log-linear features. Those features, i.e. the probability distribution in Equation 11, indicate each phrase pair’s distribution across sub-corpora. Therefore, we call this method distributional VSM adaptation. This method has a few advantages compared to the original VSM adaptation. In the original VSM adaptation, we need a similarity function to compute the domain similarity between the dev set and phrase pair, such as the Bhattacharya coefficient in Equation 14. To select a good similarity function needs additional experiments or a priori knowledge, like (Chen et al., 2013b) did. And there are also some free parameters needs to be tuned before-hand, such as \(C\) in Equation 9. But the distributional VSM adaptation can avoid these additional experiments, moreover, experiments (see 5.3) show that this method greatly improves the performance over the original VSM adaptation.

### 5 Experiments

#### 5.1 Data setting

We carried out experiments in two different settings, both involving data from NIST Open MT 12.\(^3\) The first setting is based on data from the Chinese to English constrained track, comprising about 283 million English running words. We manually grouped the training data into 14 corpora according to genre and origin. Table 1 summarizes information about the training.

\(^3\)http://www.nist.gov/itl/iad/mig/openmt12.cfm
Table 1: NIST Chinese-English data. In the genres column: nw=newswire, bc=broadcast conversation, bn=broadcast news, wl=weblog, ng=newsgroup, un=United Nations proceedings.

devtest
genres
corpus  # segs  # en tok  %
fbis  250K  10.5M  3.7
financial  90K  2.5M  0.9 financial
bc  79K  1.3M  0.5
bn  75K  1.8M  0.6
nw  25K  696K  0.2
bc  24K  596K  0.2
nw  1.3M  39.5M  14.0
Hansard
hk  400K  9.3M  3.3
legal
kn  702K  16.6M  5.9
nw
is  558K  18.0M  6.4
wl
lex&ne  1.3M  2.0M  0.7
lexicon
others nw  146K  5.2M  1.8
nw
sinorama  282K  10.0M  3.5
nw
un  5.0M  164M  58.2
un
TOTAL  10.1M  283M  100.0
(all)

devtest
genres
corpus  # segs  # en tok  %
tune  1.506  161K
nw
wl
NIST06  1.664  189K
nw
bn
ng
NIST08  1.357  164K
nw
wl

5.2 System
Experiments were carried out with an in-house, state-of-the-art phrase-based system. The whole corpora were first word-aligned using IBM2, HMM, and IBM4 models, then split to subcorpora according to genre and origin; the phrase table was the union of phrase pairs extracted from these alignments, with a length limit of 7. The translation model (TM) was Kneser-Ney smoothed in both directions (Chen et al., 2011). We use the hierarchical lexicalized reordering model (RM) (Galley and Manning, 2008), with a distortion limit of 7, lexical weighting in both directions, word count, a distance-based reordering model, a 4-gram language model (LM) trained on the target side of the parallel data, and a 6-gram English Gigaword LM. The system was tuned with batch lattice MIRA (Cherry and Foster, 2012).

5.3 Results
This paper has described three modifications to existing techniques: smoothing for log-linear mixtures and two new versions of VSM - grouped VSM and distributional VSM. First, we did
Table 2: NIST Arabic-English data. In the *genres* column: nw=newswire, bc=broadcast conversation, bn=broadcast news, ng=newsgroup, wl=weblog.

<table>
<thead>
<tr>
<th>corpus</th>
<th># segs</th>
<th># en toks</th>
<th>% genres</th>
</tr>
</thead>
<tbody>
<tr>
<td>gale_bc</td>
<td>57K</td>
<td>1.6M</td>
<td>3.3 bc</td>
</tr>
<tr>
<td>gale_bn</td>
<td>45K</td>
<td>1.2M</td>
<td>2.5 bn</td>
</tr>
<tr>
<td>gale_ng</td>
<td>21K</td>
<td>491K</td>
<td>1.0 ng</td>
</tr>
<tr>
<td>gale_nw</td>
<td>17K</td>
<td>659K</td>
<td>1.4 nw</td>
</tr>
<tr>
<td>gale_wl</td>
<td>24K</td>
<td>590K</td>
<td>1.2 wl</td>
</tr>
<tr>
<td>isi</td>
<td>1,124K</td>
<td>34.7M</td>
<td>72.6 nw</td>
</tr>
<tr>
<td>other_nw</td>
<td>224K</td>
<td>8.7M</td>
<td>18.2 nw</td>
</tr>
<tr>
<td>TOTAL</td>
<td>1,512K</td>
<td>47.8M</td>
<td>100.0 (all)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>devtest</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NIST06</td>
<td>1,664</td>
<td>202K</td>
</tr>
<tr>
<td>NIST08</td>
<td>1,360</td>
<td>205K</td>
</tr>
<tr>
<td>NIST09</td>
<td>1,313</td>
<td>187K</td>
</tr>
</tbody>
</table>

Table 3: Results of log-linear mixtures with or without smoothing. */** or +/++ means result is significantly better than baseline or system without smoothing (*p < 0.05 or *p < 0.01, respectively).

<table>
<thead>
<tr>
<th></th>
<th>Chinese-English</th>
<th>Arabic-English</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MT06</td>
<td>MT08</td>
</tr>
<tr>
<td>baseline</td>
<td>36.0</td>
<td>29.4</td>
</tr>
<tr>
<td>log-lin w/o smooth</td>
<td>35.8</td>
<td>29.0</td>
</tr>
<tr>
<td>log-lin w/ smooth</td>
<td>37.9**++</td>
<td>31.1**++</td>
</tr>
</tbody>
</table>

Table 4 shows experiments with grouped VSM DA. We classify the phrase pairs extracted from in-domain dev set into *K* groups. The *K* = 1 case is equivalent to original VSM. The *K* = 2 case labels every phrase pair as either domain-specific or general-domain. The *K* = *N* case means that every phrase pair is labeled with the number of the closest sub-corpus (there are *N* sub-corpora). Finally, the *K* = *N* + 1 case labels every phrase pair either with the number of the closest sub-corpus, or as general-domain (the *N* + 1 group). Performance steadily improves as the number of groups increases.
Table 4: Grouped VSM adaptation. */** or +/-++ means result is significantly better than baseline or original VSM adaptation system, i.e., $K = 1$. (p < 0.05 or p < 0.01, respectively).

<table>
<thead>
<tr>
<th></th>
<th>Chinese-English</th>
<th>Arabic-English</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MT06</td>
<td>MT08</td>
</tr>
<tr>
<td>baseline</td>
<td>36.0</td>
<td>29.4</td>
</tr>
<tr>
<td>$K = 1$</td>
<td>37.0*</td>
<td>30.3*</td>
</tr>
<tr>
<td>$K = 2$</td>
<td>37.3**+</td>
<td>30.6**</td>
</tr>
<tr>
<td>$K = N$</td>
<td>37.9***+</td>
<td>31.2*****+</td>
</tr>
<tr>
<td>$K = N + 1$</td>
<td>38.2***++</td>
<td>31.6*****++</td>
</tr>
</tbody>
</table>

Table 5: The results of baseline, original VSM adaptation and distributional VSM adaptation.

Table 5 compares original VSM and distributional VSM. It shows that the original VSM adaptation reported in (Chen et al., 2013b) does yield improvement over a non-adaptive baseline. However, if instead of computing a domain similarity score, we directly maximize BLEU on the dev set by tuning the weights of distribution features, we got further significant improvements. On the Chinese task, the further improvements were +0.7-0.8 BLEU, while on Arabic, the further improvements were smaller but still significant: +0.3-0.4 BLEU. (Cherry, 2013) showed that in the case of reordering features, directly maximizing BLEU outperforms maximum entropy optimization; the experiments in Table 5 yield a similar conclusion. Given that recently developed tuning algorithms such as MIRA can handle a very large feature set, we may consider having all possible features directly tuned to maximize BLEU (or similar criteria).

Now, we compare all six DA techniques, Table 6 reports the results. All techniques improved on all test sets across two language pairs over the non-adaptive baseline, and all these improvements are significant. From the average absolute improvement across test sets, the original VSM adaptation yield improvement of around 1.1 BLEU on average. And it is inferior to the remaining five DA techniques; these five techniques obtained similar improvement over the baseline, around +1.7-2.0 BLEU. Linear mixtures did best on Arabic MT08, provenance features did best on Chinese MT06, while $N + 1$ grouped VSM did best on the other two sets. So there is no clear winner among these techniques.

Our last experiment studies whether all these DA techniques exploit the same information, or if they are somewhat complementary. The results are reported in Table 7. “Single best” is a strong baseline, since it involves the best result in each column of Table 6 rather than the best single row. We first figured out the best mixture model combination, by exploring all 4 possible combinations of the three mixture model adaptation methods. We found that combination of linear mixture and provenance features got the best performance, even better than using all three mixture model methods together. The possible reason is that they are likely to be complementary, since the former adapts phrase translation probabilities and the latter adapts lexical weights. This combination did very well on Arabic-English but gave only small improvements on Chinese-English.

Then, we add VSM adaptation method by a form of greedy search: beginning with the best
Table 6: The comparison of all 7 adaptation techniques. The \( \Delta \) is computed on all four test sets.

<table>
<thead>
<tr>
<th></th>
<th>Chinese-English</th>
<th>Arabic-English</th>
<th>avg ( \Delta )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MT06</td>
<td>MT08</td>
<td>MT08</td>
</tr>
<tr>
<td>baseline</td>
<td>36.0</td>
<td>29.4</td>
<td>46.4</td>
</tr>
<tr>
<td>linnmix</td>
<td>37.8</td>
<td>31.2</td>
<td><strong>48.8</strong></td>
</tr>
<tr>
<td>smoothed log-linmix</td>
<td>37.9</td>
<td>31.1</td>
<td>48.2</td>
</tr>
<tr>
<td>provenance</td>
<td><strong>38.4</strong></td>
<td>31.4</td>
<td>48.2</td>
</tr>
<tr>
<td>original VSM</td>
<td>37.0</td>
<td>30.3</td>
<td>47.7</td>
</tr>
<tr>
<td>distr. VSM</td>
<td>38.0</td>
<td>31.2</td>
<td>48.0</td>
</tr>
<tr>
<td>( N + 1 ) gr. VSM</td>
<td>38.2</td>
<td><strong>31.6</strong></td>
<td>48.3</td>
</tr>
</tbody>
</table>

mixture model combination and then incorporating the technique that yields the highest BLEU improvement over the system it’s added to. For the next combination, we add \( N + 1 \) grouped VSM adaptation and observe significant improvements over “single best” for all test sets. For the final combination, we add distributional VSM DA, gaining +2.9 BLEU on both Chinese MT06 and MT08, +3.3 BLEU on Arabic MT08 and +2.6 BLEU on Arabic MT09 over the non-adaptive baseline. Adding the two VSM adaptation methods to the best mixture combinations, further significant improvements were obtained on three out of four test sets. We were unable to obtain further gains by adding the original VSM adaptation technique to the system: apparently, the combination of four techniques shown in the last row of the table covers all the information exploited by the six DA techniques described in this paper.

Table 7: Combinations of techniques. */** or +/++ means result is significantly better than single best result in Table 6 (“single best” row) or system without newly added technique (i.e., than previous row) \( p < 0.05 \) or \( p < 0.01 \), respectively). # means the result in the final combination (the last row) is significantly better than the best mixture model combination (“linmix & prov.” row) at level \( p < 0.05 \).

<table>
<thead>
<tr>
<th></th>
<th>Chinese-English</th>
<th>Arabic-English</th>
<th>avg ( \Delta )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MT06</td>
<td>MT08</td>
<td>MT08</td>
</tr>
<tr>
<td>baseline</td>
<td>36.0</td>
<td>29.4</td>
<td>46.4</td>
</tr>
<tr>
<td>single best</td>
<td>38.4</td>
<td>31.6</td>
<td>48.8</td>
</tr>
<tr>
<td>linnmix</td>
<td>37.8</td>
<td>31.2</td>
<td>48.8</td>
</tr>
<tr>
<td>prov. (provenance)</td>
<td>38.4</td>
<td>31.4</td>
<td>48.2</td>
</tr>
<tr>
<td>linnmix &amp; prov.</td>
<td>38.6</td>
<td>31.7+</td>
<td>49.3**+</td>
</tr>
<tr>
<td>linnmix &amp; prov. &amp; gr. VSM</td>
<td>38.7*</td>
<td>32.0*+</td>
<td>49.6**+</td>
</tr>
<tr>
<td>linnmix &amp; prov. &amp; gr. VSM &amp; distr. VSM</td>
<td>38.9**</td>
<td>32.3**+#</td>
<td>49.7***#</td>
</tr>
</tbody>
</table>

6 Conclusions

We have proposed two extensions to the original vector space model (VSM) adaptation, which are distributional VSM, and grouped VSM. They both significantly improved over the original VSM adaptation. We also improved log-linear mixture significantly by smoothing the mixture components. Then we systematically compared the VSM adaptation techniques to the mixture model adaptation techniques, which includes linear mixture, log-linear mixture and provenance features. According to BLEU, the original VSM DA has weaker performance than the other techniques, yielding around 1.1 improvement on average over a non-adaptive baseline. The
remaining five techniques obtained similar improvements across four test sets in two large-scale data conditions, in the range +1.7-2.0 BLEU on average; there is no clear winner among them.

Although the improvements obtained from these techniques are not strictly additive, combining a subset of them yields further significant improvement: we obtained +2.6-3.3 BLEU improvement on average over the non-adaptive baseline with our best combination. In future work, we will try combining language model and reordering model adaptation with the techniques we have described here.

Acknowledgement

This research was supported in part by DARPA contract HR0011-12-C-0014 under subcontract to Raytheon BBN Technologies.

References


Combining Domain and Topic Adaptation for SMT

Eva Hasler

Barry Haddow

Philipp Koehn

1 ILCC, School of Informatics, University of Edinburgh

2 Center for Language and Speech Processing, Johns Hopkins University

Abstract

Recent years have seen increased interest in adapting translation models to test domains that are known in advance as well as using latent topic representations to adapt to unknown test domains. However, the relationship between domains and latent topics is still somewhat unclear and topic adaptation approaches typically do not make use of domain knowledge in the training data. We show empirically that combining domain and topic adaptation approaches can be beneficial and that topic representations can be used to predict the domain of a test document. Our best combined model yields gains of up to 0.82 BLEU over a domain-adapted translation system and up to 1.67 BLEU over an unadapted system, measured on the stronger of two training conditions.

1 Introduction

Domain adaptation is a very active area of research in statistical machine translation (SMT) and there is a large and growing body of work on different techniques to adapt translation and language models (Foster and Kuhn, 2007; Matsoukas et al., 2009; Foster et al., 2010; Sennrich, 2012) to specific target domains that are usually known in advance, for example the news domain. An extension of the standard domain adaptation task is multi-domain adaptation where a translation system is adapted to several known target domains (see for example Cui et al. (2013)). In cases where the target domains are not assumed to be known, dedicated domain classifiers can be trained and used to automatically predict the target domain and choose an appropriate model based on the prediction (Banerjee et al., 2010).

Recently, there has been increased work on the application of topic modelling to translation model adaptation (Gong et al., 2011; Su et al., 2012; Eidelman et al., 2012; Hewavitharana et al., 2013; Hasler et al., 2014a) in an attempt to move away from the notion of a domain as the source of a corpus. Topic adaptation techniques build on the assumption that the origin of sentences and documents is unknown at test time, and the unsupervised nature of topic models is useful for detecting structure across corpus boundaries in training sets to adapt to diverse test sets. While domain adaptation techniques rely on a given, hard clustering of the data, topic adaptation aims to induce a soft clustering that is more suited to the task.

While topic models are very useful for detecting and grouping the semantic differences in documents, making use of our knowledge about corpus boundaries in the training data could potentially help to adapt more specifically to style or genre on top of adapting to topics. By predicting the domain label of test documents, we can combine both approaches to translate unlabelled documents from different genres and topics. We show that domain and topic adaptation can be complementary and that finding the right balance between the two could lead to a more
efficient architecture that combines online and offline computation.

2 Related work

Domain classification for multi-domain adaptation has been the focus of several researchers in recent years. Xu et al. (2007) tune domain-specific features weights and build domain-specific language models. They use the perplexity of in-domain language models to classify test documents and select the appropriate weights and models per document. Banerjee et al. (2010) train domain-specific translation models and use SVMs to detect the domain of an input sentence to route it to a domain-specific model. Wang et al. (2012) follow a slightly different approach by re-using the same translation model for all domains and tuning domain-specific features weights with modified objectives. Sennrich et al. (2013) adapt the four standard translation model features to unsupervised clusters of the development data obtained by k-means clustering.

Another line of research aims to improve topic modelling by encoding domain information via a Dirichlet Forest Prior (Andrzejewski et al., 2009). By specifying Must-Link and Cannot-Link relations between words, topic modelling is guided to either separate words into different topics or merge them into the same topic. While the idea of combining domain and topic adaptation within the same model is appealing, the model requires manually constructed lists of words and seems more suited for fine-tuning specific topics, a process they call interactive topic modeling.

Different from previous work, we investigate the utility of combining domain adaptation with topic adaptation to capture potentially different levels of structure in test documents of unknown origin. We also show that topic modelling makes it straightforward to predict the domain of a test document, circumventing the need for separately trained domain classifiers. This allows us to combine domain-adapted translation models with topic-adapted models dynamically at test time.

3 Topic modelling approach

We follow the approach described in Hasler et al. (2014b) to build a Phrase Pair Topic (PPT) model. We use a model with 50 topics for all translation experiments and evaluate different numbers of topics for the domain classifiers. In the PPT model, phrase pairs are represented as distributional profiles which are pseudo documents containing source words found in all the sentence contexts of a phrase pair in the training data. These pseudo documents are the input to the topic model which clusters context words into topics and infers a topic distribution for each phrase pair. At test time, context topic vectors are inferred by applying the model to each of the test documents. This contextual information can be used to measure the cosine similarity between a document context and each applicable phrase pair or for other topic-adapted features as described in the next section. Context adaptation works by adding topic-adapted features to each phrase pair in the filtered phrase table, depending on its topical similarity with the test document. Thus, the topic-adapted features are recomputed for each test document.

3.1 Topic features

We consider several sets of topic features all of which are derived from distributions learned by the PPT model. The feature sets contain the individual features described below, where $s$ and $t$ denote a source and target phrase, $c$ denotes the test document context, $k$ denotes a latent topic and $\theta$ denotes a topic vector:
Conditional translation probability

\[ p(t|s,c) = \sum_k p(t,k|s,c) \]
\[ p(t,k|s,c) \propto p(t,s,k|c) = p(t|s,k) \cdot p(s|k) \cdot p(k|c) \]

Joint-conditional probability

\[ p(t,c|s) = p(c|t,s) \cdot p(t|s) \approx p(\theta_c|\theta_{pp}) \cdot p(t|s) \approx \cos(\theta_c|\theta_{pp}) \cdot p(t|s) \]

Target-unigrams

\[ trgUnigrams_t = \prod_{i=1}^{\text{num}} \frac{P_{doc}(w_i)}{P_{baseline}(w_i)} \cdot \frac{P_{doc}(w_i)}{P_{topic0}(w_i)} \]
\[ f(x) = \frac{2}{1 + \frac{1}{x}} \quad x > 0 \]

Sim-phrasePair similarity = \( \cos(\theta_{pp}, \theta_c) \)

Sim-targetPhrase similarity = \( \cos(\theta_{tp}, \theta_c) \)

Sim-targetWord similarity = \( \cos(\theta_{tw}, \theta_c) \)

The first two features are probabilistic features that take the topical context into account in computing the probability of a target phrase given a source phrase. The first feature, **Conditional**, factorises the joint probability of a target phrase \( t \), source phrase \( s \) and topic \( k \) given a context \( c \) into the probabilities \( p(t|s,k) \) and \( p(s|k) \), which are estimated from relative counts of how often source and target phrases co-occur with each topic in the distributional profiles, and \( p(k|c) \) which represents the inferred topic mixture for the test context. The second feature, **Joint-conditional**, estimates the joint probability of a target phrase and a test context given a source phrase. It is factorised as the (baseline) probability of a target phrase given a source phrase and the probability of the test context given the source and target phrase. The latter is approximated by the probability of the test context topic mixture given the phrase pair topic mixture, which is further approximated by the cosine similarity between the two topic mixtures.

The **Target-unigrams** feature is inspired by the lazy MDI adaptation of Ruiz and Federico (2012) and measures the probability ratio of a word under the document topic mixture versus under the baseline model\(^1\). We include an additional term to measure the topical relevance of a word by comparing against its probability under the asymmetric topic 0 of the PPT model\(^2\). **Sim-phrasePair** measures the cosine similarity of a phrase pair topic vector and the topic vector of a test context. **Sim-targetPhrase** is similar but uses an average topic vector over all phrase pairs with the same target phrase. **Sim-targetWord** instead replaces the phrase pair topic vector with the word topic vector of the word in the target phrase with the lowest topical entropy\(^3\). Target word topic vectors are derived from phrase pair topic vectors by averaging over all vectors of phrase pairs that include the target word. The features **Conditional**, **Target-unigrams** and

\(^1\)The baseline model here corresponds to the relative frequency of target unigrams in the training data.

\(^2\)Topic 0 has higher a priori probability and is supposed to capture common words that occur in the context of many translation units (Hasler et al., 2014b).

\(^3\)The intuition behind this feature is that words with low topical entropy are expected to be more topically relevant.
Sim-phrasePair are similar or equivalent to features described in Hasler et al. (2014a,b), the remaining three features are new.

While the probabilistic features have some notion of the frequency of translations in the training corpus, similarity features are purely based on topic information and could be unreliable for rare translation units. On the other hand, similarity features are more efficient to compute at test time than the conditional translation probability which requires summation and normalisation for each phrase pair.

For the adaptation experiments, we evaluate a topic feature set that contains all the features described above, as well as smaller subsets thereof. The large feature set overlaps with the unadapted and domain-adapted features sets in that each contains probabilistic translation features. The smaller sets only contain features that have no direct correspondence in the baseline models. The topic feature sets are defined as:

Overlap Conditional, Joint-conditional, Target-unigrams, Sim-phrasePair, Sim-targetPhrase, Sim-targetWord

Sim-combine similarity = \frac{1}{3} (Sim-phrasePair + Sim-targetPhrase + Sim-targetWord)

Sim-combine-loglin Sim-phrasePair, Sim-targetPhrase, Sim-targetWord

Sim-combine+trgUnigrams Sim-combine, Target-unigrams

4 Data

Our experiments were carried out on a French-English data set consisting of the TED corpus (Cettolo M. and Federico, 2012), parts of the News Commentary corpus (NC) and parts of the Commoncrawl corpus (CC) from the WMT13 shared task (Bojar et al., 2013) as described in Table 1, condition 1. To ensure that the baseline model does not have an implicit preference for any particular domain, we selected subsets of the NC and CC corpora such that the training data contains 2.7M English words per domain. The data set simulates an environment where very diverse documents have to be translated, which is a typical scenario for web translation engines, for example.

We evaluate our models on a second training set (condition 2) where we add the Europarl corpus to the translation and language model training data. This increases the number of parallel training sentences to 2.3M. For condition 2, the phrase pair contexts for topic modelling are extracted from a much larger number of sentence pairs, therefore we sample up to 50 contexts per phrase pair to keep the training size manageable. We also do not learn topic vectors for singleton phrase pairs or phrase pairs that occur more than 20000 times in the training data, as we expect that such pairs are less dependent on the context.

<table>
<thead>
<tr>
<th>Data</th>
<th>Mixed</th>
<th>CC</th>
<th>NC</th>
<th>TED</th>
<th>Europarl</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train (condition 1)</td>
<td>354K (6450)</td>
<td>110K</td>
<td>103K</td>
<td>140K</td>
<td>-</td>
</tr>
<tr>
<td>Train (condition 2)</td>
<td>2.3M</td>
<td>110K</td>
<td>103K</td>
<td>140K</td>
<td>1.9M</td>
</tr>
<tr>
<td>Dev</td>
<td>2453 (39)</td>
<td>818</td>
<td>817</td>
<td>818</td>
<td>-</td>
</tr>
<tr>
<td>Test</td>
<td>5664 (112)</td>
<td>1892</td>
<td>1878</td>
<td>1894</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 1: Number of sentence pairs and documents (in brackets) in the data sets.

5 Predicting domain labels

While previous approaches to automatic domain classification have built dedicated classifiers such as SVMs and perceptrons or used in-domain language model perplexity, we use our
trained topic model to assign domain labels to documents. We apply the PPT model to all
documents from the training domains of condition 1 (C, NC, TED) to get one topic vector per
training document. We then experiment with three types of classifiers using the induced topic
vectors:

**Single-prototype** Compute the average of all document vectors of the same training domain
(→ domain vectors), then compute the cosine similarity of a test document with the three
domain vectors and predict the domain with the highest similarity.

**Multi-prototype** Compute the cosine similarity of a test document with the topic vectors of all
training documents and predict the domain according to the label of the most similar training
document.

**Single-prototype-threshold** Like single-prototype but with a threshold of 0.35 for prediction5.
If a test document is not similar to any of the domain vectors according to the threshold, predict
unknown and use the baseline model in place of a domain-adapted model.

The results of the single- and multi-prototype classifiers on the development and test doc-
ments are shown in Table 2. While for NC and TED documents, we can get perfect domain
predictions with the single-prototype classifier, the accuracy on CC is at most 0.82, depending
on the number of latent topics in the topic vectors. However, the multi-prototype classifier does
better for CC in all cases. This suggests that there are subclusters of documents in the CC cor-
pus to which some of the CC test documents are similar while not being as similar to a global
average of all CC documents. Table 3 shows the accuracy of the single-prototype classifier when
using a fixed threshold, with the results split into correct, other and unknown. While NC and
TED documents are still labelled accurately, the proportion of correct predictions drops for CC.
This confirms that NC and TED can be considered domains in the sense that the documents all
have certain properties in common, while this is not the case for CC. This is also supported
by Figure 1 which shows the average domain vectors for each of the three corpora, with some
of the topical peaks labelled according to their most likely words. While CC documents can
belong to rather diverse clusters such as IT, arts, hotel reviews or speech, NC documents belong
to more related topics along the themes of politics and economy. These topics are more likely to
be active within the same document and thus a document with political or economical content
would likely overlap with the NC domain vector on several dimensions. TED documents share
two topical components that capture words that are typical in speech like 1st and 2nd person
verb forms (speech) as well as a rather broad science topic. Thus, a document with a high
proportion of these verb forms would be likely to be classified as TED.

<table>
<thead>
<tr>
<th>Model</th>
<th>Cc</th>
<th>NC</th>
<th>TED</th>
</tr>
</thead>
<tbody>
<tr>
<td># dev+test docs</td>
<td>88</td>
<td>39</td>
<td>24</td>
</tr>
<tr>
<td>k=10</td>
<td>sgl</td>
<td>mlt</td>
<td>sgl</td>
</tr>
<tr>
<td></td>
<td>0.70</td>
<td>0.88</td>
<td>1.0</td>
</tr>
<tr>
<td>k=20</td>
<td>0.82</td>
<td>0.94</td>
<td>1.0</td>
</tr>
<tr>
<td>k=50</td>
<td>0.73</td>
<td>0.93</td>
<td>1.0</td>
</tr>
<tr>
<td>k=100</td>
<td>0.76</td>
<td>0.93</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Table 2: Accuracy of domain prediction using single-prototype (sgl) or multi-prototype (mlt)
domain vectors with different numbers of topics (k).

5Cosine similarity ranges from 0 to 1. The threshold was set on the development set.
We further observe in Table 3 that the rate of predicting unknown for CC documents increases with the number of topics. The reason for this is that we use the same classification threshold for all $k$ while the cosine similarities of higher-dimensional topic vectors are typically lower than those of low-dimensional vectors$^6$. For the experiments in the next section, we used the single-prototype-threshold classifier with $k = 50$.

<table>
<thead>
<tr>
<th>Model</th>
<th>CC</th>
<th>NC</th>
<th>TED</th>
</tr>
</thead>
<tbody>
<tr>
<td># dev+test docs</td>
<td>88</td>
<td>39</td>
<td>24</td>
</tr>
<tr>
<td>k=10</td>
<td>corr</td>
<td>other</td>
<td>unk</td>
</tr>
<tr>
<td>0.68</td>
<td>0.30</td>
<td>0.02</td>
<td>1.0</td>
</tr>
<tr>
<td>k=20</td>
<td>0.76</td>
<td>0.15</td>
<td>0.09</td>
</tr>
<tr>
<td>k=50</td>
<td>0.60</td>
<td>0.19</td>
<td>0.21</td>
</tr>
<tr>
<td>k=100</td>
<td>0.55</td>
<td>0.12</td>
<td>0.33</td>
</tr>
</tbody>
</table>

Table 3: Accuracy of domain prediction using single-prototype vectors with a threshold of 0.35 and different numbers of topics ($k$). $Corr$: correct domain predicted, $other$: wrong domain predicted, $unk$: no domain predicted.

### 6 Experimental setup

All of the test corpora contain document boundaries which allows us to consider document context during translation and switch translation and language models at document boundaries. While the domain-adapted baselines use gold domain labels, we use automatically predicted domains when combining domain-adapted and topic-adapted models$^7$. We use a tuning set containing data from all three test domains and tune a single set of feature weights for all portions of the test set. Translation quality is evaluated using the average feature weights of three optimisation runs with PyO (Hopkins and May, 2011). We use the mteval-v13a.pl script to compute case-insensitive BLEU scores and use bootstrap resampling (Koehn, 2004) to measure significance of the BLEU scores on the mixed test set.

---

$^6$This trend was observed by Banchs and Costa-jussà (2011) for vectors derived from Latent Semantic Indexing.

$^7$Note that topic adaptation does not rely on domain labels.
6.1 Unadapted baseline system
Our baseline is a phrase-based French-English system trained on the concatenation of all parallel data for condition 1 and 2, respectively. It was built with the Moses toolkit (Koehn et al., 2007) using the 14 standard core features including a 5-gram language model, trained on the concatenation of the target sides of the training data.

6.2 Domain-adapted systems
We use the linear mixture model (DA-TM) of Sennrich (2012) (available in the Moses toolkit) to adapt the translation model to each of the three test domains CC, NC and TED. The domain labels of the documents are used to group documents of the same domain together. We build adapted tables for each domain by treating the remaining documents as out-of-domain data. For development and test, the domain labels are used to select the respective adapted model for decoding. We also use domain-adapted language models (DA-LM) which are linear interpolations of separate language models for each training domain, tuned to minimise perplexity on an in-domain development set per domain.

6.3 Topic-adapted systems
In order to integrate document-specific features into decoding, we build a (filtered) phrase table with topic-adapted features for each test document which is loaded before decoding each document. It would be straightforward to achieve a tighter integration with the SMT system by setting up feature functions that have access to document-level information, but for now we use a simple architecture where a wrapper script runs the decoder for each document.

7 Results
In this section we present results of different combinations of the baseline model, the domain-adapted and the topic-adapted models. Results are reported separately per test domain as well as on the entire mixed test set. We first describe results for training condition 1 in Sections 7.1 and 7.2, before showing results on training condition 2 in Section 7.3. We also provide qualitative evaluation of translation outputs in Section 7.4.

7.1 Overlapping topic feature set
Table 4 shows the results when adding the overlapping topic feature set (containing probabilistic and non-probabilistic translation features) on top of unadapted and domain-adapted systems. Adding topic-adapted features always yields improvements over the respective baseline system, even though the amount of previous adaptation has an influence on the relative gain. Topic adaptation works best for TED documents but we observe that the improvement decreases with increasing domain-adaptation. Depending on the amount of previous adaptation, the BLEU improvements range between 1.44 and 0.31. These results add to our observations from Section 5 that on top of showing the characteristics of a domain, TED documents exhibit a further layer of structure that can be exploited with topic adaptation. For CC, the improvement of topic adaptation is quite stable at between 0.6 and 0.7 BLEU because domain adaptation has almost no effect on performance here. This is in line with our observations from Section 5 that CC behaves least like a domain in comparison with the other test corpora. For NC documents, the topic-adapted features yield a small improvement of 0.24 BLEU over the unadapted system but no further improvement over the domain-adapted models. A possible explanation is that because of the close relation between the dominant topics in the NC corpus (politics/economy), domain adaptation methods are sufficient to capture the important characteristics of the documents.
Table 4: BLEU scores of unadapted/adapted baseline models (training condition 1) and additional topic-adapted features (Overlap) with their gain over the respective baseline (bottom of each block). The best system on the mixed test set is marked in bold. *: $p \leq 0.001$ marks significantly better scores compared to the respective baseline.

<table>
<thead>
<tr>
<th>Model</th>
<th>Mixed</th>
<th>CC</th>
<th>NC</th>
<th>TED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>26.86</td>
<td>19.61</td>
<td>29.42</td>
<td>31.88</td>
</tr>
<tr>
<td>+ topics</td>
<td>*27.57</td>
<td>20.35</td>
<td>29.68</td>
<td>33.22</td>
</tr>
<tr>
<td></td>
<td>+0.71</td>
<td>+0.74</td>
<td>+0.26</td>
<td>+1.34</td>
</tr>
<tr>
<td>DA-TM</td>
<td>27.24</td>
<td>19.61</td>
<td>29.87</td>
<td>32.73</td>
</tr>
<tr>
<td>+ topics</td>
<td>*27.73</td>
<td>20.33</td>
<td>29.88</td>
<td>33.55</td>
</tr>
<tr>
<td></td>
<td>+0.49</td>
<td>+0.69</td>
<td>+0.01</td>
<td>+0.82</td>
</tr>
<tr>
<td>DA-LM</td>
<td>27.16</td>
<td>19.71</td>
<td>29.77</td>
<td>32.46</td>
</tr>
<tr>
<td>+ topics</td>
<td>*27.60</td>
<td>20.37</td>
<td>29.80</td>
<td>33.20</td>
</tr>
<tr>
<td></td>
<td>+0.44</td>
<td>+0.63</td>
<td>+0.03</td>
<td>+0.74</td>
</tr>
<tr>
<td>DA-TM+LM</td>
<td>27.34</td>
<td>19.59</td>
<td>29.92</td>
<td>33.02</td>
</tr>
<tr>
<td>+ topics</td>
<td>*27.63</td>
<td>20.22</td>
<td>29.90</td>
<td>33.33</td>
</tr>
<tr>
<td></td>
<td>+0.29</td>
<td>+0.60</td>
<td>-0.02</td>
<td>+0.31</td>
</tr>
<tr>
<td>Gain of best system over baseline</td>
<td>+0.87</td>
<td>+0.72</td>
<td>+0.46</td>
<td>+1.67</td>
</tr>
</tbody>
</table>

Overall, the best results on the mixed test set are achieved with a combination of domain and topic adaptation of the translation model (DA-TM + topics). This system yields a 0.82 BLEU improvement over the DA-TM model and a 1.67 improvement BLEU over the unadapted baseline, on TED documents. On the mixed test set, the gain over the DA-TM model is 0.49 and the overall gain is 0.87 BLEU.

7.2 Smaller topic feature sets

While the results from the previous section show that topic adaptation is beneficial at all levels of domain adaptation as long as the test documents are “topic-adaptable”, the role of domain adaptation is not that clear yet as the difference between the best topic-adapted system with and without domain-adapted features is relatively small (27.73 vs. 27.57 BLEU on the mixed test set). Therefore, we study the effect of adding domain-adapted features to already topic-adapted systems with smaller topic feature sets, thereby avoiding feature overlap between the systems. Another goal is to measure the contribution of particular topic features and find the best feature combination.

Table 5 shows the results when adding the domain-adapted features to the topic feature sets that do not contain probabilistic features. The upper part of the table shows the performance with single topic features, the lower part shows combinations of two or three topic features. In all experiments, the topic features improve over the unadapted baseline and the additional domain-adapted features improve over the topic-adapted model. Among the single topic features, the Sim-phrasePair feature yields the best performance on the mixed test set (27.32) and this trend persists when adding the domain-adapted features (27.53).

However, the best overall performance is achieved with the Sim-combine feature in combination with the domain-adapted features. For this setup, both adaptation methods yield a gain of ~0.4 BLEU on the mixed test set, adding up to a total gain of 0.83 as shown at the bottom of the table. The performance of this model on the mixed test set is close to the performance of the best model in Table 4, which indicates that we can achieve good performance with a small set
of topic-adapted features that encode information not captured by the domain-adapted features. As topic adaptation requires dynamic computation at test time, an architecture where part of the adaptation is done offline, as is the case for domain adaptation, could reduce the computational effort at test time.

### 7.3 Results for training condition 2

In this section, we evaluate the topic modelling approach on the same test set but with a larger amount of training data for the translation model, language model and topic model (condition 2 in Table 1). The results are shown in Table 6. First we note that the baseline system yields lower overall performance after the addition of the Europarl corpus. This is mostly due to a big hit in performance on the TED test set\(^8\). The domain-adapted models are able to balance the additional data and the combination of phrase table and language model domain adaptation yields an improvement of \(\sim 2\) BLEU on the mixed test set compared to the baseline. As for training condition 1, adding topic-adapted features always improves the performance, depending on the amount of previous adaptation. This can be seen most clearly on the TED test set where topic adaptation yields an improvement of 2.15 BLEU over the unadapted baseline, improvements of 1.10 and 0.91 over domain adaptation of the translation or language model, and an improvement of 0.28 over the domain-adapted model with both translation and language model adaptation.

The best overall performance is achieved with a combination of all three adaptation methods, as marked in bold in Table 6. While for CC and NC, the performance of the larger model

---

\(^8\)The performance on TED in comparison to training condition 1 drops to 30.01 when adding Europarl data to the translation model and to 30.61 when adding Europarl data to the language model, respectively.

---

<table>
<thead>
<tr>
<th>Model</th>
<th>Mixed</th>
<th>CC</th>
<th>NC</th>
<th>TED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>26.86</td>
<td>19.61</td>
<td>29.42</td>
<td>31.88</td>
</tr>
<tr>
<td>+ TrgUnigrams</td>
<td>27.04</td>
<td>19.86</td>
<td>29.25</td>
<td>31.57</td>
</tr>
<tr>
<td>+ DA-TM</td>
<td>**27.50</td>
<td>19.96</td>
<td>29.77</td>
<td>33.34</td>
</tr>
<tr>
<td>+ Sim-phrasePair</td>
<td>27.32</td>
<td>20.19</td>
<td>29.31</td>
<td>32.66</td>
</tr>
<tr>
<td>+ DA-TM</td>
<td>27.53</td>
<td>20.04</td>
<td>30.05</td>
<td>32.98</td>
</tr>
<tr>
<td>+ Sim-targetPhrase</td>
<td>27.21</td>
<td>19.92</td>
<td>29.39</td>
<td>32.58</td>
</tr>
<tr>
<td>+ DA-TM</td>
<td>**27.52</td>
<td>19.96</td>
<td>29.94</td>
<td>33.20</td>
</tr>
<tr>
<td>+ Sim-targetWord</td>
<td>26.99</td>
<td>19.89</td>
<td>29.16</td>
<td>32.12</td>
</tr>
<tr>
<td>+ DA-TM</td>
<td>**27.44</td>
<td>19.91</td>
<td>29.98</td>
<td>32.94</td>
</tr>
<tr>
<td>+ Sim-combine</td>
<td>27.29</td>
<td>20.10</td>
<td>29.49</td>
<td>32.60</td>
</tr>
<tr>
<td>+ DA-TM</td>
<td>**27.69</td>
<td>**20.13</td>
<td>**29.90</td>
<td>**33.37</td>
</tr>
<tr>
<td>+ Sim-combine-loglin</td>
<td>27.18</td>
<td>20.13</td>
<td>29.55</td>
<td>32.34</td>
</tr>
<tr>
<td>+ DA-TM</td>
<td>*27.41</td>
<td>19.93</td>
<td>29.86</td>
<td>32.97</td>
</tr>
<tr>
<td>+ Sim-combine+trgUnigrams</td>
<td>27.21</td>
<td>20.05</td>
<td>29.36</td>
<td>32.78</td>
</tr>
<tr>
<td>+ DA-TM</td>
<td>**27.47</td>
<td>19.87</td>
<td>29.76</td>
<td>33.36</td>
</tr>
<tr>
<td>DA gain of best system</td>
<td>+0.40</td>
<td>+0.03</td>
<td>+0.41</td>
<td>+0.77</td>
</tr>
<tr>
<td>Gain of best system over baseline</td>
<td>+0.83</td>
<td>+0.52</td>
<td>+0.48</td>
<td>+1.49</td>
</tr>
</tbody>
</table>

Table 5: BLEU scores of smaller topic feature sets with added domain-adapted features (training condition 1). The best system on the mixed test set is marked in bold and its improvements over the topic-adapted system and the baseline are shown at the bottom of the table. **: \(p \leq 0.01\), *: \(p \leq 0.05\) mark significant improvements over the topic-adapted systems.
### Table 6: BLEU scores of unadapted/adapted baseline models (training condition 2) and additional topic-adapted features (Overlap) with their gain over the respective baseline (bottom of each block). The best system on the mixed test set is marked in bold. **: $p \leq 0.01$, *: $p \leq 0.05$ mark significantly better scores compared to the respective baseline.

<table>
<thead>
<tr>
<th>Model</th>
<th>Mixed</th>
<th>Cc</th>
<th>NC</th>
<th>TED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline + topics</td>
<td>25.74</td>
<td>20.01</td>
<td>29.01</td>
<td>27.82</td>
</tr>
<tr>
<td>DA-TM + topics</td>
<td><strong>26.54</strong></td>
<td>20.13</td>
<td>29.53</td>
<td>30.86</td>
</tr>
<tr>
<td>DA-LM + topics</td>
<td>27.01</td>
<td>20.26</td>
<td>30.48</td>
<td>30.43</td>
</tr>
<tr>
<td>DA-TM+LM + topics</td>
<td><strong>27.91</strong></td>
<td>20.10</td>
<td>30.68</td>
<td>32.70</td>
</tr>
<tr>
<td>Total gain over baseline</td>
<td>+2.17</td>
<td>+0.37</td>
<td>+1.79</td>
<td>+5.16</td>
</tr>
</tbody>
</table>

is equal or better than the best model in Table 4, the performance on TED falls short of that model by $\sim 0.6$ BLEU. This is likely due to the fact that adding Europarl data is particularly harmful for translating TED documents. Therefore, in future work we will look at combining the adaptation approaches studied here with data selection methods such as the work of Axelrod et al. (2011).

### 7.4 Qualitative evaluation

In this section, we analyse some concrete output examples that visualise the differences in the translations produced by the different models for training condition 1. Figure 2 shows two input and reference sentences with their translations under the unadapted baseline, the domain-adapted model and the model with both domain-adapted and topic-adapted features. In the first example, the baseline system does not translate the source verb *remontent* appropriately. This is fixed by the domain-adapted model and in addition, the topic-adapted model finds a contextually better translation that matches the reference. In the second example, the domain-adapted model fixes the wrong lexical choice of the baseline model and the topic-adapted model maintains the same translation. Thus, these are examples where domain adaptation is doing most of the adaptation work.

Figure 3 shows two examples where all models make different lexical choices and only the addition of the topic-adapted model yields the correct lexical selection. In these examples, both the baseline and the domain-adapted model choose a translation that corresponds to a different sense of the French source word *(speed/bitrate/throughput)*, while the topic-adapted model selects a translation capturing the same sense as the reference translation *(flow/flows)*.

The example in Figure 4 shows an incremental improvement from the unadapted model to the domain-adapted model and the topic-adapted model. Here, the domain-adapted model improves slightly over the baseline model by producing a more fluent translation. However, the underlined segments are still translated incorrectly, for example *historique de recherche*.

---

9 The outputs correspond to the models in the first line and the second block of Table 4.
elles représentent les étendues de l’imagination humaine qui remontent à l’aube du temps.

they represent the bodies of the human imagination that date back to the dawn of time.

they represent the bodies of the human imagination that go back to the dawn of time.

they represent branches of the human imagination that go back to the dawn of time.

ils l’ont fait en drainant les terres.

they did in drawing the land.

draining the land.

they did in draining the land.

draining the land.

they did it by draining the land.

draining the land.

le débit est en augmentation très rapide. le débit a augmenté.

the speed is growing very rapidly. the bitrate has increased.

the throughput is rising very fast. the throughput has increased.

the flow is growing very rapidly. the flow has increased.

flows are increasing very rapidly. the flows have increased.

and, si je veux m’éloigner et tout regarder je peux découper mon historique peut-être mon historique de recherche.

and, if i want to move me and look at everything i can go into my historical historic perhaps my research.

and, if i want to get away from and look at everything i can go into my maybe historical record of my research.

and, if i want to get away from it and look at everything i can go into my history can be my search history.

and, if i want to step back and look at everything, i can slice and dice my history perhaps by my search history.

et, si je veux m’éloigner et tout regarder je peux découper mon historique peut-être mon historique de recherche.

and, if i want to move me and look at everything i can go into my historical historic perhaps my research.

and, if i want to get away from and look at everything i can go into my maybe historical record of my research.

and, if i want to get away from it and look at everything i can go into my history can be my search history.

and, if i want to step back and look at everything, i can slice and dice my history perhaps by my search history.

is translated as record of my research. The topic-adapted model fixes the translations of the underlined segments and finds the correct translations history and search history.

These examples show that domain and topic adaptation both contribute to the improved translation quality and that depending on the input example, the contribution of one of the two models may be more important. While we cannot draw any definite conclusions about the kind
of improvement each model makes, there seems to be a tendency that the topic-adapted model contributes more to improved lexical choice. We assume that the difference between domain and topic adaptation lies in the granularity of the modelled distributions over translations rather than a clearly defined difference in the level of adaptation, such as style or genre versus topic. However, this would mean that combining models of different granularity implicitly accounts for these levels of adaptation.

8 Conclusion

We have presented an empirical study on the effect of combining domain adaptation and topic adaptation within the same translation system. We have measured the relative benefit of both types of adaptation on a diverse set of test documents and found that the two approaches can be complementary depending on the text type and the amount of overlap between their feature sets. We have shown that the improvements gained by our topic modelling approach hold for domain-adapted models with smaller or larger amounts of training data and are particularly prominent when the training and test domains diverge strongly. We have further shown that the domain of a test document can be predicted accurately by using trained topic models to build domain vector prototypes. Combining domain adaptation, topic adaptation and automatic domain prediction is useful when translating documents from unknown origin and could also help to reduce the load of test time computations while still benefitting from dynamic topic adaptation. Our best combined model yields BLEU improvements of up to 1.67 over an unadapted baseline system and 0.82 over a domain-adapted system, measured on the training condition that yields the stronger baseline system.

Acknowledgements

This work was supported by funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement 287658 (EU BRIDGE). We thank Phil Blunsom and the anonymous reviewers for helpful comments.

References


Abstract

In this paper we investigate the problem of adapting a machine translation system to the feedback provided by multiple post-editors. It is well known that translators might have very different post-editing styles and that this variability hinders the application of online learning methods, which indeed assume a homogeneous source of adaptation data. We hence propose multi-task learning to leverage bias information from each single post-editors in order to constrain the evolution of the SMT system. A new framework for significance testing with sentence level metrics is described which shows that Multi-Task learning approaches outperforms existing online learning approaches, with significant gains of 1.24 and 1.88 TER score over a strong online adaptive baseline, on a test set of post-edits produced by four translators texts and on a popular benchmark with multiple references, respectively.

1 Introduction

In a professional localization environment, a document is post-edited by several professional translators with assistance of tools such as translation memory, dictionary, spell checkers etc. To speed up the process, lately localization companies have started using computer assisted translation (CAT) tools with statistical machine translation (SMT) systems in the backend. The role played by the SMT engine is to provide a translation hypothesis that the translator can post edit to produce high quality translations (Federico et al., 2012).

In recent works on online adaptation by Mathur et al. (2013) and Denkowski et al. (2014), the SMT is fed with the post edited sentence, allowing the models to adapt to the corrections made by the translators. These kind of systems works well if the document is being post edited by a single translator because models can adapt to the style of that translator. Problems arise when a document is being post edited by a group of translators which is usually the case with big size documents. In fact, if the SMT system adapts to the corrections of all translators together, it will likely mix or overlap stylistic features of the post-editors and thus not learn to mimic well any of them. On the other side, if the system adapts to each individual post-editor, then clearly
useful feedback from other post-editors gets wasted.

The main motivation for adapting a SMT system in the backend of CAT tool is that the translation improves over time since fewer mistakes are made after learning from post editions. For example, translator A does not post edit phrase *Fibre Channel* because he thinks that the phrase is a named entity, while B post edits *Fibre Channel → Canale a fibre* because he does not recognize it as a named entity. In the backend the SMT system first updates the model such that it keeps the named entity intact but after second post-edition the system adapt the model to translate *Fibre Channel → Canale a fibre*. Now, if the system receives again the input *Fibre Channel*, it will prefer to output *Canale a fibre* which will be an incorrect suggestion for translator A. This repetition of translation error slows down the process of post-editing which is completely opposite to the idea of using SMT system in the background.

In this paper, we aim at building a SMT system which can solve this dilemma of contrasting updates. A localization company would expect the SMT system to incorporate these updates and improve the translation quality with time. To do so, we propose using multi-task learning (henceforth MTL) (Caruana, 1993) in machine translation systems. Here, we can consider the translators as different tasks and their post edits as an incoming stream of data the system wants to adapt to. Moreover, this system also maintains a prior relationship between the translators, according to the framework specified in multi-task learning.

The paper is structured as follows. First, we describe previous work on using online learning algorithm in CAT scenario and the generic online multi-task learning algorithm developed by Cavallanti et al. (2010). Then, Section 4 describes the online multi-task learning algorithm which can be applied in CAT scenario. Experiments and results are shown in Section 5. We conclude the paper with a preview of interesting related works and few words about the future work.

### 2 Background: Online Large Margin Training

Previous work by Mathur et al. (2013) applies an online large margin algorithm (MIRA), that updates the weights \( w \) of a phrase-based SMT model according to the loss that is occurred due to an incorrect translation. The margin is coupled with the following loss function based on the complement of the sentence level BLEU (BLEU+1, henceforth sBLEU) (Lin and Och, 2004; Nakov et al., 2012):

\[
l_j = s\text{BLEU}(y^*) - s\text{BLEU}(y_j)
\]

where \( y^* \) is the oracle (closest translation to the reference) and \( y_j \) is the \( j \)-th candidate being processed inside an \( N \)-best list. According to (Watanabe et al., 2007), weights are updated so that the loss is not larger than the difference between the scores given by the model:

\[
l_j \leq w^T \Delta h_j
\]

where \( \Delta h_j \) is the difference between the feature vectors of the oracle and the candidate, and \( w \) is the weight vector. Hence, the size of the weight update is:

\[
\arg \min_w ||w - w'|| + C \sum_j \xi_j
\]

subject to

\[
w^T \Delta h_j + \xi_j \geq l_j
\]

\[
\xi_j \geq 0 \quad \forall j \in \{1 \ldots N\}
\]
$C$ is an aggressiveness parameter which controls the size of the update and $\xi$ are slack variables. Following (Watanabe et al., 2007), the Lagrangian dual form of criterion (3) can be derived:

$$\max_{\alpha(\cdot)\geq 0} -\frac{1}{2} \| \sum_j \alpha_j \cdot \Delta h_j \|^2 + \sum_j \alpha_j l_j - \sum_j \alpha_j w^T \Delta h_j$$

subject to

$$\sum_j \alpha_j \leq C$$

which leads to a quadratic programming problem and to the weight vector update:

$$w = w' + \sum_j \alpha_j \cdot \Delta h_j.$$  

We determine the lagrangian multipliers $\alpha_j$ at each iteration by applying a QP-solver based on gradient descent.

3 Online Multi-Task Learning

In online multi-task learning (henceforth OMTL) (Cavallanti et al., 2010), training is done jointly on $k$ tasks so as to improve generalization capability for all tasks. Here, the task can be either binary classification or linear regression. The overall goal of OMTL is to learn the $k$ weight vectors simultaneously, one for each task in an online fashion.

The protocol for OMTL at each time $t$ is as follows:

1. receive an input pair $(x, s)$, where $x$ is the example and $s$ is the task id
2. predict the value $\hat{y} = w^T_s h_s(x)$, using current weights $w_s$ for task $s$
3. receive the correct label $y$
4. update all the $k$ weight vectors $w_s$ with $s = 1, \ldots, k$

OMTL is a matrix-based regularization approach described in details in Cavallanti et al. (2010). The update step in online learning is the standard Perceptron rule (Rosenblatt, 1958) with different learning rates for each task. These learning rates are defined in an interaction matrix which encodes the relatedness among the different tasks. The $\alpha_{s_1,s_2}$ element of the interaction matrix is the learning rate for task $s_1$ when task $s_2$ is being executed.

$$A^{-1} = \frac{1}{k+1} \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1k} \\ a_{21} & a_{22} & \cdots & a_{2k} \\ \cdots & \cdots & \cdots & \cdots \\ a_{k1} & a_{k2} & \cdots & a_{kk} \end{bmatrix}$$

with update rule for weight vector $w_s$ equal to:

$$w_s = w'_s + \hat{y} (A \otimes I_d)^{-1}_s H_s(x)$$

where $\otimes$ denotes the Kronecker product\(^1\) of the interaction matrix ($A^{-1}$) of dimension $k \times k$ and identity matrix ($I_d$) of dimensions $d \times d$, making a $kd \times kd$ matrix. The $kd \times kd$ matrix in the update rule co-regularizes the weight vector ($w_s$) by forcing the learner to account for the relatedness between the tasks. $H_s(x) = (\underbrace{0, \ldots, 0}_{(s-1)d \text{ times}}, h_s(x), \underbrace{0, \ldots, 0}_{(k-s) \text{ times}}) \in \mathbb{R}^{kd}$ with $d$ being the number of features.

\(^1\otimes\) shows mixed-product property, so one can calculate $A^{-1}$ and then compute the Kronecker product of $A^{-1} \otimes I_d^{-1}$.\(1\)
4 MIRA with multitasking

MIRA has been successfully applied to tune the log linear weights of SMT model in post editing scenario by Mathur et al. (2013). Here, we extend the online algorithm to fit the same scenario where input comes from $k$ different translators (tasks$^2$) and the learner has to predict the weights of all tasks simultaneously.

We modify Equation 5 by adding the matrix co-regularization factor of $(A \otimes I_d)^{-1}$ (from Equation 7), such that the difference of feature vector from $j^{th}$ candidate translation (i.e. $\Delta h_j$) affecting the change in weights for current task $s$ take into account the bias from each task. After substitution, our update rule becomes:

$$w_s = w'_s + \sum_j \alpha_j \cdot \langle \Delta h_{s,j} \rangle$$  
where

$$\langle \Delta h_{s,j} \rangle = (A \otimes I_d)^{-1} \cdot \Delta H_{s,j} \quad \text{and}$$

$$\Delta H_{s,j} = (\underbrace{0, \ldots, 0}_{(s-1)d \text{ times}}, \Delta h_{s,j}, \underbrace{0, \ldots, 0}_{(k-s)d \text{ times}})$$ (8)

Here, $\Delta H_{s,j}$ is a compound row vector for candidate translation $j$ of size $kd$ with $d$ being the size of the standard log linear features used in SMT$^3$. $(A \otimes I_d)^{-1}$ is the co-regularization factor of $kd \times kd$ dimensions. $A^{-1}$ as seen from Equation 6 defines the task relatedness. In CAT scenario we can see the interaction matrix as the matrix which defines relatedness between different translators. This relatedness can be captured by finding a correlation between the translators on their previous post-editions of a given dataset. The similar their post-editions on a particular dataset (with that the machine translation suggestion coming from one SMT system) the more is the relatedness between the translator. In Section 5.2, we show a way to compute the interaction matrix.

5 Experiments and Results

5.1 Data

We evaluated our method on three translation tasks defined over three different domains, namely Information Technology (IT), Travel domain (BTEC) and Legal domain.

The IT test set involves the translation of technical documents from English into Italian and has been used in the field test carried out under the MateCat$^4$ project. It has been translated by four translators, i.e. four different translations of the source document are available.

BTEC is a publicly available corpus in the travel domain, proposed as translation task in the IWSLT evaluation campaigns up to 2010. In addition to its availability, BTEC is of interest for us because the test set contains six human references, allowing to simulate the multi-task scenario.

Legal domain data has been release as a part of JRC-acquis corpus (Steinberger et al., 2006). The dataset contains translation of legal documents from English to Italian. This dataset was also a part of the field test carried out under the same MateCat project, so essentially we have post-edited data from 4 different translators on a test set of 90 sentences.

Since our methods regard the adaptation of MT models, the potential impact strictly depends on how much the considered text is repetitive. For measuring that text feature, we use the repetition rate proposed by Bertoldi et al. (2013). Equation 9 shows the formula for calculating the repetition rate of a document, where $\text{dict}(n)$ represents the total number of different

---

$^2$In this paper we use the terms tasks and translators interchangeably as the tasks are translators in the CAT scenario.

$^3$To keep the notation light we again drop the dependency of $h$ from $x$.

$^4$http://www.matecat.com
\(n\)-grams and \(n_r\) is the number of different \(n\)-grams occurring exactly \(r\) times. Statistics of the parallel sets on source and target sides along with the repetition rates are reported in Table 1.

\[
RR = \left( \prod_{n=1}^{4} \frac{\sum_S \text{dict}(n) - n_1}{\sum_S \text{dict}(n)} \right)^{1/4}
\]

Table 1: Statistics of parallel data.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Set</th>
<th>#srcTok</th>
<th>SrcRR</th>
<th>#tgtTok</th>
<th>TgtRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>IT(\text{en}\rightarrow\text{it})</td>
<td>Train</td>
<td>57M</td>
<td>na</td>
<td>60M</td>
<td>na</td>
</tr>
<tr>
<td></td>
<td>Dev</td>
<td>3.3K</td>
<td>19.08</td>
<td>3.6K</td>
<td>18.01</td>
</tr>
<tr>
<td></td>
<td>Test</td>
<td>3K</td>
<td>31.32</td>
<td>3.3K</td>
<td>22.18</td>
</tr>
<tr>
<td>BTEC(\text{en}\rightarrow\text{it})</td>
<td>Train</td>
<td>0.14M</td>
<td>na</td>
<td>0.13M</td>
<td>na</td>
</tr>
<tr>
<td></td>
<td>Dev</td>
<td>2K</td>
<td>9.47</td>
<td>1.9K</td>
<td>6.73</td>
</tr>
<tr>
<td></td>
<td>Test</td>
<td>1.9K</td>
<td>12.5</td>
<td>1.8K</td>
<td>7.76</td>
</tr>
<tr>
<td>Legal(\text{en}\rightarrow\text{it})</td>
<td>Train</td>
<td>63M</td>
<td>na</td>
<td>65M</td>
<td>na</td>
</tr>
<tr>
<td></td>
<td>Dev</td>
<td>2.9K</td>
<td>14.37</td>
<td>3.2K</td>
<td>11.25</td>
</tr>
<tr>
<td></td>
<td>Test</td>
<td>2.7K</td>
<td>13.59</td>
<td>2.85K</td>
<td>12.00</td>
</tr>
</tbody>
</table>

Preparing Data for MTL  Since we have \(k\) translations for a source document, we shuffle the references/post-editions such that we have one source document and one target document with the sentences containing meta information for the translators who produced these translations. Table 2 shows a sample of source and target document from IT dataset. The figure reads: sentence #1 is translated by translator #0, then feedback (sentence #2) goes to the system with its post-edited translation, system performs multi-task learning and so on. If one removes the meta-information about the translator’s ID, the resulting development set is used for online learning (refer Section 2). If one also removes the feedback, then the development set is used for baseline system (refer Section 5.2).

This shuffling of data also impacts the repetition rate. In fact, the repetition rates on the target side of IT test set for each translator varied from 26.95 to 28.70, while the repetition rate on the shuffled target side is 22.18, as reported in Table 1: this could be due to the fact that translators tend to be not consistent among themselves, yielding less repetitions in each post-edited test set than in the shuffled test set.

Table 2: Excerpt from IT development set tagged with meta data.

<table>
<thead>
<tr>
<th>#Sentence</th>
<th>Sentence</th>
<th>OnlineLearning</th>
<th>Translator ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Input Date:_0</td>
<td>Not Activated</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>Input Date:_Data di input:_0</td>
<td>Activated</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>Evaluates conditionally:_1</td>
<td>Not Activated</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>Evaluates conditionally:_Valuta in modo condizionale:_1</td>
<td>Activated</td>
<td>1</td>
</tr>
</tbody>
</table>
MERT (Och, 2003) implementation provided in the Moses toolkit. Performance is computed not with corpus level metrics but with sentence level metrics. We decided to do this to avoid a metric mismatch between the evaluation and actual optimization where the margin is calculated by the sentence level BLEU scores (refer to Section 2). Therefore, we computed sBLEU scores and sentence level TER (Snover et al., 2006) scores and reported their average over the whole documents. We call them avg-sBLEU and avg-sTER.

Calculating $A^{-1}$ matrix: Interaction matrix can be computed in different ways. It basically conveys the relatedness/correlation between the translators who are post-editing a particular document. Usually a localization company keeps a ranking of the hired translators with them; either we can use the ranking to exploit the relatedness between the translators or we can calculate their correlation based on a known previous post-edited data set. Here, we assume that the relatedness between the translators can be seen as the similarity between their post-edited segments given that the MT suggestions were from the same system for all translators. This assumption is quite intuitive.

To compute the similarity, we calculate sentence level TER scores between the MT suggestions and the post-edited segments. In the cases where we do not have post-edited MT suggestions, for example BTEC where only multiple references are available, we simulate the conditions of post-editing by using the SMT translations provided by our own baseline system as MT suggestions. Now, the relatedness can be seen as the correlation between the sentence wise TER scores. We compute the correlation using a widely accepted correlation metric, namely the Pearson correlation coefficient (henceforth $r$). Once it is calculated, we rescale these coefficients so that the values are between $[0,1]$, instead of $[-1,1]$ as given by $r$. We do this rescaling of correlations because matrix-based regularization is not able to handle the negative relatedness between the tasks. These values are computed on the corresponding development sets (which also contain post-edited segments from same translators) and are used to construct the $A^{-1}$ matrix. Since the $r$ is bi-directional, the interaction matrix is symmetric in nature. $r$ values between the translators for IT and BTEC datasets are shown in Tables 3 and 4 respectively.

<table>
<thead>
<tr>
<th>Translators</th>
<th>T1</th>
<th>T2</th>
<th>T3</th>
<th>T4</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>1</td>
<td>0.82</td>
<td>0.83</td>
<td>0.70</td>
</tr>
<tr>
<td>T2</td>
<td>0.82</td>
<td>1</td>
<td>0.86</td>
<td>0.79</td>
</tr>
<tr>
<td>T3</td>
<td>0.83</td>
<td>0.86</td>
<td>1</td>
<td>0.77</td>
</tr>
<tr>
<td>T4</td>
<td>0.70</td>
<td>0.79</td>
<td>0.77</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 3: Pearson correlation amongst translators on IT dataset.

<table>
<thead>
<tr>
<th>Translators</th>
<th>T1</th>
<th>T2</th>
<th>T3</th>
<th>T4</th>
<th>T5</th>
<th>T6</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>1</td>
<td>0.69</td>
<td>0.68</td>
<td>0.92</td>
<td>0.96</td>
<td>0.97</td>
</tr>
<tr>
<td>T2</td>
<td>0.69</td>
<td>1</td>
<td>0.57</td>
<td>0.64</td>
<td>0.64</td>
<td>0.66</td>
</tr>
<tr>
<td>T3</td>
<td>0.68</td>
<td>0.57</td>
<td>1</td>
<td>0.71</td>
<td>0.66</td>
<td>0.67</td>
</tr>
<tr>
<td>T4</td>
<td>0.92</td>
<td>0.64</td>
<td>0.71</td>
<td>1</td>
<td>0.90</td>
<td>0.91</td>
</tr>
<tr>
<td>T5</td>
<td>0.96</td>
<td>0.64</td>
<td>0.66</td>
<td>0.90</td>
<td>1</td>
<td>0.98</td>
</tr>
<tr>
<td>T6</td>
<td>0.97</td>
<td>0.66</td>
<td>0.67</td>
<td>0.91</td>
<td>0.98</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4: Pearson correlation amongst translators on BTEC dataset. These correlations are computed on a simulated environment.

Now, we give a brief description of the various SMT systems involved in the experiments:

**Baseline:** SMT models are trained on the domain specific training data; log linear weights are tuned on shuffled development set without any feedback and meta data about translator’s ID.

**Online:** Feedback is added to the development set without the translator’s ID. First, log linear weights are tuned on this development data by means of MERT; then, keeping them fixed to the optimal values, additional hyper parameters (used in Online system) are tuned again on the development set by means of the Simplex algorithm (Nelder and Mead, 1965). This system contains a single weight vector for all the translators and is the same as explained in (Mathur...
et al., 2013).

**MTL-pearson:** Meta-information is added to the development set, and log linear weights are tuned on the dev set. There is an additional bias feature while using multi-task learning which is tuned using Simpex algorithm on the dev set. The elements of interaction matrix are the scaled $r$s. This system keeps track of $k$ different weight vectors for each translator.

**MTL-halfupdate:** The diagonal elements of the interaction matrix are set to 1, the off-diagonal elements to 0.5. This means that for every update in the current task $j \in 1 \ldots k$ we do a half-update to rest of the tasks. Note that this system does not need a development set to calculate the interaction matrix unlike MTL-pearson.

**K-independent:** The interaction matrix is set to be the identity matrix; it means that the tasks are independent of each other because no correlation is assumed between the translators. This system differs from Online system because here there is a separate instance of online learning for every translator, while in Online system there is a single instance of online learning for all the translators.

### 5.3 Results

Table 5 shows the avg-sTER$^5$ and avg-sBLEU scores over whole test set for all the systems. On the IT test set MTL-pearson shows gain of 1 avg-sBLEU points and 3.3 avg-sTER points over the Baseline system and 1.24 avg-sTER points over the strong Online system.

However, MTL-pearson does not perform well on BTEC test set, that is we are not able to capture well the task-relatedness in this scenario. Since the actual post-edit translations for BTEC are not available, we simulated them by generating MT suggestions from baseline system, which likely affects the effectiveness of the method. Nevertheless, MTL-halfupdate being a default system is able to capture quite well the correlation between the translators and significantly outperforms all the other systems. We can then conclude that if one does not have access to prior information about the translators for calculating the relatedness amongst them it is a good idea to back-off to use the default half-updates option.

On the Legal domain test set Multi-Task learning is not able to significantly improve over the online learning system. One reason for this could be the total number of sentences in the test set (90), that is each post-editor post edits only 22-25 sentences which is quite less in number as compared to other dataset where total number of sentences are 176 (IT) and 250 (BTEC) and hence each post-editor edits 44 and 42 sentences respectively. The other reason could be the relatively low repetition rate observed on the Legal test set.

<table>
<thead>
<tr>
<th>System</th>
<th>IT avg-sTER</th>
<th>IT avg-sBLEU</th>
<th>BTEC avg-sTER</th>
<th>BTEC avg-sBLEU</th>
<th>Legal avg-sTER</th>
<th>Legal avg-sBLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>46.91</td>
<td>38.28</td>
<td>42.76</td>
<td>46.69</td>
<td>39.44</td>
<td>41.09</td>
</tr>
<tr>
<td>Online</td>
<td>44.86</td>
<td>39.21</td>
<td>42.64</td>
<td>46.72</td>
<td>38.96</td>
<td>41.56</td>
</tr>
<tr>
<td>MTL-pearson</td>
<td><strong>43.62</strong></td>
<td><strong>39.27</strong></td>
<td><strong>41.76</strong></td>
<td><strong>47.17</strong></td>
<td><strong>38.93</strong></td>
<td><strong>41.58</strong></td>
</tr>
<tr>
<td>MTL-halfupdate</td>
<td>44.63</td>
<td>38.94</td>
<td><strong>40.76</strong></td>
<td><strong>47.71</strong></td>
<td>38.93</td>
<td>41.58</td>
</tr>
<tr>
<td>K-independent</td>
<td>46.55</td>
<td>38.04</td>
<td>42.25</td>
<td>47.05</td>
<td>38.93</td>
<td>41.55</td>
</tr>
</tbody>
</table>

Table 5: BLEU scores achieved by using different techniques of online learning. Best BLEU and TER scores are marked in bold fonts.

**Significance Testing:** Here, we employ a non-parametric multiple hypothesis testing framework such as Friedman tests. The strategy for significance testing is as follows:

$^5$It has been shown in the past by Snover et al. (2006) that in post-edit scenario TER has higher correlation than BLEU against the post-editing effort, and so we fix our primary metric to be avg-sTER.
1. We mark epochs at every 10% of test set i.e. \( t \) epochs at 10%, 20% .. 100%.

2. At every epoch we measure the average performance of the system in question i.e. calculate avg-sTER.

3. In the end we have avg-sTER scores of five different systems at \( t \) different epochs.

4. The average performance of the aforementioned methods on the epochs can be seen as multiple systems trying to solve multiple problems. To calculate the p-values of these multiple systems, we use Friedman test (Friedman, 1937).

5. Once p-values are calculated, we use a post-hoc Holm’s procedure (Holm, 1979) to check for the significance.

Results are reported in Table 6.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IT</td>
</tr>
<tr>
<td>MTL-pearson vs. Online</td>
<td>0.022°</td>
</tr>
<tr>
<td>MTL-halfupdate vs. Online</td>
<td>0.003°</td>
</tr>
<tr>
<td>K-Independent vs. Online</td>
<td>0.311</td>
</tr>
<tr>
<td>MTL-pearson vs. K-Independent</td>
<td>0.200</td>
</tr>
<tr>
<td>MTL-halfupdate vs. K-Independent</td>
<td>0.050</td>
</tr>
<tr>
<td>MTL-pearson vs. MTL-halfupdate</td>
<td>0.500</td>
</tr>
</tbody>
</table>

Table 6: p-values given by Friedman test. ° depicts a significant difference between the systems that are being compared.

We plotted the incremental avg-sTER scores over \( t \) different epochs on all test sets in Figure 1.

First of all, it is worth to compare the plots of MTL-pearson and of Online systems on IT test set, for which the improvement of over 3 avg-sTER points reported in Table 5 is significant (Table 6). In fact, the gap between the MTL-pearson system and the Online system is visible in the plot only after the 6th epoch, that is for 6 out of 10 epochs differences are not big enough; nevertheless; the difference is significant. MTL-halfupdate performs better than any other system at least on 6 out of 10 epochs, but even on all epochs with respect to the Online system: this is why it outperforms the Online at 95% of confidence interval. Interesting to note that MTL-halfupdate is the best performing system till 6 epochs; after that, MTL-pearson becomes the best one: this basically says that for the starting 60% of the data the translators had a correlation of half with each other, while later they were as coherent as they were when they post-edited the development set (because MTL-pearson correlation is calculated on development set). This also means that the relatedness between the translators is evolving even while post-editing the same dataset.

On BTEC test set, MTL-halfupdate consistently outperforms all other SMT systems on each epoch; this explains why it is significantly better than all other systems. On 9 epochs out of 10, MTL-pearson is better than the Online system; hence, the difference is significant. Results on BTEC put in evidence the importance of estimating a reliable interaction matrix to allow multi-task learning working at its best, but also that half-update is an effective back-off solution.

Significance tests on Legal test set\(^6\) shows that MTL-* systems are better than the Online

\(^6\)The error curve in Legal domain shows an apparently surprising increasing trend. This is due to the nature of the test set where the starting sentences are easier to translate than the later ones.
system with a p-value of 0.066.

Figure 1: Learning curve of different systems on IT (top left), BTEC (top right) and Legal (bottom) test sets.

So far, the evaluations were done on a shuffle of test set where the translators were assigned in sequence, i.e. first sentence to first translator, second to second and so on. This is usually not the case in a real world scenario, because a sentence can be assigned to any of the translators and not necessarily in a sequence. To replicate such scenario, we developed an assigning scheme through which each translator is assigned equal number of segments from a document to post-edit. The scheme is as follows:

1. For \( n \) translators, all possible permutations of the series 1...\( n \) is computed (total of \( n! \)).
2. The document to be post edited is divided in blocks of \( n \) sentences.
3. For each block we randomly pick a permutation series among the \( n! \) choices, and assign it to the block in question.

Following this scheme, we created 100 different shuffles of the IT test set which are closer to the real life setting. Similar to the learning curve we built before, we averaged out avg-sTER scores over 100 shuffles on sequential epochs i.e. (10%, 20%...100% of data). Figure 2 reports the learning curves of different adaptive systems over epochal data.

Here, unlike in the previous case, for each of 176 sentences we have 100 different sentence wise TER scores using 5 different systems. Since just the IT domain is considered, data are more homogeneous and then we could apply Approximate Randomization (Noreen, 1989), a statistical test that is well established in the NLP community (Chinchor et al., 1993). The test has been shown (Riezler and Maxwell, 2005) to be less prone to type-I errors than the bootstrap method (Efron and Tibshirani, 1993). We report the significance results in Table 7.

Even after the shuffling, we see that MTL-pearson system resistant to the shuffles and still performs significantly better than any other system. However, we observe a contradictory infor-
Figure 2: Learning curves of different systems on shuffled IT test set.

Table 7: p-values given by Approximate Randomization test. All the reported results in the table are significant.

<table>
<thead>
<tr>
<th>Systems Compared</th>
<th>P-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline vs. Online</td>
<td>0.001</td>
</tr>
<tr>
<td>Baseline vs. MTL-pearson</td>
<td>0.001</td>
</tr>
<tr>
<td>MTL-pearson vs. K-Independent</td>
<td>0.001</td>
</tr>
<tr>
<td>MTL-HalfUpdate vs. Online</td>
<td>0.04</td>
</tr>
<tr>
<td>MTL-pearson vs. MTL-HalfUpdate</td>
<td>0.001</td>
</tr>
<tr>
<td>Online vs. MTL-pearson</td>
<td>0.007</td>
</tr>
</tbody>
</table>

Table 8 shows an excerpt from the IT test set. The phrase backup in the source sentence (#21) is translated to copia di riserva by both K-Independent and MTL-pearson systems but the translator post-edits the phrase in both translation hypotheses to backup. Later, in sentence #23 the phrase appears again and this time Multi-Task correctly outputs the translation of the phrase backup to backup but K-independent system is not able to correct the mistake. Reiterating, K-Independent system runs a single instance of online learning for each of the post-editors. In the example the first sentence is post-edited by translator #3 and the latter by translator #1, thus, the system is not able to recognize the mistake committed for the translator #1 and consequently cannot correct it for translator #3. While the system MTL-pearson learns jointly over the corrections by all the translators and thus able to correct the translation hypothesis the next time.

Overall, the results show that Multi-Task learning outperforms the existing standard SMT and the strong online learning systems. If we have the meta information on the post-editors apriori i.e. their mutual correlation, we can boost the performance of the adaptive system. One can use the MTL-pearson system if the correlation matrix can be calculated accurately; if not, it is preferable to back-off to MTL-halfupdate system.
with minimal copying of data from the production volume to backup volume.

### Related Works

Despite several online adaptation strategies have been proposed in the past, only a few deal with adaptation of post-edited/evaluation data while most works are on adaptation over development data during tuning of parameters (Och and Ney, 2003).

Cesa-Bianchi et al. (2008) proposed an online learning approach during decoding. They construct a layer of online weights over the regular feature weights and update these weights at sentence level using margin infused relaxed algorithm (Crammer and Singer, 2003); to our knowledge, this is the first work on online adaptation during decoding. Martinez-Gomez et al. (2011, 2012) presented a comparison of online adaptation techniques in post editing scenario. They compared different adaptation strategies on feature weights and features itself.

Multi-Task learning has been explored in SMT in the context of tuning the sparse log linear weights by Simianer et al. (2012) where they split the training set in random shards and perform a joint feature selection over these shards using $\ell_1/\ell_2$ regularization. In this way after each epoch, the size of feature vector decreases and only the important features are taken into account. In our paper instead of $\ell_1/\ell_2$ regularization we have use a matrix-based regularization approach on the core features for online adaptation of all the translation models.

Multi-Task learning has also been used in re-ranking the N-best list by Duh et al. (2010). Each N-Best list is considered as a different task and the weights are jointly learnt over a large set of sparse features. Simianer et al. (2011) trained a discriminative model using multi-task learning over a set of $k$ documents belonging to different topics but with strong commonalities.

Recent application of multi-task learning has been in quality estimation for machine translation by Cohn and Specia (2013) where the authors model annotator bias using multi-task Gaussian processes. Their model outperforms the annotator specific model and thus boosting the use of Multi-Task learning in NLP applications. Another application of MTL has been in supervised domain adaptation for quality estimation (C. de Souza et al., 2014). In this work the authors leverage all available training labels from different domains in order to learn a robust model for a target domain with very little labeled data. The approach proposed outperforms independent models trained separately on each domain.

### Conclusion

We addressed the problem of adapting in a CAT framework a single SMT system to multiple post-editions, i.e. to an incoming stream of feedback from different translators. In such a situation, standard online learning methods can lead to incoherent translations by the SMT system. To the best of our knowledge, this kind of problem has never been addressed before for adapting SMT systems in CAT scenario. As a solution we propose to adopt a multi-task learning scheme, which relies on the correlation amongst the translators computed using prior knowledge; the online learner is then constrained to take into account the relatedness amongst
the translators.

Different online systems have been compared against each other, and online multi-task learning SMT system outperformed in most cases the strong online learning SMT system taken as baseline. Whenever not enough information about the correlation amongst the translators is available, our experimental outcomes suggest to use multi-task learning with half-updates, which is a good generalization of the interaction between the translators. We also compared the Multi-Task approach to the K-Independent system where each translator has been allotted an online learning SMT system; evidently, multi-task also fared better against this system setup. Moreover, MTL can also be applied to tune the log-linear weights of SMT models when multiple references are given.

In our approach, once the correlation matrix has been computed, it is kept fixed throughout the learning process. Instead, as evinced by our experiments, the interaction between translators can evolve over time; we plan to further investigate this aspect in the future.

Acknowledgements

This work was supported by the MateCat project, which is funded by the EC under the 7th Framework Programme.

References


The Repetition Rate of Text as a Predictor of the Effectiveness of Machine Translation Adaptation

Mauro Cettolo
cettolo@fbk.eu
Nicola Bertoldi
bertoldi@fbk.eu
Marcello Federico
federico@fbk.eu
FBK, Fondazione Bruno Kessler, 38123 Povo, Trento, Italy

Abstract

Since the effectiveness of MT adaptation relies on the text repetitiveness, the question on how to measure repetitions in a text naturally arises. This work deals with the issue of looking for and evaluating text features that might help the prediction of the impact of MT adaptation on translation quality. In particular, the repetition rate metric, we recently proposed, is compared to other features employed in very related NLP tasks. The comparison is carried out through a regression analysis between feature values and MT performance gains by dynamically adapted versus non-adapted MT engines, on five different translation tasks. The main outcome of experiments is that the repetition rate correlates better than any other considered feature with the MT gains yielded by the online adaptation, although using all features jointly results in better predictions than with any single feature.

1 Introduction

Language and content repetitiveness\(^1\) is a key factor for the successful deployment of translation memories (TMs) (Somers, 2003) as well as statistical machine translation (MT) (Koehn, 2010). The capability of a TM to provide useful translation suggestions for a text segment relies on the chance of finding segments with very similar content – i.e. with a significant percentage of overlapping words – inside a large repository of already translated texts. On the other hand, statistical MT also relies on the assumption that the segment to be translated shares with the training data a significant amount of patterns, from single words to groups of words.

Advances on the integration of human post-editing into MT have recently revealed the potential of incremental and online MT adaptation. While doing their job, post-editors are also generating fresh in-domain training data. Adding these data to the MT engine would for instance mean letting MT focus on the lexical choices of the post-editor or even avoid future translation mistakes. Given that the internal knowledge of statistical MT is mainly represented by translation patterns, the potential impact of MT adaptation strictly depends on how much such type of patterns will re-occur in future sentences. We can hence claim that text repetitiveness is a precondition for the effectiveness of MT adaptation in general, whether this is performed on sentence batches as for incremental adaptation, or single sentence as in online adaptation. Thus, the question becomes how to measure repetitions in a text in a way to help the prediction\(^2\) of the impact of MT adaptation on translation quality for that text. Such a prediction

\(^1\)In this paper the word repetitiveness is not used with a negative meaning, e.g. boring, unpleasant.

\(^2\)Although sometimes they are given slightly different meaning, in this work we consider prediction and forecast as synonyms.
could of course be useful to avoid the cost and even damage of applying adaptation in case the text results unfit or guide the choice among alternative adaptation methods.

This work deals with the issue of identifying and evaluating source text features that do significantly predict the performance of MT adaptation. Actually, the analysis of text and the design of features for modeling text characteristics are issues investigated in various NLP topics. As a consequence, many features have been already proposed and investigated by the scientific community which we can draw inspiration from; nevertheless, none of them was specifically designed for capturing the repetitiveness of text. Indeed, in the case of MT related tasks, like the quality estimation of MT output, in addition to features computed on the source text, features have been proposed which involve the translated/target text or even the MT models: although they can be really effective, we focus our investigation to the source side only, since we are interested in deciding what kind of MT system is most suitable for translating a given text before having any MT engine at disposal.

In this paper we experimentally assess the repetition rate, that we recently proposed in (Bertoldi et al., 2013) where no support to its effectiveness was provided, as a single light measure to characterize a full document to be translated. Roughly, the repetition rate computes the rate of event types (single words and \textit{n}-grams) that occur more than once in a text; for making this statistics independent from the size of the document, it is computed on a fixed-size sliding window. We measured the prediction power of the repetition rate on several MT adaptation tasks and compared it against other text features that were proposed for very related NLP tasks. The comparison was carried out through a regression analysis between feature values and MT performance gains by an online adapting MT engine versus a static, non-adapted MT engine. Five different experimental tasks were considered, defined over two domains and three language pairs.

The main outcome of experiments is that the repetition rate correlates better than any other considered feature with the MT gains yielded by the online adaptation, although using all features jointly results in better predictions than with any single feature. Therefore, it seems feasible to decide in advance whether to activate or not the online adaptation procedure, just looking at the values of very few features, even just the repetition rate, of the text to be translated.

The remainder of the paper is organized as follows. First, an overview of related works with particular attention to text features is provided in Section 2. The repetition rate is formally described in Section 3, while Section 4 lists the other features we have selected for comparison purposes. Data, their analysis, the experimental setup and results are presented and commented in Section 5. A summary and the list of future works end the paper.

2 Related Work

To our knowledge this is the first work that deals with the problem of predicting the effectiveness of MT adaptation by means of features of the input text. On the contrary, the identification of text properties has been an essential problem in many NLP tasks, like text categorization, readability assessment, text comprehension, text complexity evaluation, automatic text-plagiarism detection, source and translation classification, information retrieval. Moreover, a number of features have been used in previous work for quality (also referred to as “confidence”) estimation of MT output (Blatz et al., 2003), the task closer to the problem we are dealing with.

Aware of not being exhaustive, we survey a list of features used in some of those tasks that in principle could be useful for our purposes.

Blatz et al. (2003) describe 91 sentence-level confidence features and some additional word level features used in experiments. We cannot borrow most of them because they either are dependent on the MT models or involve the target text, and as stated in the introduction we are not interested in features with such dependency; concerning the measures on the source side,
we think that the source length is not relevant for us, the log-probability and the perplexity of
the source sentence are somehow included in our investigation, while the twelve quartile range
measures of the source \(n\)-gram frequency deserve a more thorough discussion. They are defined
as follows: Each list of distinct \(n\)-grams in the training corpus is first ordered by frequency and
then split into four parts containing approximately an equivalent number of elements (quartiles);
for each source sentence, the percentage of 1-, 2-, and 3-grams in each of the four frequency
quartile ranges is then computed, for a total of 12 values. Since they characterize single sen-
tences, the 12 values cannot be used as they are to model a whole text; therefore, we will not
consider them in our experiments. Nevertheless, in the definition of our repetition rate, the ra-
tionale behind those quartiles is somehow considered: in fact, as we will see, we partition the
\(n\)-grams into two groups, depending whether they occur once or more times.

In text categorization (Sebastiani, 2002), features such as single tokens or stems are mostly
used. In the typically employed bag-of-words representation, information about dependencies
and the relative position of tokens are not used. Anyway, they can be introduced at some extent
through phrasal features consisting of more than one token: syntactic and statistical phrases
\((n\text{-grams})\) have been investigated for a long time and many works report classification improve-
ments over the use of single tokens, especially by introducing not too long \(n\)-grams (Fürnkranz,
1998), outcome that we exploit by defining the repetition rate over 1- to 4-grams.

Readability assessment is a form of text classification aiming at retrieving texts that suit a
particular target reading level. In a school setting, it can help teachers to find texts appropriate
to their students; other real-life contexts where it can play an important role are those involving
people with intellectual disabilities, dyslexics, immigrant populations, and second or foreign
language learners. Commendably, Vajjala and Meurers (2012) present dozens of features used
in previous research on text readability and complexity and group them into three broad cate-
gories: lexical, syntactic and traditional features. Examples of features from the first group are
the type-token ratio (see Section 4) and the lexical density, defined as the ratio of the number
of lexical word tokens (nouns, adjectives, verbs, adverbs) and the number of all tokens (total
number of words) in the analysed text. Syntactic features include mean length of clauses and
sentences, and co-ordinate phrases and complex nominals per clause. The average sentence
length in words and the number of characters or syllables per word are listed as traditional fea-
tures. Reported experiments showed that the most predictive features are from the lexical group,
result that led us to include them in our comparison.

Plagiarism detection can be divided into two main strategies, namely intrinsic plagiarism
detection, that utilizes only information within the suspected document, and external plagiarism
detection, that compares the suspected document against a set of possible sources. Just as a
hint, the last “International competition on plagiarism detection” focused on external plagiarism
detection only (Potthast et al., 2013). For both types, first works relied on \(n\)-grams, possibly
sorted to bring them into a canonical form which cancels out plagiarism obfuscation.\(^3\) Further
efforts were devoted to text pre-processing, like synonym normalization, stemming, stop words
deletion. Successively, attention was extended to lexical, syntactic and semantic features, like
reordering and alignment of words, POS and phrase tags, semantic similarity of sentences,
etc. (Lin et al., 2012).

The same or similar features mentioned so far also appear in works on translationese:
\(n\)-grams in (Baroni and Bernardini, 2006); POS-based features and average sentence length,
parse tree depth, proportion of simple/complex sentences, ambiguity as the average of senses
per word, word length as the proportion of syllables per word, lexical richness, and information
load as the proportion of lexical words to tokens in (Ilisei et al., 2010); most frequent words and

\(^3\)Obfuscation is the strategy adopted by real plagiarists to rewrite their source passages in order to make detection
more difficult.
a list of some hundred function words are instead used in (Islam and Hoenen, 2013) and (Koppel and Ordan, 2011), respectively.

3 Repetition Rate

We recently introduced the repetition rate (Bertoldi et al., 2013) as a way to measure the repetitiveness inside a text, by looking at the rate of non-singleton $n$-gram types ($n=1\ldots 4$) it contains. As shown there, this rate decays exponentially with $n$. For combining values with exponential decay, a reasonable scheme is to average their logarithms, or equivalently to compute their geometric mean. Furthermore, in order to make the measure comparable across different sized documents, statistics are collected on a sliding window of one thousand words, and properly averaged. Formally, the Repetition Rate (RR) in a document can be expressed as:

$$RR = \left( \prod_{n=1}^{4} \frac{\sum_{S} (V(n) - V(n,1))}{\sum_{S} V(n)} \right)^{1/4}$$

where $S$ is the sliding window, $V(n,1)$ is the number of singleton $n$-gram types in $S$, and $V(n)$ is the total number of $n$-gram types in $S$. RR ranges between 0 to 1, where the extreme points are respectively reached when all $n$-grams observed in all text windows occur exactly once (RR=0) and more than once (RR=1).

In addition to get RRs that are comparable across texts of different lengths, the reason for using a sliding window is to preserve as much as possible the sequential structure of the original text, and hence its linguistic features, as opposed to what would happen if the sentences to be processed together were sampled.

4 Features for Comparison

4.1 Lexical features

Type-token ratio (TTR) is the ratio $T/N$ of the number $T$ of word types to the total number $N$ of word tokens in a text. It has been widely used as a measure of lexical diversity or lexical variation in language acquisition studies. However, since it is dependent on the text size, various alternative transformations of TTR came into existence. Then, besides TTR, we also considered Vajjala and Meurers (2012): square root TTR, defined as $T/\sqrt{N}$; corrected TTR, $T/\sqrt{2N}$; and bilogarithmic TTR, $\log T/\log N$.

4.2 Entropy-based features

In information theory, perplexity is a measurement of how well a probability distribution or probability model predicts a sample. In case of a discrete probability distribution $p$, the perplexity is defined as

$$PP = 2^{H(p)} = 2^{-\sum_{x} p(x) \log_2 p(x)}$$

where $H(p)$ is the entropy of the distribution and $x$ ranges over events. In NLP, perplexity is a way of evaluating LMs: the better the model for a given text, the lower the PP computed on that text. In our experiments, PP refers the source side of considered subsamples (see Section 5.3) with respect to LMs estimated on the source side of bitexts used for training the models of MT engines; in fact, only the source side of the text to be translated is assumed to be available.

Clearly, we are interested in intrinsic features of texts; on the contrary, the PP of a document is computed with respect to an "external" LM; therefore, we also considered a "self-contained" version of the perplexity, named incremental perplexity (incPP): for each segment of a document, its perplexity is computed on the LM estimated on previous segments.
Table 1: Overall statistics on parallel data used for evaluation purposes: number of segments and running words of source and target sides.

<table>
<thead>
<tr>
<th>domain</th>
<th>pair</th>
<th>segments</th>
<th>tokens source</th>
<th>tokens target</th>
</tr>
</thead>
<tbody>
<tr>
<td>IT</td>
<td>en→it</td>
<td>1,614</td>
<td>14,388</td>
<td>14,837</td>
</tr>
<tr>
<td></td>
<td>en→fr</td>
<td>1,614</td>
<td>14,388</td>
<td>15,860</td>
</tr>
<tr>
<td>Legal</td>
<td>en→it</td>
<td>472</td>
<td>10,822</td>
<td>11,508</td>
</tr>
<tr>
<td></td>
<td>en→fr</td>
<td>472</td>
<td>10,822</td>
<td>12,810</td>
</tr>
<tr>
<td></td>
<td>en→es</td>
<td>472</td>
<td>10,822</td>
<td>12,699</td>
</tr>
</tbody>
</table>

and the so-far PP value is incrementally updated; this procedure is iterated until the whole document is processed.

4.3 Out-of-vocabulary rate

The out-of-vocabulary rate (OOV) measures the number of unknown words; usually, it is expressed in percentage. We computed it for each considered subsamples with respect to the same external LMs used for the computation of the perplexity. Although the OOV can hardly be associated to repetitions, we decided to test it as well because some dynamic adaptation techniques, as that of our experiments, can learn unknown words and then yield MT performance gains.

5 Experiments and Evaluation

The investigation presented in this paper has been conducted on data employed in field tests organized by the MateCat project\(^4\) which is developing a Web-based CAT tool for professional translators that integrates new MT functions, like offline and online adaptation performed on user feedback. Texts belong to two domains, namely information technology (IT) and legal (LGL); the language directions are from English into French and into Italian for both domains, while Spanish is the target language for the LGL domain only.

5.1 Evaluation data

For the IT domain, the evaluation document was supplied by the industrial partner of MateCat and consists of 1,614 segments.

For the LGL domain a document (2013/488/EU) was taken from the website of the European Union law,\(^5\) for which translations into the three languages of interest were available. The document was pre-processed so that the segments of the three versions were all aligned. The full document consists of 605 segments and 13,900 English words; the first segments including about 3,000 words were used for development purposes, while the last 472 segments have been used in the experiments reported here.

Table 1 provides some statistics of evaluation texts. The target word counts refer to human references. Note that for each domain, the document to be translated is shared among all language-pairs.

5.2 Preliminary analysis

First of all, we checked if repetitions are randomly distributed or, on the contrary, if some words tend to re-occur more frequently in some portion than in others of the evaluation sets. Figures 1 plot the position of English words of the evaluation sets of the two domains: each

\(^4\)http://www.matecat.com
\(^5\)http://eur-lex.europa.eu
word encountered in the text is assigned a progressive index (ordinate) and for all positions (abscissa) where it occurs a colored point is added to the plot. The envelope of the curves corresponds to the dictionary growth, while the distributions of points show those areas (if any) where repetitions are concentrated. The most evident deviation from the uniform distribution occurs for a portion of one thousand words of the IT test set centered at position 10,000 (top blue circle): a definitely higher growing rate of dictionary can be seen there, with many new words that re-occur often just in that text window. As an example, this is the segment number 1026 (out of 1614), at the beginning of that part of the document:

If Web Services is enabled while deploying Weblogic 12c...

The words Web, Services, Weblogic and 12c are new, that is they were not observed before. Moreover, they re-occur just within the successive 100 segments (29, 16, 13 and 5 times, respectively) but no more in the remaining four hundred segments.

Another interesting phenomenon of that portion of the document is highlighted by the down blue circle: many words whose index is close to 200 do not occur there, while they appear quite uniformly in the rest of the text. For example, the words return (code 212) and value (code 250), which are observed 141 and 118 times in overall, are much less frequent in the hundred segments between 1026 and 1125, where value occurs just twice, return never.

That excerpt is a striking example of a text which differs from the rest of the document in terms of repetitions. Other portions isolated from the rest can be seen not only in the IT document but even in the Legal text, although to a lesser extent. Such a “localism” in the repetitions yielded us to perform measures on subsamples (windows) rather than on the whole documents, as described in Section 5.3.

For each test set (again, for the English source side only), Figure 2 plots the cumulative distributions of the distances between repetitions: each point \((x, y)\) of the curves says that repetitions distant no more than \(x\) positions cover a percentage of all repetitions equal to \(y\). For distances showed in the plot (lower than 120), the IT curve is above the LGL curve by 7-8 percentage points: it means that repetitions in the IT document occur significantly closer than in the LGL data. As a consequence, those adaptation methods affected not only by the amount of repetitions but also by their closeness, will be more effective in the IT domain than in the LGL domain.
Figure 2: Cumulative distribution of repetition distances in the source side of the two test sets: each point \((x, y)\) of curves says that the repetitions occurring at a distance not larger than \(x\) are the \(y\%\) of the total amount of repetitions.

5.3 Experimental setup
The main goal of this investigation is to discover if the features listed in Sections 3 and 4 can predict the effectiveness of adaptation metrics, and in case to what extent. A straightforward approach is to measure the correlation between the values of features and of automatic MT quality metrics. As we are interested in the impact of text features on MT adaptation, we chose as target values for our predictors the relative gains in MT scores achieved with adaptation, i.e. the difference between the MT scores of the engines with and without the online adaptation module. Concerning which MT quality metric, for this investigation, we focused on the BLEU score, leaving the experiments on other metrics to future activity.

The localism observed in Section 5.2 determined us to perform the measure of features and of BLEU gains on shifting windows of text. The involvement of windows has also the positive side-effect to allow the computation for each test set of a number \(N\) of \((\text{features}, \text{gain})\) pairs instead of a single pair got from the whole document. Figure 3 illustrates the scheme used to compute a list of \((\text{features}, \text{gain})\) pairs for a given document. It is assumed that the source text and the reference translation of that document are given; moreover, two MT engines are available, one representing the reference system (in our experiments, that without the adaptation module), the other being the boosted system which should yield some performance gain (in our experiments, that including the online adaptation module). First, the whole source text is translated by the two engines, so that the two automatic translations \(\text{MT}_1\) and \(\text{MT}_2\) are obtained. Then, for each window \(W\) of text, the BLEU scores of the two translations are computed and their difference is paired to the text feature(s) we are interested in, like the RR of the source side. Once the whole document have been processed by means of the sliding window \(W\), the procedure ends by outputting a record of \(N\) pairs, which defines the dataset for the computation of the correlation. In our experiments, the size of \(W\) is set to 2,000 words, a reasonable trade-off between the needs of preserving the localism and of reliable computations. \(N\) is equal to the number of considered windows, which is determined by the window size, the size of the whole document and the moving step. Since the two sizes were given, the moving step was chosen so that \(N\) is large enough to allow a reliable computation of
the correlation ($N \geq 100$).

For measuring the statistical dependency we are interested in, several regression algorithms could be used, such as polynomial regression, robust regression, regression trees, etc. We decided to employ support vector machines (SVMs) (Vapnik, 1995). SVMs are supervised learning models used for classification and regression analysis and are particularly suitable for our problem where not large training data are involved. In fact, SVM can generalize complicated input patterns with only a very few support vectors.

Practically, we used the LIBSVM (Chang and Lin, 2011), a software that provides support vector regression (SVR). In particular, we adopted a linear kernel with an $\epsilon$-SVR, since in a preliminary investigation this setup overtook other kernel types (we tried: polynomial, radial basis function and sigmoid). In $\epsilon$-SVR the goal is to find a function $f(x)$ that has at most $\epsilon$ deviation from the actually obtained targets $y_i$ for all the training data and at the same time as flat as possible. The model produced by $\epsilon$-SVR only depends on a subset of the training data, because the cost function for building the model ignores any training data that is close (within a threshold $\epsilon$) to the model prediction. The two parameters $C$ and $\epsilon$ in loss function of $\epsilon$-SVR were set by running a randomized parameter optimization (Bergstra and Bengio, 2012).

For $\epsilon$-SVR, LIBSVM outputs mean squared error (MSE) and the squared correlation coefficient $r^2$; in our experiments, they were computed by cross validation on the basis of a 10-parts split of evaluation sets.

5.4 SMT engines

As already stated, two domains and three language pairs, for a total of five different tasks, are involved in our experiments. In the following two sections, training data and SMT engines involved in the experimental stage are described.

5.4.1 Training data

For training purposes we relied on several language resources, including parallel corpora and translation memories. As far as the IT domain is concerned, software manuals from the OPUS corpus (Tiedemann, 2012), namely KDE4, KDE4-GB, KDEdoc, and PHP were used. They
are all publicly available. In addition, a proprietary large translation memory (TM), that is a collection of parallel entries, was employed. It mostly consists of real projects on software documentation commissioned by a specific customer.

For what concerns the legal domain, the publicly available JRC-Acquis collection (Steinberger et al., 2006) was used, which mostly includes EU legislative texts translated into 22 languages.

Table 2 provides detailed statistics on the actual bitexts used for training purposes. In particular, the train entries refer to the whole generic training texts, while development set entries to additional data on which the parameters of the phrase-based MT model were optimized.

The domain selection entry of the IT en→fr task refers to data selected from out-of-domain texts (Giga English-French, United Nation, and Common Crawl corpora\(^6\) (Bojar et al., 2013)) by using the in-domain text as seed in the method proposed by Axelrod et al. (2011) and available within the XenC toolkit (Rousseau, 2013); this was done to augment the amount of training data, since the size of in-domain text available for that language pair (15.4/17.9 million words) is about four times smaller than for the other tasks.

Table 2: Overall statistics on parallel data used for training and development (tuning) purposes: number of segments and running words of source and target sides. Symbol \(M\) stands for \(10^6\).

<table>
<thead>
<tr>
<th>domain</th>
<th>pair</th>
<th>corpus</th>
<th>segments</th>
<th>tokens</th>
<th>source</th>
<th>target</th>
</tr>
</thead>
<tbody>
<tr>
<td>IT</td>
<td>en→it</td>
<td>train</td>
<td>5.4 M</td>
<td>57.2M</td>
<td>59.9M</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>development set</td>
<td>2,156</td>
<td>26,080</td>
<td>28,137</td>
<td></td>
</tr>
<tr>
<td></td>
<td>en→fr</td>
<td>train</td>
<td>1.1 M</td>
<td>15.4M</td>
<td>17.9M</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>domain selection</td>
<td>1.2 M</td>
<td>20.0M</td>
<td>22.2M</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>development set</td>
<td>4,755</td>
<td>26,747</td>
<td>30,100</td>
<td></td>
</tr>
<tr>
<td>Legal</td>
<td>en→it</td>
<td>train</td>
<td>2.7 M</td>
<td>61.4M</td>
<td>63.2M</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>development set</td>
<td>181</td>
<td>5,967</td>
<td>6,510</td>
<td></td>
</tr>
<tr>
<td></td>
<td>en→fr</td>
<td>train</td>
<td>2.8 M</td>
<td>65.7M</td>
<td>71.1M</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>development set</td>
<td>600</td>
<td>17,737</td>
<td>19,613</td>
<td></td>
</tr>
<tr>
<td></td>
<td>en→es</td>
<td>train</td>
<td>2.3 M</td>
<td>56.1M</td>
<td>62.0M</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>development set</td>
<td>700</td>
<td>32,271</td>
<td>36,748</td>
<td></td>
</tr>
</tbody>
</table>

5.4.2 Decoders

The SMT systems have been built upon the open-source MT toolkit Moses (Koehn et al., 2007). The translation and lexicalized reordering models were trained on parallel training data, i.e., entries train (IT English-to-Italian, legal English-to-Italian/French/Spanish tasks), and train plus domain selection (IT English-to-French task) of Table 2. Back-off 5-gram language models smoothed with the improved Kneser-Ney technique (Chen and Goodman, 1999) were estimated on the target side of the available bilingual training data. The standard MERT procedure provided within the Moses toolkit was used to optimize the weights of the log-linear interpolation model on development sets whose content is coherent to training data and of adequate size (entries development set of Table 2).

For each task, an SMT engine was built over the above mentioned models and used for the standard translation of the test sets. Since the models do not change during the translation,

---

<table>
<thead>
<tr>
<th>pair</th>
<th>MT engine</th>
<th>IT</th>
<th>LGL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>BLEU</td>
<td>TER</td>
</tr>
<tr>
<td>en→it</td>
<td>STA</td>
<td>57.5</td>
<td>26.3</td>
</tr>
<tr>
<td></td>
<td>DYN</td>
<td>64.6</td>
<td>21.0</td>
</tr>
<tr>
<td>en→fr</td>
<td>STA</td>
<td>41.4</td>
<td>37.9</td>
</tr>
<tr>
<td></td>
<td>DYN</td>
<td>56.3</td>
<td>28.9</td>
</tr>
<tr>
<td>en→es</td>
<td>STA</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>DYN</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

Table 3: Overall performance of MT engines with respect to human references on evaluation sets.

these systems are named static (STA) and represent the reference systems.

For each static system, a companion dynamic (DYN) system has been built that dynamically adapts to post-eds, as they become available. More in detail, in the DYN system a global model, which is the same as that in the STA system, is combined with a local model, which is empty when the learning process starts. The combined model is used to generate the translation of the current input. The user post-edits the automatic translation and the amended text feeds back the system. The local model is then refined on the user feedback and the combined model updated, ready to translate the next segment. The process iterates over all sentences of the document to be translated.

In our systems, the local model is implemented by a caching mechanism. The caching regards both translation and language models: phrase pairs extracted from the alignment of the source and post-edit are extracted and inserted into the cache-based translation model, while n-grams of the post-edit fill the cache-based language model. More details are provided in (Bertoldi et al., 2013). Note that in our experiments the post-editing is simulated by using human references.

5.5 Results and comments

First of all, Table 3 provides BLEU (Papineni et al., 2002), TER (Snover et al., 2006) and GTM (Turian et al., 2003) scores computed on the evaluation documents with respect to human references for each of the five considered translation tasks.

In both domains, but especially for IT, the improvements over the static systems yielded by the dynamic adaptation technique are remarkable. Focusing on the BLEU score, in the IT domain it improves by more than 7 absolute points for English-to-Italian (57.5 to 64.6), and almost 15 absolute points for English-to-French (41.4 to 56.3); in the LGL domain, the gain is quite limited in English-to-Italian (1.5 absolute points), but definitely notable – almost 5 points – for the other two directions.

The successive experiments regard the actual prediction of MT score gains on the basis of text features. Following the shifting window scheme described in Section 5.3 and shown in Figure 3, for each window the BLEU difference (ΔBLEU) of STA and DYN MT was recorded and features computed; the total number of considered windows was 135 for the IT test set, 200 for the LGL domain. The linear regression between ΔBLEU and either each single feature or all features was then calculated by means of the LIBSVM software (Section 5.3): the cross-validation MSE and \( r \) are collected in Table 4. Postponing for a while any comment on the IT en→it task which is an outlier, the main outcomes on the other four tasks are:
Table 4: Cross validation mean squared errors (cv-MSE) and correlation coefficients (cv-r) between ∆BLEU and text features for the five different tasks.

<table>
<thead>
<tr>
<th>cvMSE/cv-r</th>
<th>IT en→fr</th>
<th>en→it</th>
<th>LGL en→es</th>
<th>en→fr</th>
<th>en→it</th>
</tr>
</thead>
<tbody>
<tr>
<td>RR</td>
<td>5.98/0.76</td>
<td>11.35/0.05</td>
<td>0.81/0.51</td>
<td>1.15/0.46</td>
<td>0.98/0.60</td>
</tr>
<tr>
<td>TTR</td>
<td>8.25/0.64</td>
<td>11.58/0.02</td>
<td>0.87/0.43</td>
<td>1.06/0.53</td>
<td>1.19/0.46</td>
</tr>
<tr>
<td>sqrtTTR</td>
<td>7.57/0.68</td>
<td>11.74/0.00</td>
<td>0.90/0.39</td>
<td>2.15/0.33</td>
<td>1.17/0.46</td>
</tr>
<tr>
<td>crrTTR</td>
<td>7.57/0.68</td>
<td>12.08/0.08</td>
<td>3.66/0.21</td>
<td>1.09/0.51</td>
<td>1.18/0.47</td>
</tr>
<tr>
<td>blgTTR</td>
<td>8.45/0.63</td>
<td>11.11/0.15</td>
<td>0.89/0.42</td>
<td>1.07/0.52</td>
<td>1.18/0.46</td>
</tr>
<tr>
<td>PP</td>
<td>14.06/0.26</td>
<td>11.11/0.01</td>
<td>1.15/0.05</td>
<td>1.46/0.08</td>
<td>1.55/0.10</td>
</tr>
<tr>
<td>incPP</td>
<td>8.37/0.63</td>
<td>11.43/0.07</td>
<td>0.88/0.43</td>
<td>1.04/0.55</td>
<td>1.18/0.47</td>
</tr>
<tr>
<td>OOV</td>
<td>11.04/0.46</td>
<td>11.53/0.22</td>
<td>1.07/0.22</td>
<td>1.45/0.22</td>
<td>1.55/0.29</td>
</tr>
<tr>
<td>all features</td>
<td>5.85/0.77</td>
<td>6.35/0.69</td>
<td>0.56/0.69</td>
<td>0.81/0.66</td>
<td>0.55/0.79</td>
</tr>
</tbody>
</table>

- RR is highly correlated with ∆BLEU in any task;
- in three out of four tasks, RR is the best predicting feature; only in the LGL en→fr task it is not the best, but it is anyway competitive both in terms of correlation and of mean squared error;
- although it is a good predictor, TTR performs pretty worse than RR;
- the variants of TTR proposed for making its measure independent from the size of the text do not seem to outperform the original formulation;
- incPP undeniably outperforms PP, it is competitive with TTR and it is even the best predictor in the LGL en→fr task;
- if considered all together, the features definitely correlate with MT gains better than individually, especially on LGL tasks; for the IT en→fr however, it has to be considered that some single features are indeed very good predictors (mainly RR), with performance hard to beat.

Concerning the IT en→it task, no single feature is capable to effectively predict the MT gain; on the contrary, the correlation is very high if they work together. The latter outcome is positive because it confirms that the gains from MT adaptation are somehow predictable; from the other side, inefficacy of individual features is disappointing, especially considering the opposite results on the companion IT en→fr task, where the same source document is translated. For further investigation, the scatter plot (with the regression line) of the 135 (RR, ∆BLEU) points of the IT en→it task is shown in the plot on the left of Figure 4. It is evident the weak dependency between the two variables, as already suggested by the negligible correlation coefficient (0.05) reported in Table 4.

Nevertheless, the points appear not to be randomly distributed; in fact, by naively splitting the test set in four equal-sized and contiguous parts, the points are grouped as shown in the second plot of Figure 4, where much smaller deviations from the regression lines are revealed: in fact, the correlation coefficient for each quarter is 0.84, 0.56, 0.54 and 0.54, respectively. And they could be even higher if the split was more precise, for example by separating the II and III blocks so that the blue points with RR higher than 35 could be merged with the green points. A similar behavior has been observed for the other features, as well. That means that for the IT en→it task, the linear dependency between single features and MT gains exists but it is local and changes through the test document. This issue will be investigated in the future.
6 Conclusions

In this paper, we experimentally assessed the repetition rate as a novel text feature for the prediction of the effectiveness of MT adaptation. Taking the gains yielded by a paradigmatic online adaptation technique as the dependent variable, we performed a regression analysis over a number of text features considered as independent variables. Results on five MT tasks, different in terms of domain or language pair, showed that the repetition rate is the best predictor, although the most accurate regression model uses the features all together.

Concerning the future work, besides the problem on the IT en→it task sketched out at the end of Section 5.5, we will handle some other open issues. First of all, here we focused our analysis to MT gains expressed in terms of BLEU score; we think it is recommendable to consider other measures as well, like TER and GTM, to be sure that our outcomes are not “metric-dependent”. Another interesting extension regards “negative” text samples: in previous papers, we showed that if the RR of the text to be translated is not high enough, the online adaptation cannot significantly improve the reference performance of the static MT engine; then, we will extend the systematic investigation presented here to such problematic tasks, like the translation of news and of TED talks.

Results showed that taken all together, the considered features correlate better than individually with MT gains: it will be shown the relative contribution of each single feature to the overall performance. Finally, the assessment of minor aspects of our experimental setup should be considered: (i) the size \( n \) of the \( n \)-grams involved in the definition of the RR, here set to 4; (ii) the size of the subsample \( S \) on which the RR is computed, here set to 1,000 words (see Section 3); and (iii) the size of the sliding window for handling the localism, here set to 2,000 words (see Section 5.3).

Acknowledgements

This work was supported by the MateCat project (grant agreement 287688),\(^4\) which is funded by the EC under the 7\(^{th}\) Framework Programme.
References


Expanding Machine Translation Training Data with an Out-of-Domain Corpus using Language Modeling based Vocabulary Saturation

Burak Aydın burak.aydin@tubitak.gov.tr
TÜBİTAK-BİLGEM, Gebze 41470, KOCAELİ, TURKEY
Department of Computer Engineering, Boğaziçi University, Bebek 34342, İSTANBUL, TURKEY

Arzucan Özgür arzucan.ozgur@boun.edu.tr
Department of Computer Engineering, Boğaziçi University, Bebek 34342, İSTANBUL, TURKEY

Abstract
The training data size is of utmost importance for statistical machine translation (SMT), since it affects the training time, model size, decoding speed, as well as the system’s overall success. One of the challenges for developing SMT systems for languages with less resources is the limited sizes of the available training data. In this paper, we propose an approach for expanding the training data by including parallel texts from an out-of-domain corpus. Selecting the best out-of-domain sentences for inclusion in the training set is important for the overall performance of the system. Our method is based on first ranking the out-of-domain sentences using a language modeling approach, and then, including the sentences to the training set by using the vocabulary saturation filter technique. We evaluated our approach for the English-Turkish language pair and obtained promising results. Performance improvements of up to +0.8 BLEU points for the English-Turkish translation system are achieved. We compared our results with the translation model combination approaches as well and reported the improvements. Moreover, we implemented our system with dependency parse tree based language modeling in addition to the n-gram based language modeling and reported comparable results.

1 Introduction
Most of the statistical methods that attempt to solve natural language processing problems achieve better results with increasing training data sizes. In statistical machine translation, the amount of data directly affects a system’s overall success. Increasing the size of the training data by effectively utilizing the data available in other domains (e.g. web, news, medical) using domain adaptation and data selection techniques is a promising research direction for improving the performance of an SMT system. This is especially important for low-resource languages and domains, for which there are only limited amounts of training data available. The English-Turkish language pair is an example low-resource language pair for machine translation. Most of the publicly available corpora for this language pair contain only thousands of sentences, whereas the training sets of language pairs with more resources (e.g. English-French) usually contain millions of sentences. The number of parallel sentences in the training set for English-Turkish hardly reaches to millions, even when all available corpora from different domains are combined.
In this paper, we introduce an approach that effectively combines different data selection methods for expanding in-domain training data with the available out-of-domain data in statistical machine translation. The method first scores the sentences in the out-of-domain corpus based on their similarities to the in-domain corpus using a language modeling approach. Then, it adapts the vocabulary saturation filter technique, which has recently been proposed in (Lewis and Eetemadi, 2013) for reducing the training data and model sizes, to the domain adaptation problem. The proposed approach is applied to English-Turkish machine translation by using n-gram based as well as dependency parse tree based language modeling and improvements in terms of BLEU scores are achieved.

The paper is organized as follows. Section 2 presents the related work on data selection and domain adaptation in SMT. Section 3 briefly explains the Vocabulary Saturation Filter algorithm and the proposed approaches. Section 4 describes the data and the experimental setup, and provides the obtained results. Section 5 discusses the results of the study, and Section 6 outlines possible future directions for research.

2 Related Work

In statistical machine translation, several approaches have been proposed for data selection, domain adaptation, and data preprocessing and cleaning. Eck et al. (2005) selected a subset of a monolingual corpus and human-translated it to use for a low-resource language pair. The aim for selecting a suitable subset is not only for decreasing the model size, but also for improving the translation quality as in the work of Okita (2009).

Data selection and preprocessing methods generally aim to be successful in reducing data and model size significantly with a minimum score loss (Lewis and Eetemadi, 2013). Domain adaptation techniques on the other hand, target not only to optimize the data and model size, but also to improve system score. A number of different approaches including language modeling (Bulyko et al., 2007) and source-sentence classification (Banerjee et al., 2010) have been proposed for domain adaptation. Machine translation systems mostly work well only in one domain and domain adaptation techniques usually improve the score of a system in that domain. Wang et al. (2012) attempted to build a system that works well in multiple-domains simultaneously. Their method tries to use models of different domains in a combined system and automatically detects the domain and its parameters at runtime. Axelrod et al. (2011) compared different data selection methods by selecting subsets from a large general domain parallel corpus and proposed a new data selection method, which uses bilingual cross-entropy difference. In addition to the usage of conventional n-gram language models, Duh et al. (2013) used neural language models to select training data from general domain.

In some circumstances, there may be lack of in-domain bilingual data. Wu et al. (2008) used out-of-domain corpora to train a baseline system and then used in-domain translation dictionaries and in-domain monolingual corpora to improve the in-domain performance. Their method unifies old and newly produced resources in a combined framework.

Moore and Lewis (2010) tried to solve the efficient data selection problem in the language model training step, which is an essential feature of statistical machine translation systems. In this work, they compared the cross-entropy according to domain-specific and non-domain-specific language models for each sentence that is used to produce the non-domain-specific model. They calculated the cross-entropy difference to select the data. Using this approach, they produced better language models to use in their machine translation system.

Bertoldi and Federico (2009) attempted to significantly improve the performance of machine translation by exploiting large monolingual in-domain data. They synthesized a bilingual corpus by translating the monolingual adaptation data. Their work is based on adapting an already developed translation system into another domain in which there is no enough parallel
data available.

Dependency parsing based language modeling has also been investigated in many studies. Shen et al. (2008) built a framework to employ a target dependency language model (DLM) for machine translation. It predicts the next child based on the previous children of the current head. DLM was used in many tasks such as sentence realisation (Guo et al., 2008), speech recognition (Lambert et al., 2013), and sentence completion (Gubbins and Vlachos, 2013). In our dependency-based language modeling approach we represent sentences with trigrams (i.e., dependent, head, and dependency type) extracted from their dependency parse trees. The out-of-domain sentences are ranked based on the dependency relation language models learned from the in-domain corpus.

Another relevant approach is the phrase table combination method proposed by Bisazza et al. (2011). They tried to expand the in-domain phrase table with an out-of-domain phrase table in an efficient manner. In the fill-up method, they used the phrases from the out-of-domain table only if they are not available in the in-domain table. We compared our approach with their method and also with the linear interpolation technique. Our systems obtained better BLEU scores in overall. Additionally, the system in (Bisazza et al., 2011) uses all of the available corpora for phrase table building, whereas ours uses a proportion of the out-of-domain data in addition to the in-domain data.

Recently, Lewis and Eetemadi (2013) proposed the Vocabulary Saturation Filter algorithm. The algorithm tries to significantly reduce the training data size. It starts by counting the n-grams from the beginning of the corpus and when all n-grams of a specific sentence reach to a previously defined threshold frequency \( t \), then this sentence is excluded from the resulting subset that is to be used for machine translation system training. They showed that unigrams are enough to choose a subset, since higher order n-grams resulted in selecting the majority of the original corpus.

Our approach is an effective combination of language modeling and vocabulary saturation filtering (VSF) for expanding training data using an out-of-domain corpus. VSF has originally been used for data size reduction (Lewis and Eetemadi, 2013), but in this study we adapt it to use in expanding in-domain data with an out-of-domain corpus for machine translation. Before applying the VSF technique, we pre-rank the out-of-domain parallel corpus based on the sentence perplexities calculated using an in-domain language model. We investigate using dependency tree based language modeling as well as n-gram based language modeling. We apply the approach to the English-Turkish language pair and report promising results.

3 Method

3.1 Vocabulary Saturation Filter (VSF)

The effect of more data on improving BLEU scores is clearly observed through experiments: as more data is added, BLEU scores increase. However, the relationship between the quantity of data and BLEU is not linear, such that addition of new data does not increase much after some point. There is a saturation point of data and the VSF algorithm attempts to find it (Lewis and Eetemadi, 2013). It selects the data within a threshold and uses it for machine translation. The algorithm is very successful in reducing training data size (Lewis and Eetemadi, 2013). In this paper, we use VSF to improve translation results by expanding the training data with out-of-domain data.

3.2 N-gram Language Modeling based VSF on In-domain Data

The VSF algorithm has originally been proposed to reduce training data size with a possible loss in BLEU scores and its power has been shown for French-English translation by applying the algorithm on the sentences in the given order (Lewis and Eetemadi, 2013). In other words,
no any sentence sorting has been performed before applying the algorithm. The data selection with VSF starts from the beginning of the corpus, hence the sentence order makes difference for the algorithm’s choice. Therefore, in this study we propose to pre-sort the sentences using a language modeling based approach before applying the VSF algorithm. The procedure for the described method can be formalized in the following steps:

- Build a language model using the target side of the parallel corpus.
- Score the target side of the parallel corpus based on the previously built language model.
- Rank the parallel corpus with respect to the sentence perplexity scores.
- Apply VSF algorithm on the ordered corpus to select a subset and use it as training data for the SMT system.

The results of the original approach and language modeling based approach are shown in Table 5 in the results section.

3.3 Dependency Language Modeling based VSF on In-domain Data

In the previous section, the pre-ranking process on the training set is applied based on n-gram language modeling. The corpus is ranked with respect to sentence perplexity scores. We proposed other sorting mechanisms than n-grams based modeling and experimented in end-to-end machine translation systems to investigate their effects. The main idea is to use the dependency relations between the words in a sentence and to identify whether these relations will provide a better ranking for the sentences in the training corpus. Notice that the data selection procedure here is almost the same as the method described in the previous section, the only difference is the sorting mechanism. We tried several combinations of these dependency relation features and reported the results. The representations extracted from the tree in Figure 1 are exemplified in Table 1. The dependency relations for the Turkish sentences are obtained using the parser developed by Eryigit et al. (2008).

After representing the training corpus with the settings exemplified in Table 1, we collected the statistics based on these modified corpus’ dependency-based unigrams and sorted the corpus accordingly. For example, for System 1 in Table 1 each word_label_word relation in a sentence is treated as a unigram. Next, VSF is applied through the corpus to select a more efficient subset from the beginning. The selected corpus is used for training statistical machine translation systems and the results are compared with the baseline and the n-gram based ranked VSF systems in Table 6.

We decided to focus on the setting which gives the best BLEU score and applied it for sentence ranking when trying to utilize out-of-domain data.
Table 1: Dependency-based representation of the sentence in Figure 1

<table>
<thead>
<tr>
<th>System</th>
<th>Representation</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>No dependency</td>
<td>-</td>
<td>çocuk eve gitti</td>
</tr>
<tr>
<td>System 1</td>
<td>word_label_word</td>
<td>çocuk_subject_gitti eve_adjunct_gitti</td>
</tr>
<tr>
<td>System 2</td>
<td>pos_label_pos</td>
<td>noun_subject_verb noun_adjunct_verb</td>
</tr>
<tr>
<td>System 3</td>
<td>word_label_pos</td>
<td>çocuk_subject_verb eve_adjunct_verb</td>
</tr>
<tr>
<td>System 4</td>
<td>All</td>
<td>Representations from system 1-3 are combined</td>
</tr>
</tbody>
</table>

3.4 Language Modeling based VSF for Out-of-domain Data

As briefly stated before, the method proposed in this paper is a combination of different data selection algorithms, namely Language Modeling (LM) and Vocabulary Saturation Filter (VSF). VSF has originally been proposed to reduce training data and model size with a minimum score loss. However, in this paper, we adapt the VSF approach to increase the training data size using out-of-domain data with the goal of improving the performance of an SMT system.

The VSF algorithm selects the training subset from the corpus by counting the seen n-grams. The algorithm starts to read the corpus from the beginning, so it is more likely that the sentences at the beginning of the corpus will be chosen by the algorithm. This affects the sentence choice. What we propose is to order the out-of-domain data with respect to a language model built from the in-domain data and select a subset from it through VSF in order to add into the in-domain data for training. The approach, whose workflow is shown in Figure 2, may be summarized as in the following steps.

- Build a language model using the target side of the in-domain parallel corpus.
- Score the sentences in the target side of the out-of-domain parallel corpus by the language model produced in the previous step.
- Rank the sentences in the out-of-domain corpus based on sentence perplexity scores.
- Apply the VSF algorithm on the sorted corpus.
- Use the selected out-of-domain corpus sentences together with the in-domain corpus sentences for training a machine translation system.

5-gram language models are built from the in-domain data after tokenization using the SRILM toolkit (Stolcke, 2002). After scoring the out-of-domain sentences with the language model learned from the in-domain data, the sentences are ranked according to their perplexity scores. Since lower perplexity scores correspond to better fitting to the applied language model, sentences with lower perplexity scores appeared at the top of the corpus. Hence, their chances to be selected by the VSF algorithm increased. We applied the VSF technique on the sorted out-of-domain sentences by using the unigrams. In other words, we counted the seen unigrams while selecting the subset from the ranked corpus, since higher order n-grams lead to the selection of almost the entire corpus.

We also ranked the out-of-domain corpus with respect to dependency-based relations investigated in the previous section. Only the best scoring system is applied for the utilization of out-of-domain data and its results are also reported. The results of all systems that utilize out-of-domain data are shown in Table 7.
The proposed approach aims at increasing the score through out-of-domain data selection, rather than solely reducing training data size as in the original VSF method. Using VSF with perplexity-based pre-ranking of out-of-domain sentences for expanding the in-domain training data for statistical machine translation is the main contribution of this paper.

4 Experiments

4.1 The Machine Translation Systems

We used a string-to-string machine translation system for English-Turkish in order to investigate the effects of the proposed method. We implemented the SMT systems with a phrase-based approach (Koehn et al., 2003). We generated word alignments using MGIZA (Gao and Vogel, 2008) and used Moses Open Source toolkit (Koehn et al., 2007) for decoding. The parameters of the system are tuned and optimized with the minimum error rate training (MERT) algorithm (Och, 2003). We tuned the system with 3 different seeds and reported the best result obtained for each setting. We trained conventional 5-gram language models (LMs) from the available parallel corpora. All language models were trained with the SRILM toolkit using the modified Kneser-Ney smoothing technique (Kneser and Ney, 1995) and then, binarized using KenLM (Heafield, 2011). We used the sentence perplexity scores produced by SRILM in order to pre-rank the parallel corpora. Moreover, in our implementation of the VSF algorithm, the threshold \( t \) values were chosen in the range \([1, 10]\) for the language pair. The details of the corpora used as in-domain and out-of-domain data are provided in the data section. For n-gram choice in VSF, unigrams are used as in the original study, since higher order n-grams select more than half of the original data. Choosing the major proportion of the data diminishes the system score as shown with the experiments, where all out-of-domain data are added to the training phase.


<table>
<thead>
<tr>
<th>Data Set</th>
<th>Sentences</th>
<th>Unique Words</th>
<th>Total Words</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turkish</td>
<td>131K</td>
<td>158K</td>
<td>1.8M</td>
</tr>
<tr>
<td>English</td>
<td>131K</td>
<td>45K</td>
<td>2.5M</td>
</tr>
</tbody>
</table>

Table 2: WIT training data statistics

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Sentences</th>
<th>Unique Words</th>
<th>Total Words</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turkish</td>
<td>165K</td>
<td>143K</td>
<td>3.9M</td>
</tr>
<tr>
<td>English</td>
<td>165K</td>
<td>60K</td>
<td>4.6M</td>
</tr>
</tbody>
</table>

Table 3: SETIMES training data statistics

4.2 Data

For the experiments, we used WIT (Cettolo et al., 2012) data as in-domain and SETIMES (Tyers and Alperen, 2010) data as out-of-domain data. The WIT\(^1\) corpus contains a collection of transcribed and translated talks and the core is the TED talks. On the other hand, SETIMES\(^2\) corpus is in the news domain, collected from a website covering events in the Balkans. The statistics for the English-Turkish parallel corpora are given in Table 2 and Table 3.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Sentences</th>
</tr>
</thead>
<tbody>
<tr>
<td>dev2010</td>
<td>887</td>
</tr>
<tr>
<td>test2010</td>
<td>1568</td>
</tr>
<tr>
<td>test2011</td>
<td>1433</td>
</tr>
<tr>
<td>test2012</td>
<td>1698</td>
</tr>
<tr>
<td>test2013</td>
<td>1022</td>
</tr>
</tbody>
</table>

Table 4: English-Turkish test data statistics

As test sets, we used the test2010, test2011, test2012 and test2013 sets. The system is tuned with the dev2010 data set. These test sets were used in the IWSLT\(^3\) competitions in the respective years. These test and development sets also contain collections of talks retrieved from TED talks. The sentence counts for the test and development sets are given in Table 4.

4.3 Results

The models are trained with the corpora described in the previous section. The BLEU score (Papineni et al., 2001) is used as an evaluation metric on the test sets. First, we implemented the baseline systems trained with the original in-domain data. Then, we added all of the out-of-domain data to the baseline system and retrieved the results. Using the whole out-of-domain data did not increase the BLEU score as much as expected. Afterwards, we ranked the out-of-domain data with respect to the sentence-perplexity scores based on a language model built with in-domain data. Then, we selected subsets of the sorted corpus with various VSF frequency threshold settings and used them in the machine translation systems.

---

1https://wit3.fbk.eu/
2http://opus.lingfil.uu.se/SETIMES.php
3http://www.iwslt2013.org/
Table 5: BLEU scores for the system on in-domain data only: t is the frequency threshold for the VSF algorithm

<table>
<thead>
<tr>
<th>System</th>
<th>Sentence Count</th>
<th>test2010</th>
<th>test2011</th>
<th>test2012</th>
<th>test2013</th>
</tr>
</thead>
<tbody>
<tr>
<td>PB baseline</td>
<td>131K</td>
<td>7.94</td>
<td>7.94</td>
<td>8.02</td>
<td>7.09</td>
</tr>
<tr>
<td>VSF(t=1)</td>
<td>77K</td>
<td>7.34</td>
<td>7.46</td>
<td>7.48</td>
<td>6.69</td>
</tr>
<tr>
<td>VSF(t=2)</td>
<td>92K</td>
<td>7.56</td>
<td>7.51</td>
<td>7.57</td>
<td>7.06</td>
</tr>
<tr>
<td>VSF(t=5)</td>
<td>108K</td>
<td>7.60</td>
<td>7.71</td>
<td>7.56</td>
<td>6.83</td>
</tr>
<tr>
<td>n-gram sorted data + VSF(t=1)</td>
<td>85K</td>
<td>7.32</td>
<td>7.42</td>
<td>7.59</td>
<td>6.81</td>
</tr>
<tr>
<td>n-gram sorted data + VSF(t=2)</td>
<td>100K</td>
<td>7.29</td>
<td>7.60</td>
<td>7.70</td>
<td>6.88</td>
</tr>
<tr>
<td>n-gram sorted data + VSF(t=5)</td>
<td>115K</td>
<td>7.54</td>
<td>7.70</td>
<td>7.83</td>
<td>6.94</td>
</tr>
</tbody>
</table>

According to the results in Table 5, the data reduction also worked for the English-Turkish system. Using the 58% of the total data, we recovered 93%, 94%, 94%, and 95% of the BLEU scores for test sets test2010, test2011, test2012, and test2013, respectively. On the other hand, compared with the original VSF, the language modeling based approach did not improve performance much for this case. This shows that language modeling based sorting for an only in-domain parallel corpus may not be a good metric. If we have only one parallel corpus available and we are to sort it, then metrics such as translation quality of sentence pairs can provide more promising results in the only in-domain data case.

Table 6: BLEU scores for the dependency-based ranked systems stated in Table 1 on in-domain data only: t is the frequency threshold for the VSF algorithm

<table>
<thead>
<tr>
<th>System</th>
<th>Sentence Count</th>
<th>test2010</th>
<th>test2011</th>
<th>test2012</th>
<th>test2013</th>
</tr>
</thead>
<tbody>
<tr>
<td>PB baseline</td>
<td>131K</td>
<td>7.94</td>
<td>7.94</td>
<td>8.02</td>
<td>7.09</td>
</tr>
<tr>
<td>ngram based + VSF (t=1)</td>
<td>85K</td>
<td>7.32</td>
<td>7.42</td>
<td>7.59</td>
<td>6.81</td>
</tr>
<tr>
<td>System 1 + VSF (t=1)</td>
<td>89K</td>
<td>7.42</td>
<td>7.37</td>
<td>7.59</td>
<td>6.32</td>
</tr>
<tr>
<td>System 2 + VSF (t=1)</td>
<td>74K</td>
<td>6.92</td>
<td>6.86</td>
<td>7.34</td>
<td>6.48</td>
</tr>
<tr>
<td>System 3 + VSF (t=1)</td>
<td>91K</td>
<td>6.78</td>
<td>7.06</td>
<td>7.32</td>
<td>6.12</td>
</tr>
<tr>
<td>System 4 + VSF (t=1)</td>
<td>92K</td>
<td>7.09</td>
<td>6.75</td>
<td>7.14</td>
<td>6.22</td>
</tr>
</tbody>
</table>

After using the n-gram language modeling based ranking approach, we experimented with various dependency based ranking approaches to select in-domain data. As shown in Table 6 the selection of data with dependency-based rankings did not overperform the n-gram based approach. The representation System1 was the best scoring representation, hence we experimented with this setting in the machine translation systems that include the utilization of out-of-domain data.

In the English to Turkish translation systems that utilize out-of-domain data, the best scoring system for test2012 set uses only 33% of the SETIMES data, which is our out-of-domain data. For test2011, again the same system achieves the best score. The improvement in the BLEU score is around 0.3 points. The improvements over the individual systems for test2011 and test2012 were computed to be statistically significant with a 95% confidence interval ($p<0.05$) (Koehn, 2004). Note that the BLEU scores for this language pair are generally low due to the differences between the Turkish and English languages. Turkish is morphologically more complex and the word orders between this language pair differ as well. Additionally,
Table 7: BLEU scores for the systems utilizing out-of-domain data: t is the frequency threshold for the VSF algorithm (Sentence count starting with ‘+’ indicates the additional amount of sentences included to the data of the baseline system shown in the first row).

<table>
<thead>
<tr>
<th>System</th>
<th>Sentences</th>
<th>test2011</th>
<th>test2012</th>
<th>test2013</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Only WIT</td>
<td>131K</td>
<td>7.94</td>
<td>8.02</td>
<td>7.09</td>
</tr>
<tr>
<td>2. (1) + SETIMES</td>
<td>+165K</td>
<td>8.00</td>
<td>8.1</td>
<td>7.16</td>
</tr>
<tr>
<td>3. (1) + ngram sorted-SETIMES + vsf(t=1)</td>
<td>+53K</td>
<td><strong>8.24</strong></td>
<td><strong>8.38</strong></td>
<td><strong>7.27</strong></td>
</tr>
<tr>
<td>4. (1) + ngram sorted-SETIMES + vsf(t=2)</td>
<td>+66K</td>
<td>8.12</td>
<td>8.2</td>
<td>7.15</td>
</tr>
<tr>
<td>5. (1) + ngram sorted-SETIMES + vsf(t=5)</td>
<td>+80K</td>
<td>8.05</td>
<td>8.15</td>
<td>6.88</td>
</tr>
<tr>
<td>6. (1) + dep. sorted-SETIMES + vsf(t=1)</td>
<td>+72K</td>
<td>8.27</td>
<td>8.38</td>
<td>7.19</td>
</tr>
<tr>
<td>7. (1) + dep. sorted-SETIMES + vsf(t=2)</td>
<td>+93K</td>
<td>7.89</td>
<td>8.11</td>
<td><strong>7.42</strong></td>
</tr>
<tr>
<td>8. (1) + dep. sorted-SETIMES + vsf(t=5)</td>
<td>+118K</td>
<td>8.18</td>
<td>8.32</td>
<td>7.2</td>
</tr>
<tr>
<td>9. linear(WIT + SETIMES)</td>
<td>+165K</td>
<td>7.64</td>
<td>7.81</td>
<td>7.16</td>
</tr>
<tr>
<td>10. fillup(WIT + SETIMES)</td>
<td>+165K</td>
<td>7.46</td>
<td>7.84</td>
<td>6.87</td>
</tr>
</tbody>
</table>

the size of the data for this pair does not reach to million sentences, which is generally a case for language pairs like French-English. In (Yılmaz et al., 2013), it is discussed that SETIMES data was not helpful to increase the BLEU scores for English-Turkish translation in the IWSLT test sets. However, our results show that this data can be effectively utilized to improve the translation qualities of the corresponding sets.

The other proposed sorting metrics related to dependency relations and part-of-speech tags have also shown minor improvements on some of the test sets. For test set test2013, the best scoring system is the one trained with the out-of-domain corpus ranked using dependency relation based language modeling. We compared our approach with the phrase table combination methods between different domains proposed in (Biszazza et al., 2011). Our system outperformed the phrase table combination approach, which did not bring any improvement for the English-Turkish language pair on the data sets used.

Additionally, we compared our system with TUBITAK’s best system for English-Turkish translation in the IWSLT 2013 evaluation campaign by implementing their work. In (Yılmaz et al., 2013), it is stated that adding all of the SETIMES data did not improve the performance of their system, it even decreased it. Their best system was trained with hierarchical phrase-based translation (Chiang, 2007) and made use of morphological and lexical features specific to the Turkish language. We adopted their work and also reported that the addition of the entire out-of-domain data to the baseline system decreases the BLEU score. Next, we integrated our proposed data selection methods to the system. The results that we obtained are shown in Table 8. The BLEU score has increased by 0.8 points and this score is higher than the best score in the corresponding IWSLT evaluation campaign for English-Turkish translation. The increase was tested using (Koehn, 2004) and computed to be statistically significant with a 95% confidence interval (p<0.05). Although our replication of the system by Yılmaz et al. (2013) did not include some of the features that they have used and shown to improve performance, our system is still able to outperform their reported best system for the test2013 data set. In this system setting, we also experimented applying VSF on non-sorted (original) SETIMES data for expanding in-domain data to investigate whether sorting makes a difference or not. The results in Table 8 show that the n-gram sorting based VSF selection approach performs better than the non-sorting based VSF selection approach, even though it did not help much in the experiments where the models were only trained with in-domain data.
The results show that the proposed technique successfully utilizes the available out-of-domain data and leads to improvements in BLEU for the specified domain. The approach is more a data selection technique among domains, rather than domain adaptation in which a pre-built system in a specific domain is being adapted to a different domain. It is useful for building better and more successful systems for a domain, where there is not much data, but there is a lot of data in different domains.

5 Discussion

In this study, we introduced an approach for expanding machine translation training data by utilizing an out-of-domain corpus through vocabulary saturation. We proposed using sentence ranking strategies based on n-gram and dependency relation language modeling. We evaluated the proposed methods for English-Turkish translation. This language pair does not have sufficient amount parallel texts, hence it is important to fully utilize and use all the available texts from different domains. Due to the morphological and word-sequential differences between the English and Turkish languages, most translation systems produce low BLEU scores. Turkish is an agglutinative language and is subject-object-verb oriented, whereas English is more compact and subject-verb-object oriented. Our results show that the proposed technique leads to significant improvement upon the best English-Turkish translation system reported in the IWSLT 2013 evaluation campaign. It also significantly outperforms the phrase table combination approach proposed for utilizing out-of-domain data by Bisazza et al. (2011).

The proposed approach may easily be integrated to state-of-the-art machine translation systems and applied to other language pairs. Since additional and valuable out-of-domain data is selected through this method, we believe it will lead to improvement of machine translation systems’ overall successes for other languages as well. The VSF and n-gram language modeling perplexity-based rankings are algorithms that have already been proposed for machine translation. However, the combination of these approaches for expanding in-domain training data with out-of-domain data is a new approach for statistical machine translation.

<table>
<thead>
<tr>
<th>System</th>
<th>Sentences</th>
<th>test2013</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. TUBITAK IWSLT Best</td>
<td>131K</td>
<td>8.41</td>
</tr>
<tr>
<td>2. (1) + SETIMES</td>
<td>+165K</td>
<td>8.37</td>
</tr>
<tr>
<td>3. (1) + ngram sorted-SETIMES + vsf(t=1)</td>
<td>+53K</td>
<td>9.14</td>
</tr>
<tr>
<td>4. (1) + ngram sorted-SETIMES + vsf(t=2)</td>
<td>+66K</td>
<td>9.20</td>
</tr>
<tr>
<td>5. (1) + ngram sorted-SETIMES + vsf(t=5)</td>
<td>+80K</td>
<td>8.58</td>
</tr>
<tr>
<td>6. (1) + dep. sorted-SETIMES + vsf(t=1)</td>
<td>+72K</td>
<td>8.66</td>
</tr>
<tr>
<td>7. (1) + dep. sorted-SETIMES + vsf(t=2)</td>
<td>+93K</td>
<td>8.61</td>
</tr>
<tr>
<td>8. (1) + dep. sorted-SETIMES + vsf(t=5)</td>
<td>+118K</td>
<td>8.75</td>
</tr>
<tr>
<td>9. (1) + no-sort-SETIMES + vsf(t=1)</td>
<td>+74K</td>
<td>7.85</td>
</tr>
<tr>
<td>10. (1) + no-sort-SETIMES + vsf(t=2)</td>
<td>+94K</td>
<td>8.65</td>
</tr>
<tr>
<td>11. (1) + no-sort-SETIMES + vsf(t=5)</td>
<td>+121K</td>
<td>8.91</td>
</tr>
</tbody>
</table>

Table 8: BLEU scores of the IWSLT 2013’s best system and the proposed approaches: t is the frequency threshold for the VSF algorithm (Sentence count starting with ‘+’ indicates the additional amount of sentences included to the data of the baseline system shown in the first row)
6 Future Work

In the proposed methodology, there are several future directions to investigate. Ranking the corpus with an external language model is a possible direction to follow. Other potential avenues for research are using different sentence sorting metrics and features. Instead of the sentence perplexity based scoring method, other features such as sentence length or the feature functions introduced in (Taghipour et al., 2011) can be integrated into the system.

As mentioned up to now, we had data from two different domains in our experiments. We are planning to investigate the effect of adding more out-of-domain data from more different domains and to see if the system score will continue to increase or not.

Moreover, we plan to examine the effects of the method specifically on the translation model. That is, we will create phrase-tables from different domains and sort the phrases in the out-of-domain phrase table with respect to an in-domain language model. Then, we will apply VSF on the sorted out-of-domain phrase table to select a subset of phrases to concatenate with the in-domain phrase table in the machine translation system.

References


Comparison of Data Selection Techniques for the Translation of Video Lectures

Joern Wuebker¹
Hermann Ney¹,²
¹RWTH Aachen University, Aachen, Germany
²Univ. Paris-Sud, France and LIMSI/CNRS, Orsay, France

Adrià Martínez-Villaronga
Adrià Giménez
Alfons Juan
Universitat Politècnica de València, València, Spain

Christophe Servan
Marc Dymetman
Shachar Mirkin
Xerox Research Centre Europe, Meylan, France

Abstract

For the task of online translation of scientific video lectures, using huge models is not possible. In order to get smaller and efficient models, we perform data selection. In this paper, we perform a qualitative and quantitative comparison of several data selection techniques, based on cross-entropy and infrequent n-gram criteria. In terms of BLEU, a combination of translation and language model cross-entropy achieves the most stable results. As another important criterion for measuring translation quality in our application, we identify the number of out-of-vocabulary words. Here, infrequent n-gram recovery shows superior performance. Finally, we combine the two selection techniques in order to benefit from both their strengths.

1 Introduction

With the continuous growth of available bitexts and research advances of the underlying technology, statistical machine translation (SMT) has become popular for many real world tasks. The most common approach is still the phrase-based paradigm (Koehn et al., 2003), that provides an efficient framework with good translation quality for many language pairs.

This work focuses on the application of SMT to the task of translating scientific video lectures. Online scientific video lectures are becoming increasingly popular, e.g. in the context of massive open online courses (MOOCs). Being able to provide high quality automatic translations for this kind of technical talks could, e.g., prove beneficial to education at universities,
sharing technical knowledge and connecting researchers around the world.

With today’s large amounts of available data for SMT training, selecting the most valuable portions can be crucial to obtain good performance. First, for the practical task of online translation, using huge models is inefficient and can render real-time translation impossible, especially on mobile devices. The use of smaller training data leads to faster and more space-efficient translation systems. Secondly, selecting the data that is most relevant to the domain at hand, e.g. scientific lectures, can have a significant impact on translation quality. This is why we look for approaches that get or adapt small and efficient models. The task of adapting a translation system to perform well on a specific type of language is called domain adaptation and will be discussed in Section 2. One of the prominent branches of domain adaptation research is data selection.

In this work, we perform a qualitative and quantitative comparison of several data selection techniques based on two oppositional criteria, cross-entropy and infrequent $n$-gram recovery. While the cross-entropy criterion selects sentences that are most similar to a given domain, infrequent $n$-gram recovery puts the emphasis onto adding new information to the translation system. Our results show that in terms of BLEU, a combination of translation and language model cross-entropy achieves the most stable results.

However, for the task of translating scientific lectures, the number of out-of-vocabulary (OOV) words is also an important criterion to evaluate translation quality. Although in our experiments OOV words make up only a small portion of the data and thus have no visible effect on BLEU, we show examples where it does impact the translation quality as perceived by humans. With respect to the OOV rate, infrequent $n$-gram recovery has strong advantages over cross-entropy based methods.

Finally, we propose to combine the data selected with two different approaches in order to benefit from both new information provided by infrequent $n$-gram recovery and domain-related distribution.

The paper is organized as follows. We will discuss previous research on domain adaptation in Section 2 and provide a detailed description of the data selection techniques in Section 3. Section 4 gives an account of the statistical translation system used in our experiments. Finally, the experimental setup and results are discussed in Section 5 and we conclude with Section 6.

2 Domain Adaptation

Domain adaptation can be performed in different ways: using lightly-supervised approaches, model combination/update or data selection.

2.1 Lightly-supervised approaches

A common way to adapt a statistical machine translation model is to use lightly-supervised approaches. These approaches aim to self-enhance the translation model. This was first proposed by Ueffing (2006) and refined by Ueffing et al. (2007). The main idea is to filter the translations with the translated test data. This process involves a confidence measure in order to select the
most reliable data to train a small additional phrase table (PT). The generic and the new phrase tables are used jointly for translation, which can be seen as a mixture model with one specific PT built for each test set.

The lightly-supervised training approach proposed by Schwenk (2008) does not adapt the model to the test data, but it proposes to add large amounts of monolingual training data translated using a completely new model. Lambert et al. (2011) enhanced this approach by using the translations of monolingual data in the target language.

2.2 Model combination and update

One way to adapt MT models is to combine translation models. Models can be combined using the mixture-model approach, a log-linear combination or through incremental learning approaches.

To obtain a mixture of domain-specific models trained on several different domain-specific corpora, they can be combined using a log-linear or a linear approach (Foster and Kuhn, 2007; Civera and Juan, 2007). The standard log-linear model may be used to combine some domain-specific models (Koehn and Schroeder, 2007). In the same way target language models may be combined using a log-linear or a linear combination (Schwenk and Koehn, 2008). Sennrich et al. (2013) proposed to combine different specific parts of the phrase-table during translation leading to a multi-domain adaptation approach.

Niehues and Waibel (2012) compared several incremental approaches, namely the backoff, the factored, the log-linear and the fill-up (Bisazza et al., 2011) techniques. These approaches aim at adapting an MT system towards a target domain using small amounts of parallel in-domain data. The main outcome of this paper is that all the approaches successfully improve the generic model and none of them is better than the others. The performances of the approaches mainly depend on their match to the specific data.

2.3 Data selection

The main idea of data selection is to try to take advantage of a generic corpus by picking out a subset of training data that is most relevant to the domain of interest.

Two main approaches are used to perform domain adaptation. On one hand, such approaches use information retrieval techniques and similarity scores. On the other hand, language models are used associated to perplexity and cross-entropy.

Intuitively, seeking the data closest to the test set is related to information retrieval techniques. Lü et al. (2007) present this approach using the standard measure $TF.IDF$ (Term Frequency – Inverse Document Frequency) to measure the similarity between the test sentences and the training sentences. This approach is based on a bag-of-words scheme.

The second approach, based on language models (LMs), was originally proposed by Gao and Zhang (2002). Here, the generic corpus is scored against an LM trained on a seed of domain-specific data, and the cross-entropy is computed for each sentence. Then, the same generic corpus is scored against an LM trained on a random sample taken from itself. Now,
sentences of the generic corpus are sorted regarding the computation of the difference between
domain-specific score and generic score. At last, the best amount of the sorted data has to be
determined. This best point is found by minimizing the perplexity of a development set on
growing percentages of the sorted corpus.

Moore and Lewis (2010) reported that the perplexity decreases when less, but more appro-
priate data is used. Recent works expand this approach to bitexts (Axelrod et al., 2011; Mansour
et al., 2011).

Approaches like corpus weighting (Shah et al., 2010) or sentence weighting (Matsoukas
et al., 2009; Mansour and Ney, 2012) are not suitable to our translation task because these
approaches can produce huge models by considering the whole data.

3 Cross-entropy based Data Selection versus Infrequent \( n \)-gram Recovery

In this section we detail the different approaches experimented with for data selection. On one
hand we process the data selection for both LM and translation model (TM) using cross-entropy.
On the other hand, the infrequent \( n \)-gram recovery (Gascó et al., 2012), is explored.

3.1 Language Model Cross-entropy

The LM cross-entropy difference can be used for both monolingual data selection for LM train-
ing as described by Moore and Lewis (2010), or bilingual selection for translation model train-
ing (Axelrod et al., 2011).

Given an in-domain corpus \( I \) and an out-of-domain or general-domain corpus \( O \), first we
generate a random subset \( \hat{O} \subseteq O \) of approximately the same size as \( I \), and train the LMs \( LM_I \)
and \( LM_{\hat{O}} \) using the corresponding training data. Afterwards, each sentence \( o \in O \) is scored
according to:

\[
H_{LM_I}(o) - H_{LM_{\hat{O}}}(o)
\]  

(1)

where \( H \) is the length-normalised LM cross-entropy, which is defined by:

\[
H_{LM}(x) = - \frac{1}{|x|} \sum_{i=1}^{|x|} \log p_{LM}(x_i|x_{i-1})
\]  

(2)

for an LM with a 2-gram order. \( |x| \) denotes the number of tokens in sentence \( x = x_1, x_2, \ldots, x_{|x|} \). It is computed analogously for higher order LMs.

This idea was adapted by Axelrod et al. (2011) for bilingual data selection for the purpose
of translation model training. In this case, we have both source and target in-domain corpora
\( I_{src} \) and \( I_{trg} \), and correspondingly, out-of-domain corpora \( O_{src} \) and \( O_{trg} \), with random subsets
\( \hat{O}_{src} \subseteq O_{src} \) and \( \hat{O}_{trg} \subseteq O_{trg} \). We score each sentence pair \( (s, t) \) by the sum of the cross-
entropy differences on both source and target side:

\[
\hat{H}_{LM}(s, t) = H_{LM_{I_{src}}}(s) - H_{LM_{\hat{O}_{src}}}(s) + H_{LM_{I_{trg}}}(t) - H_{LM_{\hat{O}_{trg}}}(t)
\]  

(3)
Note that since the scores in Equation 3 are computed for the source and target separately, any target sentence \( t' \) whose cross-entropy score is similar to that of \( t \) can exchange \( t \) and have a similar score assigned to it by this method. As a result, poorly aligned data cannot be detected by LM cross-entropy scoring only.

### 3.2 Translation Model Cross-entropy

The IBM-Model 1 (M1) (Brown et al., 1993) is a model used in state-of-the-art SMT systems for a variety of applications. In this work, we apply M1 scores to achieve adaptation to some domain-specific data. Mansour et al. (2011) extend the formulation by Axelrod et al. (2011), which is described in Equation (3), by adding the M1 cross-entropy score to the LM cross-entropy score. The M1 cross-entropy for a sentence pair \( (s, t) = ((s_1, ..., s_{|s|}), (t_1, ..., t_{|t|})) \) is defined as:

\[
\bar{H}_{M1}(s, t) = H_{M1}(t|s) - H_{M1}(t) + H_{M1}(s|t) - H_{M1}(s)
\]

where

\[
H_{M1}(t|s) = -\sum_{i=1}^{|t|} \frac{1}{|t|} \log \left( \frac{1}{|s|} \sum_{j=1}^{|s|} p_{M1}(t_i|s_j) \right)
\]

The cross-entropy of the inverse M1 model \( H_{M1}(s|t) \) is calculated by switching \( s \) and \( t \) in Equation (5).

This metric has several advantages:

- both standard and inverse direction M1 is used, which leads to a more balanced scoring
- it uses cross-entropy difference which has a better correlation with the sample’s similarity to a specific domain than simple cross-entropy (cf. (Moore and Lewis, 2010))
- M1 is a translation model and thus can capture the translation quality of a given sentence pair.

We use a linear interpolation of LM and M1 cross-entropy scores for data selection, which Mansour et al. (2011) have shown to perform best. Such a combination is similar to an SMT system decoder score, where one combines several model scores including an LM and a TM. The score of the interpolated metric is defined by:

\[
\alpha \cdot \bar{H}_{LM}(s, t) + (1 - \alpha) \cdot \bar{H}_{M1}(s, t)
\]

In our experiments, the value of \( \alpha \) is set to \( \alpha = 0.8 \), which has proven to perform well on previous tasks. In the following sections, we will refer to the interpolated metric defined by Equation 6 as the selection based on translation model (TM) cross-entropy.

### 3.3 Infrequent n-gram Recovery

The performance of phrase-based machine translation systems relies on the quality of the phrases extracted from the training samples. Unfortunately, training corpora typically yield
sparse phrases. This means that those word alignments that appear rarely in the training corpus cannot be accurately computed and consequently the phrases cannot be properly extracted.

The goal of infrequent n-gram recovery, introduced by Gascó et al. (2012), is to increase the informativeness of the training set by adding sentences that provide information not seen in the given training corpus. The sentences selected from a generic parallel corpus (from here on, referred to as pool) must contain infrequent n-grams, i.e. n-grams that appear less than a given threshold $\tau$ in the training corpus, referred to as infrequency threshold. If the source language sentences to be translated are known beforehand, the set of infrequent n-grams can be reduced to the ones present in those sentences.

An infrequency score is defined for the sentences, so that they can be sorted to select the most informative ones. Let $X$ be the set of n-grams that appear in the sentences to be translated and $w$ one of them; $C(w)$ the counts of $w$ in the source language training set; and $N_f(w)$ the counts of $w$ in $f$, where $f$ is the sentence from the pool to be scored. The infrequency score of $f$ is defined as follows:

$$i(f) = \sum_{w \in X} \min(1, N_f(w)) \frac{\max(0, \tau - C(w))}{Z(f, |w|)}$$  \hspace{1cm} (7)

In Equation 7, in order to avoid assigning a high score to noisy sentences with many occurrences of the same infrequent n-gram, only one occurrence of each n-gram is taken into account when computing the score. Additionally, a normalization constant $Z(f, |w|)$ is included in the equation, which will be set to 1 if no normalization is used, or to the number of n-grams of order $|w|$ in $f$, i.e. $|f| - |w| + 1$, otherwise.

Each time a sentence is selected, the scores of the remaining sentences are updated in order to avoid the selection of too many sentences with the same infrequent n-gram. However, since rescoring the whole pool would incur a very high computational cost, a suboptimal search strategy is followed. The search is constrained to the set of the one million highest scoring sentences.

In the experiments performed in this work, we will consider n-grams up to order 3 and an infrequency threshold of $\tau = 25$, values that have proven to perform well in similar previous tasks. Note that, as mentioned, this selection technique depends on the sentences to be translated which, for these experiments, are the source sentences from the test set.

3.4 Comparison

In this paper, cross-entropy based and infrequent n-grams based approaches are compared. But some adjustments need to be made in order to compare them.

Different from the cross-entropy based methods described in Sections 3.1 and 3.2, the selection based on infrequent n-grams uses knowledge of the actual development and test sets (described in Section 5). For a fair comparison, we want to see if the cross-entropy based technique can also benefit from this additional knowledge. To that end, we exchanged the in-domain training corpus $I$ with a concatenation of development and test set to perform a cross-
entropy based selection from the out-of-domain data. Here, we denote the development set as \( D \) and the test set as \( T \). We only use the source side of the data, which renders IBM-Model 1 cross-entropy unusable. Thus, we use only language model cross-entropy, modifying Equation 3 by dropping the terms based on the target language. We will refer to this technique as test cross-entropy:

\[
\bar{H}_{LM}(s, t) = H_{LM_{D,src}+T_{src}}(s) - H_{LM_{O,src}}(s)
\] (8)

It should also be noted that the criteria applied for data selection are oppositional between the cross-entropy and the infrequent \( n \)-gram approaches. The cross-entropy based methods select the sentences that are most similar to a given in-domain data set. The goal here is to use the most domain-relevant data. On the other hand, infrequent \( n \)-gram recovery selects those sentences that are most different to the data that is already given, trying to provide the translation system with new information. Therefore, it seems natural to combine the two orthogonal techniques.

**Combined method:** We performed additional experiments, where part of the data is selected based on infrequent \( n \)-gram recovery and part is selected with TM model cross-entropy. This way, we hope to benefit from the new information introduced by the first while reinforcing a domain-specific distribution at the same time. In practice we start with the maximum amount of data selected by infrequent \( n \)-gram recovery. On top of this, we now add increasing amounts of data selected by TM model cross-entropy, until the full general domain data has been added.

4 Statistical Translation System

We use the standard phrase-based translation decoder from the open source toolkit *Jane* (Wuebker et al., 2012) for all translation experiments. The translation process is framed as a log-linear combination of models, which is a generalization of the source-channel paradigm introduced by Brown et al. (1993). The decoder searches for the best translation \( \hat{e}_1^I \) as defined by the models \( h_m(e_1^I, s^K_1, f^I_1) \). It can be written as (Och and Ney, 2004)

\[
\hat{e}_1^I = \arg \max_{I, e_1^I} \left\{ \sum_{m=1}^{M} \lambda_m h_m(e_1^I, s^K_1, f^I_1) \right\},
\] (9)

where \( f^I_1 = f_1 \ldots f_J \) is the source sentence, \( e_1^I = e_1 \ldots e_I \) the target sentence and \( s^K_1 = s_1 \ldots s_K \) their phrase segmentation and alignment.

The feature functions \( h_m \) include translation channel models in both directions, lexical smoothing models in both directions, an \( n \)-gram language model, phrase and word penalty, a jump-distance-based distortion model, a hierarchical orientation model (Galley and Manning, 2008) and an \( n \)-gram cluster language model (Wuebker et al., 2013). The log-linear feature weights \( \lambda_m \) are optimized on a development data set with minimum error rate training (MERT) (Och, 2003). As optimization criterion we use BLEU (Papineni et al., 2001).
5 Experiments

In this section we describe the different experiments we made in order to compare between the approaches.

5.1 The VideoLectures.NET Repository

VideoLectures.NET\(^1\) is a free and open access repository of video lectures mostly filmed by people from the Jožef Stefan Institute (JSI, Slovenia) at major conferences, summer schools, workshops and science promotional events from many fields of science. VideoLectures.NET has so far published more than 15\(K\) lectures, all of them recorded with high-quality, homogeneous standards. VideoLectures.NET is a major player in the diffusion of the open-source Matterhorn platform\(^2\).

VideoLectures.NET has been adopted as the main target repository in the transLectures\(^3\) project. The main objective of transLectures is to develop innovative, cost-effective solutions for producing accurate transcriptions and translations of lectures published on Matterhorn-related repositories. For system development and evaluation purposes, about 27 English lectures (20 hours) from VideoLectures.NET were manually transcribed and translated into several languages. In particular, 23 of these 27 lectures (16 hours) were translated into French by professional translators.

5.2 Data

Our experiments are performed on the task of translating manually transcribed English video lectures into French. In addition to around 5000 sentence pairs from VideoLectures.NET, we use the parallel TED talk data provided for the shared translation task of the International Workshop on Spoken Language Translation\(^4\) as in-domain data. The general domain data consists of several corpora. The COSMAT scientific thesis abstracts (Lambert et al., 2012) and the news-commentary-v8 corpus, provided by the ACL 2013 8th Workshop on Statistical Machine Translation\(^5\) (WMT), are directly added to the baseline without instance selection due to their small size. The large corpora on which data selection is performed, are the Europarl-v7 corpus (also provided by WMT), the JRC-Acquis corpus (Steinberger et al., 2006) and the Open Subtitles corpus\(^6\) (Tiedemann, 2012). Data statistics for the complete in-domain and out-of-domain data are given in Table 1. For the development and test sets we selected four video lectures each, that were manually transcribed and professionally translated, resulting in a total of 1013 and 1360 sentences for development and test, respectively.

In addition to the target side of the bilingual data, we leverage large amounts of monolingual resources for language model training. These include the Common Crawl Corpus, the 10\(^9\) French-English corpus, the UN corpus and the News Crawl articles, available from the WMT

\(^{1}\)http://videolectures.net
\(^{2}\)http://opencast.org/matterhorn
\(^{3}\)http://translectures.eu
\(^{4}\)http://www.iwslt2013.org
\(^{5}\)http://www.statmt.org/wmt13
\(^{6}\)http://www.opensubtitles.org
Table 1: Data statistics for the bilingual training data. 'Vocabulary' denotes the number of distinct words (i.e. unigrams) that appear in the data.

<table>
<thead>
<tr>
<th></th>
<th>English</th>
<th>French</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>in-domain</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sentences</td>
<td>159K</td>
<td></td>
</tr>
<tr>
<td>Running Words</td>
<td>3.1M</td>
<td>3.3M</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>49K</td>
<td>63K</td>
</tr>
<tr>
<td><strong>out-of-domain</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sentences</td>
<td>13.9M</td>
<td></td>
</tr>
<tr>
<td>Running Words</td>
<td>175M</td>
<td>179M</td>
</tr>
<tr>
<td>Vocabulary</td>
<td>648K</td>
<td>617K</td>
</tr>
</tbody>
</table>

website. In addition, we use the LDC French Gigaword corpus.³ Altogether, the language models are trained on 3.1 billion running words.

5.3 Experimental Setup

The baseline system is trained only on the VideoLectures.NET, TED, COSMAT and news-commentary corpora. For all other systems this data is also used, but is extended by the selected sentences. In all experiments we use the concatenation of the Europarl, JRC and Open Subtitles data as the pool for data selection. As in-domain data \( I \) for the LM and TM cross-entropy based selection, we concatenate the VideoLectures.NET and TED corpora. For the test cross-entropy technique (cf. Section 3.4), these are replaced by the concatenation of the development and test sets, which is denoted by \( D_{src} + T_{src} \) in Eq. 8. Infrequent \( n \)-gram recovery is performed separately for the development and test set. To compare the effectiveness of the different approaches, we select increasing amounts of data with each technique, starting with 250K source words and going up to the full data. The selected data is then added to the baseline system and the models are retrained. However, with infrequent \( n \)-gram recovery, the maximum number of selected source words is 5M. Then, the infrequency threshold is reached and the technique does not select any more sentences.

For the combined method (cf. Section 3.4), we use this maximum amount of data selected with infrequent \( n \)-gram recovery and gradually add additional portions by TM cross-entropy selection.

The language models are kept fixed throughout all experiments. We use a 4-gram standard language model and a 7-gram cluster language model. All results are arithmetic means over three independent MERT runs to control for optimizer stability and are reported in BLEU.

5.4 Results

The BLEU scores we obtained on the test set with the different data selection techniques are plotted in Figure 1. The baseline system without any of the additional data already reaches 33.56% BLEU, while the system using all data yields 33.96% BLEU. Using the development and test data for cross-entropy based selection (test cross-entropy) is clearly not a good idea. The small amount of training data for the language models that are used to compute the cross-
Figure 1: BLEU scores for the different data selection techniques. The x-axis denotes the number of selected source words on a logarithmic scale. The infrequent n-gram recovery selects a maximum of 5M source words, after which the infrequency threshold is reached for all n-grams. The combined method adds additional sentences selected with TM cross-entropy on top of these 5M words.

entropy seems to result in very unreliable estimations for the quality of the data. Further, we can assume that source-only cross-entropy is less stable than complementing it with the target side. However, as it directly makes use of the test set for data selection, it is quicker to recover OOVs (cf. Fig. 2). Regarding the remaining techniques, it is hard to draw a clear conclusion. Due to the small impact of the additional data, all observed values are very close to each other. Altogether, TM cross-entropy seems to yield the most stable results, where translation quality increases with the data size. Both LM and TM cross-entropy based selection reach the same BLEU level as the system using the full data with only \( \frac{1}{4} \) of the data. TM cross-entropy has a slight advantage here, reaching 34.00% BLEU. The best result with selecting only 1M sourced words (0.6% of the full out-of-domain data) is achieved by the infrequent n-gram recovery. However, we observe a drop at the next data point, suggesting that the subsequently selected 1M words perturb the domain-specific distribution, resulting in a lower score.

As was mentioned, the infrequent n-gram recovery selects a maximum of 5M words, after which the infrequency threshold is reached for all n-grams. In order to combine this method with cross-entropy based selection, we kept this maximum selection fixed and gradually added increasing amounts of data selected with the TM cross-entropy criterion. Again, adding only a little data yields a decreasing BLEU score. However, after adding an additional \( \frac{1}{4} \) of the full data, we reach a score of 34.03% BLEU, which is on the same level as the TM cross-entropy selection alone.
Figure 2: Number of words unknown to the translation model for the different data selection techniques. The x-axis denotes the number of selected source words on a logarithmic scale.

Another relevant measure for the user-oriented translation of technical talks is the number of words (i.e., unigrams) that are unknown to the translation decoder, which are left untranslated. Figure 2 displays the number of these out-of-vocabulary words for each selection technique. On this criterion, infrequent n-gram selection is clearly superior to the cross-entropy based techniques. After adding only an additional 500k source words (0.3% of the full out-of-domain data), the number of unknown words is reduced by 47% from 249 to 132. Using all data yields a total of 116 unknown tokens in the test set. From the cross-entropy based methods, selection based on the test set has the best recovery of unknown words, followed by LM cross-entropy scoring. The combined method obviously benefits from the strong performance of the infrequent n-gram recovery, but can hardly add any additional words to its vocabulary.

To illustrate the importance of translating unknown words, we have selected two example sentences from the VideoLectures.NET test set and compared their translations with TM cross-entropy selection and infrequent n-gram selection in Figure 3. In both cases, 1M words were selected from the out-of-domain data. In the first example, the English word re-sell is left untranslated by the system trained with cross-entropy selection, but correctly translated with infrequent n-gram selection. In the second example, commercialise is left untranslated by the first and correctly translated by the latter. Here, the translation does also affect the surrounding words, so that the verb get is translated to aller, which was simply dropped with the TM cross-entropy method.
Figure 3: Example sentences from the VideoLectures.NET test set. 1M source words were selected by both the TM cross-entropy and the infrequent n-gram methods.

6 Conclusion

For the task of translating online scientific video lectures efficient and compact systems are essential, as they may need to be applied in real-time or on mobile devices. Selecting only the most relevant parts of the training data reduces both model size and time and memory requirements and in previous work has also improved translation quality. Therefore, we compared several data selection techniques based on cross-entropy and infrequent n-gram recovery criteria for the translation of English-French video lectures.

As infrequent n-gram recovery uses knowledge of the test set, we also experimented with cross-entropy selection based on the test corpus for a fair comparison. However, in terms of BLEU this method did not prove to be competitive with the standard cross-entropy based approaches. Among the cross-entropy based methods, TM cross-entropy yielded the most stable results, reaching the same performance as using the entire data by selecting a quarter of it. However, it has limited capabilities of adding new words to the vocabulary. With respect to the number of unknown words, infrequent n-gram recovery clearly outperforms the cross-entropy based methods, which can be expected given its design. We illustrated the importance of recovering out-of-vocabulary words for the domain of video lectures on two example sentences. Finally, by combining the two approaches, we achieve the best results both in terms of BLEU and OOV rate.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no 287755 (transLectures), and the Spanish MINECO Active2Trans (TIN2012-31723) research project.
References


Abstract

This paper gives a general review and detailed analysis of China Workshop on Machine Translation (CWMT) Evaluation. Compared with the past CWMT evaluation campaigns, CWMT2013 evaluation is characterized as follows: first, adopting gray-box evaluation which makes the results more replicable and controllable; second, adding one rule-based system as a counterpart; third, carrying out manual evaluations on some specific tasks to give a more comprehensive analysis of the translation errors. Boosted by those new features, our analysis and case study on the evaluation results shows the pros and cons of both rule-based and statistical systems, and reveals some interesting correlations between automatic and manual evaluation metrics on different translation systems.

1 Introduction

The China Workshop on Machine Translation has always been focusing on catching the latest development of Machine Translation (MT) and promoting the communication between related organizations in China. By convention, we organized a unified machine translation evaluation in 2013, sponsored by Chinese Information Processing Society of China and held by the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS).

Compared with the previous evaluation [Zhao et al., 2009], the main improvements of CWMT2013 are as follows: First, we follow the “Gray-Box Evaluation” mode, which not only requires the participants to submit the final translation results but also
results of some key intermediate procedures as gray-box files, such as alignment results, k-best translation, etc. This mechanism makes the results more replicable and controllable, at the same time it enables the participants to identify the weak link in their system pipeline, and make targeted adjustment to improve translation quality. Second, we adopt one rule-based system along with its statistical counterparts to make a more comprehensive comparison between different kinds of MT systems. To increase the diversity of evaluation method, two additional automatic evaluation metrics are also introduced: METEOR [Banerjee and Lavie, 2005] and TER [Snover et al., 2006]. Finally, Besides the automatic evaluation, manual evaluation is also involved in this evaluation. It provides not only the fidelity score and fluency score but also the error types of the translation. This will help us to identify the advantage for each system and the distribution of error types.

Boosted by the above new features, our analysis and case study first shows that rule-based and statistical systems have very different error distributions, and the distributions also vary with different domains. Second we find a serious discrepancy between the automatic and manual evaluation results of the rule-based MT systems, and a detailed study in this problem reveals that automatic evaluation metrics such as BLEU-SBP [Chiang et al., 2008] and METEOR have some bias against rule-based systems. And we also find some correlations between other automatic evaluation metrics.

The rest of the paper is arranged as follows: in the next section, we give an overall introduction to the CWMT2013 evaluation. Section 3 presents manual evaluation results. Section 4 shows the analysis of correlations between several automatic evaluation metrics. In Section 5, we present a case-study on the mismatch between manual and automatic evaluation results. Finally we draw the conclusion and future work in Section 6.

2 Overall Introduction to CWMT2013 Evaluation

2.1 Evaluation Tracks

There are six tracks in CWMT2013 evaluation, covering 5 different language pairs and 4 domains: news domain for Chinese-to-English direction (CE), news domain for English-to-Chinese direction ($EC_n$), scientific domains for English-to-Chinese direction ($EC_s$), and three Chinese minority language tasks including, daily expression domain for Mongolian-to-Chinese (MC), government-doc domain for Tibetan-to-Chinese (TC) and news domain for Uighur-to-Chinese (UC), shown in Table 1.
Table 1: Track and system information for CWMT2013 Evaluation Tasks. The last two columns present the number of participant systems in each task, where Pr. for Primary systems, Pc. for Contrast systems.

<table>
<thead>
<tr>
<th>Task Code</th>
<th>Domain</th>
<th>Language Pair</th>
<th># of test-set</th>
<th>Pr.</th>
<th>Pc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CE</td>
<td>News</td>
<td>CH-EN</td>
<td>1</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>EC_n</td>
<td>News</td>
<td>EN-CH</td>
<td>2</td>
<td>9</td>
<td>15</td>
</tr>
<tr>
<td>EC_s</td>
<td>Scientific</td>
<td>EN-CH</td>
<td>2</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>MC</td>
<td>Daily-expression</td>
<td>MO-CH</td>
<td>2</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>TC</td>
<td>Government-doc</td>
<td>TI-CH</td>
<td>2</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>UC</td>
<td>News</td>
<td>UI-CH</td>
<td>2</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td>11</td>
<td>44</td>
<td>56</td>
</tr>
</tbody>
</table>

2.2 Participants and Systems

There are 16 participants, most of which are institutes and universities such as Chinese Academy of Sciences and Harbin Institute of Technology. Besides, we also have one industrial participant and one foreign participant. 183 translation results of both primary and contrast systems are submitted in this evaluation. The so-called primary system is the main system of each participant in this evaluation and its training data must within the range that the evaluation organizer specified. Contrast system refers to the system that participants use to produce comparative results and its training data is not restricted. We further categorize contrast systems into restricted/non-restricted systems by whether external data is used. Table 1 shows the number of the participants and their systems in each evaluation task.

2.3 Evaluation Data for MT Tracks

The evaluation corpus contains five language directions (Chinese-to-English, English-to-Chinese, Mongolian-to-Chinese, Uighur-to-Chinese, and Tibetan-to-Chinese) and four domains (news, scientific, daily expressions, and government-doc). The input and the output files in the evaluation are encoded in UTF-8 (with BOM) and in strict XML format. All development sets and test sets contain an original text and 4 references. All 4 references are translated from the original text independently by four professional translators. The test-set includes the current test-set of CWMT2013 for ranking and the progress test-set from previous CWMT evaluations to investigate the improvement of each participant system.

The evaluation data inherit all the data in previous CWMT evaluation [Zhao et al., 2009]. Further more, we add new test sets in 4 tasks (EC_s, MC, TC, UC) and update a number of training corpus in Chinese minority language-to-Chinese
<table>
<thead>
<tr>
<th>Task Code</th>
<th>Training-set</th>
<th>Dev-set</th>
<th>Progress test-set</th>
<th>Current test-set</th>
</tr>
</thead>
<tbody>
<tr>
<td>CE</td>
<td>5.84M</td>
<td>1,006</td>
<td>1,003</td>
<td>–</td>
</tr>
<tr>
<td>ECs</td>
<td>5.84M</td>
<td>1,000</td>
<td>1,002</td>
<td>1,001</td>
</tr>
<tr>
<td>ECt</td>
<td>0.9M</td>
<td>1,116</td>
<td>–</td>
<td>1,497</td>
</tr>
<tr>
<td>MC</td>
<td>0.11M</td>
<td>1,000</td>
<td>–</td>
<td>400</td>
</tr>
<tr>
<td>TC</td>
<td>0.12M</td>
<td>650</td>
<td>–</td>
<td>286</td>
</tr>
<tr>
<td>UC</td>
<td>0.11M</td>
<td>700</td>
<td>–</td>
<td>574</td>
</tr>
</tbody>
</table>

Table 2: Number of sentences in the data-sets for CWMT 2013 tasks. The statistics of evaluation data are shown in Table 2.

### 2.4 Gray-Box Evaluation

In order to get a deeper understanding of each translation system, we adopt "Gray-box testing" mode for the first time in our evaluation. It requires participants submit not only the final translation files, but also result files of several key intermediate procedures as gray-box files. Specifically as follows:

Gray-box files for statistical machine translation system includes: Source language preprocessing results of the training corpus; Target language preprocessing results of the training corpus; Word alignment results of the training corpus; Translation rule table filtered by the development set and test set; Preprocessing result of monolingual corpus for language model (LM) training; Language model documentation (instructing LM toolkit, commands and parameters used for LM training); Development set preprocessing results; Decoder configuration file; Test set preprocessing results; Decoder output; Final translation results.

Gray-box files for rule-based machine translation system includes: Test set preprocessing results; Decoder output; Final translation results; Translation rules used for translating test set sentences (optional)

After the evaluation, the organizer shares all the gray-box files of primary systems and baseline systems with the participants, so they could identify the weak link in their translation pipeline and make adjustments accordingly.

### 2.5 Baseline System

This evaluation provides one or more baseline systems for each evaluation task, including source code and corresponding gray-box files. Participants can build their own machine translation systems by optimizing the given baseline system, or they can use
their own systems. The data and translation result provided by baseline system could also be used by participants for research purpose. The evaluations baseline systems are mainly based on two open source systems: Moses [Koehn et al., 2007] and NiuTrans [Xiao et al., 2012]. The corresponding gray-box files are provided by six domestic participants. We show all baseline systems and their providers in Table 3.

<table>
<thead>
<tr>
<th>Task code</th>
<th>Systems</th>
<th>Providers</th>
</tr>
</thead>
<tbody>
<tr>
<td>CE/ECn</td>
<td>Moses</td>
<td>Harbin Institute of Technology</td>
</tr>
<tr>
<td>CE/ECn</td>
<td>Niu-Trans</td>
<td>Northeastern University</td>
</tr>
<tr>
<td>Ec</td>
<td>Moses</td>
<td>Institute of Scientific and Technical Information of China</td>
</tr>
<tr>
<td>MC</td>
<td>Moses</td>
<td>Institute of Computing Technology, CAS.</td>
</tr>
<tr>
<td>TC</td>
<td>Moses</td>
<td>Xiamen University</td>
</tr>
<tr>
<td>UC</td>
<td>Moses</td>
<td>Institute of Automation, CAS.</td>
</tr>
</tbody>
</table>

Table 3: CWMT2013 Evaluation Baseline Systems.

2.6 Performance Measurement

In this evaluation we use a variety of automatic evaluation metrics. The main evaluation metric is BLEU-SBP for its decomposability at sentence level. Other automatic evaluation metrics include: BLEU [Papineni et al., 2002], NIST [Doddington, 2002], GTM [Turian et al., 2006], mWER [Nießen et al., 2000], mPER [Gregor Leusch, 2003], ICT (a metric developed by the Institute of Computing Technology, CAS.), METEOR and TER. In Chinese-to-English direction we also introduce Woodpecker Methodology [Bo et al., 2013], since it could utilize rich linguistic knowledge by setting checkpoints in evaluation.

We adopt two new automatic evaluation metrics METEOR and TER based on the following considerations: BLEU metric is based on n-gram precision, without considering the syntax structure, synonyms, and paraphrase. To solve these problems, recently researchers put forward a variety of new evaluation methods. Among them, the automatic evaluation metric METEOR has been widely accepted. It uses stemming match, synonyms match as well as the exact literal match and considers not only precision but also recall. TER is a classic metric in machine translation [Snover et al., 2006], we use it by calculating the minimum editing distance between translation and reference to
ease the shortcoming of exact literal match.

All metrics (including WoodPecker) are case-sensitive, the evaluation of Chinese is based on Chinese character instead of word. We do significant test [Collins et al., 2005] on the BLEU-SBP results of each primary system. Specifically, for each primary system we test the significant degree of the differences between its translation results and all other primary systems, constructing the significance of difference matrix of all primary systems.

Besides the above automatic evaluation metrics, we carry out manual evaluation on EC task and UC task. The manual evaluation data of EC task comes from EC task in CWMT2011 and manual evaluation data of UC task comes from UC task in CWMT2013. We select 500 sentences from each test set as the manual evaluation corpus.

Manual evaluation focus on the loyalty and fluency of translation results, and these evaluation criteria refer to the Language Norms Based Assessment Specifications of Machine Translation Systems(draft) released by State Language Affairs Commission and the Ministry of Education of People’s Republic of China. Taking practical operability into account, we made some minor modifications. The scoring criteria are shown in Table 4.

Translation results of each participating system were manually evaluated by three native speakers. Then, we take the arithmetic mean of all loyalty/fluency scores of each system as their final loyalty/fluency evaluation scores. During manual evaluation, in addition to evaluating loyalty and fluency, evaluators also need to give translation results a brief analysis of error types pre-set by evaluation organizer including:

<table>
<thead>
<tr>
<th>Loyalty Score</th>
<th>Loyalty Criteria</th>
<th>Fluency Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No translation at all</td>
<td>Completely incomprehensible</td>
</tr>
<tr>
<td>1</td>
<td>Only a few individual words are translated</td>
<td>Only individual phrases or grammatical components are understandable</td>
</tr>
<tr>
<td>2</td>
<td>A few phrases or grammatical components are translated</td>
<td>40% of text is translated fluently, a few grammatical components are understandable</td>
</tr>
<tr>
<td>3</td>
<td>60% of text is correctly translated, or SVO of the translation is correct</td>
<td>60% of text is translated fluently</td>
</tr>
<tr>
<td>4</td>
<td>80% of text is correctly translated</td>
<td>80% of text is translated fluently, or SVO of the text is basically fluent.</td>
</tr>
<tr>
<td>5</td>
<td>All text is correctly translated</td>
<td>Translation is fluent.</td>
</tr>
</tbody>
</table>

Table 4: Scoring Criteria for Manual Evaluation
2.7 Official Evaluation Results

The official evaluation results are released online[^1]. In the following section we will do some meaningful comparison among the participating systems and give a detailed analysis on these results.

3 Analysis on Manual Evaluation Result

3.1 Error Type Analysis

By analyzing the error types of manual evaluation results in ECn task and UC task, we find out that in ECn task, the most frequent errors are "f: word selection error in translation", "b: lack of content words in translation", and "c: word order error". This validates the common wisdom that English and Chinese have very different structures resulting in a lot of long-distance-reordering which the current system couldn’t handle. It also reveals that the current system is prone to omit content words, which is mainly caused by alignment errors. In UC tasks, however, the frequency of "c: word order error" is much lower than that of ECn task, while the frequency of "b: lack of content words in translation" is much higher. This indicates that Uighur and Chinese have a more similar structure, but due to the rich morphology of Uighur, there are more alignment error and quantifier/temporal errors.

We show the distribution of the error types of the two systems in Figure 1 and Figure 2.

3.2 Statistical MT System vs. Rule-based MT System

In recent years, along with the success of the statistical MT system, rule-based MT system has been gradually fading away from the translation community. In this evaluation, we did a detailed comparision between this two kind of systems and the result is shown in Table [^5]

[^1]: http://nlp.ict.ac.cn/Admin/ckeditor/attached/file/20140310/20140310173732_36859.pdf
Figure 1: Distribution of Overall Error Type of $EC_n$ Task. n means no error.

Figure 2: Distribution of Overall Error Type of UC Task.

System | Loyalty | Fluency | BLEU-SBP
--- | --- | --- | ---
RB | 3.27 | 3.00 | 0.22
SB | 2.93 | 2.76 | 0.34
SB+SC | 3.10 | 2.97 | 0.35

Table 5: Manual and Automatic evaluation results of three systems in $EC_n$ task. RB denotes a rule-based system. SB is a statistical system, and SB+SC means statistical system with system combination technology.

The first and second column shows the manual evaluation results, we can see that rule-based system still have some advantage over statistical systems. We further analyze the result and plot the distributions of error types of the RB and SB systems in Figure 3 and Figure 4. We can see that rule-based system has a clear advantage in translating content words, resulting in a more complete translation and a higher manual evaluation score, while statistical system is trained to optimize BLEU score and makes less word selection errors.

Another interesting finding is that the system combination technology for statistical MT system brings a positive impact on both manual and automatic evaluation. The 4th row in Table 5 shows the performance of statistical system with system combination technology. We can see that both manual and automatic evaluation scores get a big boost with 1 BLEU-SBP point and about 0.2 points in Loyalty and Fluency scores.

4 Correlations between Automatic Evaluation Metrics

In this evaluation we use a variety of automatic evaluation metrics to evaluate all the systems and produce a large amount of evaluation scores, which enables us to further study the correlations between those automatic evaluation metrics. Eleven evaluation metrics are involved in most tasks including: 5-gram BLEU-SBP, METEOR, TER, 5-gram BLEU, 6-gram BLEU, 6-gram NIST, 7-gram NIST, GTM, mWER, mPER, and ICT. For each task, we calculate the Spearman Rank Correlation Coefficient (SRCC)
between the evaluation scores of two different metrics. The results of $EC_n$ and UC task are shown in Figure 5 and Figure 6. Each node denotes one or more metrics and the distance between them is based on their SRCC score. The orange double arrow connects the metrics with a higher SRCC score and the blue dotted line connects the metrics with relatively lower SRCC scores. Noted that if the SRCC score of two metrics is greater than 0.99, we merge them as one node in the figure. In $EC_n$ task, we can find that:

- Same metrics with different n-gram settings always have the highest correlation with each other, such as 5-gram and 6-gram BLEU, 6-gram and 7-gram NIST.
- BLEU-SBP has a very high correlation with BLEU.
- NIST, GTM, and mPER have a high correlation with each other.
- TER, mWER, and ICT have a low correlation with NIST and GTM.

Most of these findings are in accord with the common wisdom: metrics based on n-gram precision such as BLEU and BLEU-SBP have a high correlation. And metrics mainly based on edit-distance such as TER, mWER and ICT are much similar with each other. It’s also interesting to find that METEOR is kind of at the middle ground of all automatic metrics, since it incorporates a wide variety of linguistic knowledges. In UC task, the results is similar with the $EC_n$ tasks except that GTM has the highest correlation with NIST. And TER, mWER and ICT have a low correction with NIST and GTM.

5 Case Study: Automatic Evaluation vs. Manual Evaluation

In our analysis of the correlation between automatic and manual evaluation scores, we find an inconsistent case for rule-based MT system: Unlike statistical MT systems, the rule-based MT system has very different performances in automatic evaluation and manual evaluation. We show the SRCC between automatic evaluation metrics and manual evaluation metric with/without rule-based MT system in Table 6. We can see that rule-based system caused a drastic jump in SRCC, this denotes a obvious conflict between automatic and manual evaluation in rule-based system.
Figure 5: Correlations between Automatic Evaluation Metrics of $E_{C_n}$ task.

Figure 6: Correlations between Automatic Evaluation Metrics of UC task.
Table 6: SRCC between the Automatic Evaluation scores and Manual Evaluation scores in $E_{C_n}$ Task. The score in the left table is calculated based on results from all participating systems, whereas in the right we exclude the rule-based system to show its significant effect on SRCC.

<table>
<thead>
<tr>
<th></th>
<th>Loyalty</th>
<th>Fluency</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLEU5-SBP</td>
<td>0.33</td>
<td>0.35</td>
</tr>
<tr>
<td>METEOR</td>
<td>0.37</td>
<td>0.33</td>
</tr>
<tr>
<td>TER</td>
<td>0.33</td>
<td>0.37</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Loyalty</th>
<th>Fluency</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLEU5-SBP</td>
<td>0.91</td>
<td>0.93</td>
</tr>
<tr>
<td>METEOR</td>
<td>0.95</td>
<td>0.91</td>
</tr>
<tr>
<td>TER</td>
<td>0.91</td>
<td>0.95</td>
</tr>
</tbody>
</table>

One possible reason is the translation format of rule-based MT system participated in this evaluation caused this great performance difference: since the output of rule-based MT system sometimes contains optional words in parentheses and multiple choices of words in brackets, shown as "Org" in Figure 7. And this format will affect the n-gram precision in automatic evaluation.

\[ \text{Org: } \text{超人(已经)[起动;开始]一挑起一条重大提议他打算在一个举措里放弃他的美国国籍旨在把更多全球的影响和威望给他.} \]

\[ \text{Pos: } \text{超人已经起动一挑起一条重大提议他打算在一个举措里放弃他的美国国籍旨在把更多全球的影响和威望给他.} \]

Figure 7: Output sample of the rule-based MT system. "Org" denotes the original output of the system. "Pos" denotes the post-processed results.

To exclude the above side-effect, we carry out an additional experiment: we turn the output of rule-based system into standard translation format by removing redundant words, and evaluate the post-processed results (shown as "Pos" in Figure 7). The evaluation results are shown in Figure 8, where $S_1$ is the original rule-based system and $S_1^*$ is the same system with post-processing. We can see that format problem indeed causes a little drop in automatic evaluation score (about 0.5 points in BLEU-SBP). However, it doesn’t change the overall trend that rule-based system has very different performances in automatic and manual evaluation. This suggests that automatic evaluation metrics such as BLEU-SBP and METEOR have some bias against rule-based system which may result in a unilateral evaluation. And this mismatch further indicates that the current automatic evaluation metrics are still not good enough to reflect the real quality of the translation. We need to explore better automatic evaluation metrics which has a better correlation with manual evaluation metrics.
Figure 8: Evaluation Scores of different Systems in CWMT2013 $EC_1$ Task. $S_1$ = Rule-based system, $S_1^*$ = Rule-based system with Post-processing. $S_3, S_5, S_9$ are statistical systems.

6 Conclusions and Future Work

In this paper, we gave a detailed description of the CWMT2013 evaluation. Our analysis revealed some interesting correlations between different evaluation metrics. And the case study on rule-based system showed that automatic evaluation metrics such as BLEU-SBP and METEOR have some bias against rule-based system, causing the conflict in automatic and manual evaluation results. In the future evaluation, we will continue to explore better evaluation metrics and add more tasks on Chinese minority languages to promote the research in related fields.

Acknowledgement

We thank the three anonymous reviewers for helpful suggestions. The authors were supported by CAS Action Plan for the Development of Western China (No. KGZD-EW-501) and National Natural Science Foundation of China (Contract 61379086). Liu’s work was partially supported by the Science Foundation Ireland (Grant No. 07/CE/I1142) as part of the CNGL at Dublin City University. The views and findings in this paper are those of the authors and are not endorsed by the Chinese governments.

References


Combining Techniques from different NN-based Language Models for Machine Translation

†Jan Niehues  
*Alexander Allauzen  
*François Yvon  
†Alex Waibel  
†Karlsruhe Institute of Technology, Karlsruhe, Germany  
*Université Paris-Sud XI and LIMSI-CNRS, Orsay, France

Abstract

This paper presents two improvements of language models based on Restricted Boltzmann Machine (RBM) for large machine translation tasks. In contrast to other continuous space approach, RBM based models can easily be integrated into the decoder and are able to directly learn a hidden representation of the n-gram. Previous work on RBM-based language models do not use a shared word representation and therefore, they might suffer of a lack of generalization for larger contexts. Moreover, since the training step is very time consuming, they are only used for quite small corpora. In this work we add a shared word representation for the RBM-based language model by factorizing the weight matrix. In addition, we propose an efficient and tailored sampling algorithm that allows us to drastically speed up the training process. Experiments are carried out on two German to English translation tasks and the results show that the training time could be reduced by a factor of 10 without any drop in performance. Furthermore, the RBM-based model can also be trained on large size corpora.

1 Introduction

Language models are very important in many natural language processing tasks like, for example, machine translation and speech recognition. In most of these tasks, n-gram-based language models are successfully used. In this model the probability of a sentence is described as a product of the probabilities of the words given the previous words. For the conditional word probability a maximum likelihood estimation is used in combination with different smoothing techniques. Although this is often a very rough estimate, especially for rarely seen words, it can be trained very fast. This facilitates the use of huge corpora which are available for many language pairs.

Recently, language models based on neural network (NN) have successfully been used as an additional model in several tasks. In contrast to conventional n-gram language models, these models can cover longer contexts without a prohibitive increase of the number of parameters. Thereby, dependencies between words that do not occur in the direct neighborhood can be modeled. In state of the art language models contexts of up to 10 words are used.

Currently, most of these NN-based language models use feed-forward networks. These models can be trained on very large monolingual corpora. Furthermore, in most cases a shared word representation for all word positions is learned. Since the calculation of the language model probabilities is quite complex, often the language model can not be used during decoding,
but only in a rescoring step. Vaswani et al. (2013) presented an approach to also use feed-forward language models during decoding.

Niehues and Waibel (2012a) proposed a language model based on Restricted Boltzmann Machine (RBM). Since this model uses a quite simple layout, the probability computation is very fast and the language model can be used during decoding. In contrast, the training time of these models depends on the vocabulary size and therefore, the training time can be quite long. Furthermore, this model does not make use of a shared word representation, which can hinder its generalization power with large context.

Motivated by techniques developed for other NN-based language model, we tackle in this work these two issues of RBM-based language models. First, we propose tailored sampling algorithms to speed up the training process. Secondly, we introduce a factored representation of the weight matrix to learn a shared word representation. Furthermore, we analyzed the advantage of the RBM-based language model to use the language model during decoding.

The remaining paper is structured as follows. First we review related work and then provide in section 3 a brief overview of RBM-based language models. Section 4 describes the tailored sampling strategies while we describe how the shared word representation is integrated into the RBM layout in section 5. Afterwards we describe and discuss experimental results measured in terms of translation quality in section 6.

2 Related Work

A first approach to predict word categories using neural networks was presented in Nakamura et al. (1990). Later, Bengio et al. (2003) introduced neural networks for statistical language modeling. The authors described in detail an approach based on multi-layer neural networks and reported a significative perplexity reduction compared to conventional and class-based language models. In addition, they gave a short outlook to energy minimization networks.

An approach using multi-layer neural networks has successfully been applied to speech recognition by Schwenk and Gauvain (2002), Schwenk (2007) and Mikolov et al. (2010). One main problem of continuous space language models is the size of the output vocabulary in large vocabulary continuous speech recognition. A first way to overcome this is to use a short list. Recently, Le et al. (2011) presented a structured output layer neural network which is able to handle large output vocabularies by using a clustering tree to represent the output vocabulary. Motivated by the improvements in speech recognition accuracy as well as in translation quality, authors tried to use the neural networks also for the translation model in a statistical machine translation system. In Schwenk et al. (2007) as well as in Le et al. (2012) the authors modified the n-gram-based translation approach to use the neural networks to model the translation probabilities. In Vaswani et al. (2013), noisy-contrastive estimation was used to train the neural network. Therefore, the probabilities do not need to be normalized and the language model can be used during decoding.

A different approach also using Restricted Boltzmann Machines was presented in Mnih and Hinton (2007). However this approach exhibits the same complexity issue as feed-forward models. A head to head comparison between RBM and feed-forward language models can be found in Le et al. (2010). In Niehues and Waibel (2012a), another RBM-based language model was introduced. This approach differs from the one introduced in Mnih and Hinton (2007) by a simpler layout that allows us for a fast probability computation. This yields the integration of the model during the decoding step feasible. However, the training complexity heavily depends on the vocabulary size and this model can be trained on a limited amount of training data. In Dahl et al. (2012), a sampling method was presented to efficiently train restricted Boltzmann machines on word observations. This approach enables us to train large scale language models.
3 RBM-based language models

In this section we will briefly review the continuous space language models using RBMs introduced in Niehues and Waibel (2012a). A restricted Boltzmann machine is a generative stochastic artificial neural network that can learn a probability distribution over a set of visible variables that represent the observed features and a set of hidden variables. Variables are represented by neural units and grouped in two layers, one for the visible units and one for the hidden variables. For that purpose a RBM forms a bipartite graph as shown in Figure 1, where the connections between the visible and hidden layers have to be trained.

![Restricted Boltzmann Machine](image_url)

**Figure 1: Restricted Boltzmann Machine.**

The activation of the visible neurons will be determined by the input data. The standard input layer for neural network language models uses a 1-of-n coding to insert a word from the vocabulary into the network. This is a vector, where only the index of the word in the vocabulary is set to one and the rest to zero. The activation of the hidden units is usually binary and will be inferred from the visible units by using sampling techniques. Figure 2(a) is an example of the RBM-based language model.

To calculate the probability of a visible configuration $v$ we will use the definition of the free energy in a restricted Boltzmann machine with binary stochastic hidden units, which is

$$F(v) = - \sum_i v_i a_i - \sum_j \log(1 + e^{x_j})$$  \hspace{1cm} (1)$$

where $a_i$ is the bias of the $i$th visible neuron $v_i$ and $x_j$ is the activation of the $j$th hidden neuron.

The activation $x_j$ is defined as

$$x_j = b_j + \sum_i v_i w_{ij}$$  \hspace{1cm} (2)$$

where $b_j$ is the bias of the $j$th hidden neuron and $w_{ij}$ is the weight between visible unit $v_i$ and hidden unit $x_j$. Using these definitions, the probability of our visible configuration $v$ is

$$p(v) = \frac{1}{Z} e^{-F(v)}$$  \hspace{1cm} (3)$$

with the partition function $Z = \sum_v e^{-F(v)}$ being the normalization constant. Usually this normalization constant is not easy to compute, since it is a sum over an exponential amount of values. We know that the free energy will be proportional to the true probability of our visible vector. This is the reason for using the free energy as a feature in our log-linear model instead of the true probability. In order to use it as a feature inside the decoder we actually need to be able to compute the probability for a whole sentence. As shown in Niehues and Waibel (2012a) we can do this by summing over the free energy of all n-grams contained in the sentence.
3.1 Training

For training the restricted RBM-based language model (RBMLM) we used the Contrastive Divergence (CD) algorithm as proposed in Hinton (2002). In order to do this, we need to calculate the derivative of the probability of the training example given the weights

$$\frac{\delta \log p(v)}{\delta w_{ij}} = <v_i h_j>_{\text{data}} - <v_i h_j>_{\text{model}}$$

where $<v_i h_j>_{\text{model}}$ is the expected value of $v_i h_j$ given the distribution of the model. In other words we calculate the expectation of how often $v_i$ and $h_j$ are activated together, given the distribution of the data, minus the expectation of them being activated together given the distribution of the model, which will be calculated using Gibbs-Sampling techniques.

We need to calculate the hidden activation given the input data $p(h_i|v)$ for the first term. In this case only $n$ of the $V \times n$ visible units have non-zero values, where $n$ is the context size and $V$ the vocabulary size. Therefore, this can be done very fast. In each step of the Gibbs sampling, first the activation probability of the visible layer given the hidden layer $p(v^*|h)$ is calculated. Afterwards, the activation of the hidden variable given the sampled visible layer $p(h^*|v^*)$ needs to be calculated. In this case, not only $n$ visible units have to be considered, but all $V \times n$ ones. Consequently, the calculation is very slow especially for large vocabularies.

Usually many steps of Gibbs-Sampling are necessary to get an unbiased sample from the distribution, but in the Contrastive Divergence algorithm only one step of sampling is performed.

4 Sampling

In contrast to the time used for the calculation of the free energy, the time used for training the language model heavily depends on the vocabulary size. Therefore, the training of large scale language models is too time consuming. One solution to reduce the training time is to rely on a different sampling strategy.

4.1 Metropolis-Hasting sampling for word observations

A similar language model has been presented in Mnih and Hinton (2007). However, the training time issue due to the vocabulary size is similar. In Dahl et al. (2012), an approach based on Metropolis Hasting was presented to solve this problem. In contrast to considering all input units $v_i$ during Gibbs sampling, they first sample a list of words for every word position. Then only the visible units corresponding to these words are used during the calculation of $p(v^*|h)$ and $p(h^*|v^*)$.

While this approach uses probabilistic binary values for the input neurons, Hinton (2010) mention that it is advantageous to use the probabilities themselves. Therefore, we present a different approach for sampling which uses the probabilities themselves. Furthermore, we tried to include other information into the sampling process like the confusability of words in the translation model.

4.2 A tailored sampling approach

In order to minimize the effort for this calculation, we developed the following approximation. Instead of calculating all probabilities, we have a set of candidate words $C_j$ for every word position. Then only the probabilities $p(v^*_i|h)$ for $v_i \in C_0 \ldots C_n$ need to be calculated. Since the probabilities are calculated given the hidden activation, they should lead to similar n-grams. Therefore, most probabilities will be close to zero and some of them can be neglected. With this approximation, we no longer calculate all probabilities, but only the ones for the candidate
words $C_0 \ldots C_n$. Therefore, the time needed for the calculation no longer depends on $n \ast v$, but on $\sum_{i=1}^{n} |C_i|$. In this work, we investigate different possibilities to calculate the set of candidate words $C_i$. In a first approach, we sample the words using the unigram probabilities (unigram). We pre-calculate a set of candidates for every word and use the same set of words during training. In this case, we sample until we have $c$ distinct words.

In a second attempt, we re-sample for every training example (unigram.online) using the same probability distribution. Since this has to be done during training, we only sample $c$ times. Therefore, the number of words in the set can be lower than $c$.

Since we are using the language model in a translation system, it is more important that the language model can distinguish between words that are translations of the same source word than other words. Therefore, we use information from the translation model to select our samples (translation). We use the lexical translation probabilities $p(e|f)$ and $p(f|e)$ to calculate the translation confusability probability $p_c(e'|e) = \sum_{f \in F} p(e'|f) \ast p(f|e)$. We then use this probability to sample the set $C(e)$. Since the set of words, where the probability $p_c(e'|e)$ is non-zero, is often smaller than $c$, we fall-back to unigram probabilities, if we sample $e$ itself. This is also done in a preprocessing step until we sampled $c$ distinct words.

5 Shared Word Representation

![Diagram](image)

Figure 2: Different layouts for RBM-based language models

In the RBM-based language model as well as in other neural network language models, the input words are encoded by an 1-out-of-$n$ vector. In the RBM-based language model, this vector is then multiplied by the weight matrix $W$ of the size $N \ast H$, where $H$ is size of the hidden layer. An illustration is shown in Figure 2(a). Since many words occur quite rarely, some of the weights cannot not be estimated well.

In contrast, most other neural network language model use a shared word representation. In this case, the sparse word vector is first multiplied by a matrix $W$ shared over all word positions, resulting in a word representation vector. These representations are then multiplied by a position dependent weight matrix to form the input for the hidden layer.

In Mnih and Hinton (2007), a factorization of the weight matrix into a word dependent and context dependent part for the bi-log-linear model is suggested. Thereby, it is possible to also use a shared representation for the words.

We adapted this approach to the RBM-based language model. This results in the layout presented in Figure 2(b). Instead of multiplying the input vector by $W$, it is now first multiplied by $W$ and then by the context dependent part $C_i$. Therefore, the number of weights is no longer $n \ast V \ast H$, but $V \ast S + n \ast S \ast H$, where $S$ is the size of the shared word representation.

6 Experimental results

We evaluated the sampling and the shared word representation on two tasks translating from German to English. First, we used the RBM-based language model in a speech translation
system. Afterwards, the system was used to translate German news data into English.

### 6.1 System description

The speech translation system was trained on the European Parliament corpus, News Commentary corpus, the BTEC corpus and TED talks\(^1\). The data was preprocessed and compound splitting was applied for German. Afterwards the discriminative word alignment approach as described in Niehues and Vogel (2008) was applied to generate the alignments between source and target words. The phrase table was built using the scripts from the Moses package (Koehn et al., 2007). A 4-gram language model was trained on the target side of the parallel data using the SRILM toolkit (Stolcke, 2002). In addition we used a bilingual language model as described in Niehues et al. (2011).

Reordering was performed as a preprocessing step using POS information generated by the TreeTagger (Schmid, 1994). We used the reordering approach described in Rottmann and Vogel (2007) and the extensions presented in Niehues and Kolss (2009) to cover long-range reorderings, which are typical when translating between German and English. An in-house phrase-based decoder was used to generate the translation hypotheses and the optimization was performed using Minimum Error Rate Training (MERT) (Venugopal et al., 2005).

We tested the language models on two different sets. First, we optimized the weights of the log-linear model on a separate set of TED talks and also used TED talks for testing. The development set consist of 1.7K segments containing 16K words. As test set we used 3.5K segments containing 31K words. In addition, we optimized and tested the systems on a set of computer science lectures collected at a university.

The language models were tested on three different conditions. First, we used the baseline system, then we used a system, which has been adapted to the TED task by using an additional n-gram language model trained on the TED data. Finally, we also tested the language model in a translation system using phrase table adaptation as described in Niehues and Waibel (2012b). In summary, the language models were tested on 6 different sets and the reported results are the average BLEU scores on all test sets. The RBM-based language model was trained only on the TED data. If not stated differently, we used 32 hidden units and a context length of 4 words.

The system used for the news translation task, was trained on all the data that was available in the WMT 2014. The system was trained similarly. A detailed description of the system can be found in Herrmann et al. (2014). The system was optimized on newstest2012 and tested on newstest2013.

### 6.2 Sampling

In a first set of experiments we analyzed the correlation of translation performance and the sample size. Therefore, we used the unigram sampling approach and the results are summarized

<table>
<thead>
<tr>
<th>Sampling size</th>
<th>BLEU Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>No RBMLM</td>
<td>25.16</td>
</tr>
<tr>
<td>No Sampling</td>
<td>25.42</td>
</tr>
<tr>
<td>5</td>
<td>25.15</td>
</tr>
<tr>
<td>10</td>
<td>25.01</td>
</tr>
<tr>
<td>50</td>
<td>25.23</td>
</tr>
<tr>
<td>100</td>
<td>25.25</td>
</tr>
<tr>
<td>1000</td>
<td>25.46</td>
</tr>
</tbody>
</table>

Table 1: Results for different sample sizes

\(^1\)http://www.ted.com
Table 2: Results for 4-gram RBM-based language models on TED

<table>
<thead>
<tr>
<th>Model</th>
<th>Sampling</th>
<th>Size</th>
<th>1-Layer</th>
<th>Shared</th>
</tr>
</thead>
<tbody>
<tr>
<td>No RBMLM</td>
<td></td>
<td></td>
<td>25.16</td>
<td></td>
</tr>
<tr>
<td>SOUL</td>
<td></td>
<td></td>
<td>25.56</td>
<td></td>
</tr>
<tr>
<td>RBMLM</td>
<td>No Sampling</td>
<td>25.42</td>
<td>25.36</td>
<td></td>
</tr>
<tr>
<td>unigram</td>
<td>100</td>
<td>25.25</td>
<td>25.01</td>
<td></td>
</tr>
<tr>
<td>unigram.online</td>
<td>100</td>
<td>25.37</td>
<td>25.49</td>
<td></td>
</tr>
<tr>
<td>translation</td>
<td>100</td>
<td>25.36</td>
<td>25.26</td>
<td></td>
</tr>
<tr>
<td>RBMLM</td>
<td>unigram</td>
<td>1000</td>
<td>25.46</td>
<td>25.29</td>
</tr>
<tr>
<td></td>
<td>unigram.online</td>
<td>1000</td>
<td>25.33</td>
<td>25.30</td>
</tr>
<tr>
<td></td>
<td>translation</td>
<td>1000</td>
<td>25.37</td>
<td>25.4</td>
</tr>
</tbody>
</table>

in Table 1

We are not able to train useful language models with a sample size of five or ten candidates. In this case, the translation performance can not be improved over the baseline using no RBM-based language model. If we increase the sample size to 50 or 100 words, the language model slightly improves the translation quality, but the performance is still worse than the system using no sampling. When using a sample size of 1000 candidates, the same performance as with the language model using no sampling can be achieved.

After having an initial impression of appropriate samples sizes, we evaluated the performance of the different sampling approaches on the TED translation task. The results are summarized in the left column titled as "1-layer" of Table 2.

The first three lines describe different baseline systems. In the first system, we do not use any neural network language models. In the second system, we used a SOUL language model. This is a feed-forward neural network using a structured output layer (Le et al., 2011). This improved the performance by 0.4 BLEU points on average in all the systems. Finally, as a third baseline system, we trained an RBM-based language model on the TED data. In this case, we can improve the performance by 0.3 BLEU points. Hence, we get similar improvements by using the RBM-based language model and the SOUL language model.

After comparing the baseline systems, we tested the different sampling methods. First, we tested all three sampling techniques using a sample size of 100. The three techniques perform very similar leading to improvements of 0.1 to 0.2 BLEU points. Hence, the performance drops at most by only 0.2 BLEU points. The unigram approach for sampling performs slightly worse in this configuration compared to the other two approaches. When accepting this drop in performance, the training time of the language model drops by from 12 hours to 0.1 to 0.2 hours. Therefore, it is possible to train the model also on corpora that are larger than the quite small TED corpus. Furthermore, the training time no longer depends on the vocabulary size. While the vocabulary size quite small for the TED corpus with 28K words, for many other tasks, it is significantly larger.

Afterwards, we also tested the techniques using a sample size of 1000. In this case, the training time is still faster, reducing the training time to roughly 0.8 hours. In this case all three systems perform very similar to the one using no sampling. The performance drops by at most 0.1 BLEU points. So it is possible to reach approximately the same performance than with the baseline RBM-based language model, but using only one tenth of the training time. Again, all methods to sample the words perform quite similarly.

In conclusion, it is possible to achieve nearly the same performance using sampling than the baseline training method by using only one tenth of the training time. Furthermore, even if we reduce the training time even more by a factor of 10, we have only a slight drop in
Table 3: Results for different shared word sizes

<table>
<thead>
<tr>
<th>SharedWord</th>
<th>Hidden</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>32</td>
<td>25.30</td>
</tr>
<tr>
<td>256</td>
<td>32</td>
<td>25.38</td>
</tr>
<tr>
<td>512</td>
<td>32</td>
<td>25.40</td>
</tr>
<tr>
<td>1024</td>
<td>32</td>
<td>25.43</td>
</tr>
<tr>
<td>512</td>
<td>64</td>
<td>25.34</td>
</tr>
<tr>
<td>512</td>
<td>128</td>
<td>25.33</td>
</tr>
</tbody>
</table>

After we analyzed the influence of the different sampling techniques on the translation quality, we concentrated on evaluating the shared word representation. The results are shown in the right column of Table 2. In these experiments, a shared word representation consists of 512 neurons.

If we use no sampling, the shared representation performs similar, but slightly worse than the 1-layer typology. For the unigram sampling approach, the shared representation performs always worse than the 1-layer typology. For the other two sampling techniques, we see a similar performance for both network layouts.

In the experiments in Table 3, we analyzed the effect of the size of the shared word representation and of the hidden layer. In all the experiments we used translation sampling approach with a sample size of 1000.

We can improve the translation quality by increasing the size of the shared word representation. But starting from a size of 256 the improvements in translation quality are very small. Since the training time depends on the size of the layer, we used a size of 512 for the remaining experiments. If we compare the system using different number of neurons in the hidden layer, the translation quality cannot be improved by using a larger hidden layer.

In summary, we are able to reach a similar performance by using the shared word representation as by the 1-layer typology. But it is not possible to outperform the performance of the 1-layer typology.

6.4 N-Gram Length

After we performed a detailed analysis for 4-gram RBM-based language models, we continue with analyses the performance of the 8-gram language models. The results for these experiments are summarized in the two left most columns in Table 4.

In a first experiment, we used the SOUL language model using 4-gram and 8-gram context for comparison. We can see that it is possible to improve the performance of the SOUL LM by an additional 0.1 BLEU points leading to an improvement of 0.5 BLEU points over the baseline system.

We compared the 4-gram and 8-gram RBM-based language models on 6 different configurations. First, we compared the 1-layer layout and the layout using shared word representation. Furthermore, we tested both typologies using 3 different sampling techniques. We used the unigram, online sampling technique using sample sizes of 100 and 1000 words and the sampling using translation probabilities with 1000 samples.

If we first look at the language models using a 1-layer layout, no improvements can be achieved by using a 8-gram language model compared to the 4-gram language model. Only small changes of less than 0.1 BLEU points can be seen.
<table>
<thead>
<tr>
<th>Layout</th>
<th>Sampling</th>
<th>Sample size</th>
<th>Rescoring 4-gram</th>
<th>Rescoring 8-gram</th>
<th>Decoder 4-gram</th>
<th>Decoder 8-gram</th>
</tr>
</thead>
<tbody>
<tr>
<td>No RBMLM</td>
<td>SOUL</td>
<td></td>
<td>25.16</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-Layer</td>
<td>unigram.online</td>
<td>100</td>
<td>25.37</td>
<td>25.30</td>
<td>25.55</td>
<td>25.36</td>
</tr>
<tr>
<td>1-Layer</td>
<td>unigram.online</td>
<td>1000</td>
<td>25.33</td>
<td>25.41</td>
<td>25.27</td>
<td>25.46</td>
</tr>
<tr>
<td>1-Layer</td>
<td>translation</td>
<td>1000</td>
<td>25.37</td>
<td>25.38</td>
<td>25.64</td>
<td>25.24</td>
</tr>
<tr>
<td>Shared</td>
<td>unigram.online</td>
<td>100</td>
<td>25.39</td>
<td>25.56</td>
<td>25.51</td>
<td>25.33</td>
</tr>
<tr>
<td>Shared</td>
<td>unigram.online</td>
<td>1000</td>
<td>25.30</td>
<td>25.48</td>
<td>25.41</td>
<td>25.55</td>
</tr>
<tr>
<td>Shared</td>
<td>translation</td>
<td>1000</td>
<td>25.4</td>
<td>25.48</td>
<td>25.38</td>
<td>25.23</td>
</tr>
</tbody>
</table>

Table 4: Results for 8-gram LMs

In the lower part of the table the results for the RBMs using shared word representation are shown. In this cases the performance can be improved when going from an 4-gram context to an 8-gram context. This improvement is consistent for all sampling techniques. Similar to the SOUL language model, the improvements are between 0.1 and 0.2 BLEU points.

In conclusion, we can only improve the performance by increasing the context from four to eight words, when using the shared word representation. When comparing the performance of the RBM-based and the SOUL language models, the SOUL language model performs roughly 0.1 BLEU points better than the RBM-based language model.

6.5 Language Model Integration

In contrast to other neural network language models like the SOUL model, the RBM-based language model has a quite simple structure. Therefore, it is possible to calculate the probabilities very fast and the language model can be directly integrated into the decoder in contrast to using it only during resoring. Therefore, we investigated the effect of using the language model during decoding. The results are summarized in the right columns of Table 4.

For the language models using a context of 4-words, the performance can be improved in 4 out of 6 cases by 0.1 to 0.3 BLEU points. In the cases where the performance is worse, the difference is very small, so the performance stays mainly the same. Hence, it is possible to improve the performance by directly integrating the language model into the decoder.

If we look at the 8-gram language model, the integration no longer leads to these improvements. The performance is better when using the language model during decoding, but in all cases by less than 0.1 BLEU points. Furthermore, in three cases the performance drops, in some cases even by 0.2 BLEU points.

In summary, it was possible to improve the performance when using a 4-gram language model during decoding compared to using it only during restoring. In contrast, when using 8-gram language models, this is no longer the case.

6.6 Large-Scale Language Models

After we did a detailed analysis of the different language models on the quite small data of the TED corpus, we also used the language model on a larger scale. Therefore, we trained the language models on the target side of the parallel data of the German-English translation task of the WMT. The results for these experiments are shown in Table 5.

As shown in the table, the SOUL model could improve the translation quality by around 0.4 BLEU points. In this case, we used KBMira optimization instead of MERT, since for this model it performed significant better. In contrast to the TED translation task, the 4-gram and 8-gram language models as well as the 1-layer or Shared Word Representation layout perform similar.
<table>
<thead>
<tr>
<th>Method</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>27.02</td>
</tr>
<tr>
<td>SOUL</td>
<td>27.42</td>
</tr>
<tr>
<td>1-Layer 4-gram</td>
<td>27.20</td>
</tr>
<tr>
<td>1-Layer 8-gram</td>
<td>27.22</td>
</tr>
<tr>
<td>Shared 4-gram</td>
<td>27.19</td>
</tr>
<tr>
<td>Shared 8-gram</td>
<td>27.15</td>
</tr>
</tbody>
</table>

Table 5: Results for NN language model on WMT task

All this language model could improve the translation quality by 0.1 to 0.2 BLEU points. So it is possible to improve the translation quality using the RBM-based language model, but it not yet performing as well as the SOUL language model.

7 Conclusion

This paper describes the integration of two techniques motivated by other NN-based language models into RBM-based language models. In a first step, we suggested different sampling techniques in order to speed up the training process. Using these techniques, the training time no longer depends on the vocabulary size. Thereby, the training time could be reduced by a factor of 10 without loss in performance. Furthermore, when reducing the training time from 12 hours to 0.2 hours, the performance drops only slightly. With this technique it is possible to train RBM-based language models even on large data sets like the WMT data.

Furthermore, we used a shared word representation by factorizing the weight matrix in an RBM-based language model. On shorter context sizes of 4 words this leads to the same performance as the default layout of an RBM. But for longer context sizes of 8 words, small improvements could be seen by using the shared word representation.

By combining these two contributions, RBM language models can achieved state of the art performance and can be efficiently training on large datasets. In addition, we showed that RBM-based language models can yield similar performance to the well-known SOUL language models. Moreover, one advantage of RBM-based language models is their easy integration into the decoder. We showed that, in our experiments, the performance could only be improved when using shorter contexts of 4 words.

In future work, we will analyze how we could also leverage longer context during decoding. Furthermore, we will investigate the use of deeper neural networks.

Acknowledgments

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 287658. This work was partially financed by the DGA RAPID project RAPMAT (http://www.limsi.fr/tlp/rapmat.html).

References


Niehues, J. and Waibel, A. (2012b). Detailed analysis of different strategies for phrase table adaptation in
smt. In Proceedings of the Tenth Conference of the Association for Machine Translation in the Americas
(AMTA 2012), San Diego, California, USA.

distortion model. In Proceedings of the 11th International Conference on Theoretical and Methodological
Issues in Machine Translation (TMI-07), Skövde, Sweden.

International Conference on New Methods in Language Processing, Manchester, UK.

518.

speech recognition. In Proceedings of the IEEE International Conference on Acoustics, Speech, and
Signal Processing (ICASSP 2002), Orlando, Florida, USA.

Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Com-
putational Natural Language Learning (EMNLP-CoNLL 2007), Prague, Czech Republic. Association
for Computational Linguistics.

Conference of Spoken Language Processing (ICSLP 2002), Denver, Colorado, USA.

models improves translation. In Proceedings of the Conference on Empirical Methods in Natural Lan-
guage Processing (EMNLP 2013), pages 1387–1392, Seattle, USA.

statistical machine translation. In Proceedings of the Workshop on Data-drive Machine Translation and
Beyond (WPT-05), Ann Arbor, Michigan, USA.

Katsuhito Sudoh
NTT Communication Science Laboratories, Seika-cho, Soraku-gun, Kyoto 619-0237, Japan / Graduate School of Informatics, Kyoto University, Yoshida-hommachi, Sakyu-ku, Kyoto 606-8501, Japan

Masaaki Nagata
NTT Communication Science Laboratories, Seika-cho, Soraku-gun, Kyoto 619-0237, Japan

Shinsuke Mori
Academic Center for Computing and Media Studies, Kyoto University, Yoshida-hommachi, Sakyu-ku, Kyoto 606-8501, Japan

Tatsuya Kawahara

Abstract

This paper presents a Japanese-to-English statistical machine translation system specialized for patent translation. Patents are practically useful technical documents, but their translation needs different efforts from general-purpose translation. There are two important problems in the Japanese-to-English patent translation: long distance reordering and lexical translation of many domain-specific terms. We integrated novel lexical translation of domain-specific terms with a syntax-based post-ordering framework that divides the machine translation problem into lexical translation and reordering explicitly for efficient syntax-based translation. The proposed lexical translation consists of a domain-adapted word segmentation and an unknown word transliteration. Experimental results show our system achieves better translation accuracy in BLEU and TER compared to the baseline methods.

1 Introduction

Machine translation (MT) is now widely used for various languages and fields. One typical and important use of MT is assimilation, understanding the content of documents written in foreign languages. Among such documents, patents are very important technical documents written in official languages of the countries to which they are filed. MT of the patents is beneficial for industrial use, such as technical surveys in other countries.

One practical problem in the patent MT is its domain dependence. Patents usually include long sentences and many technical terms in various fields, which are distinct difference from general-purpose MT. This work focuses on statistical MT (SMT) for patents, from Japanese to English. It is more difficult than English-to-Japanese in: 1) long distance reordering, and 2) word segmentation and lexical translation of domain-specific terms. The first problem is due to large syntactic differences between Japanese and English. Although the reordering in the English-to-Japanese direction can be solved effectively by very simple heuristics called Head
Finalization (Isozaki et al., 2010), the reordering in Japanese-to-English is not so straightforward. The second problem results from the Japanese orthography in which there are no explicit word boundaries. General-purpose word segmenters often fail to segment the domain-specific words and those words are translated incorrectly or remain untranslated. Some domain-specific words cannot be translated as unknown words even if they are segmented correctly, due to limited SMT training data. The first problem has been addressed by a syntax-based approach (Yamada and Knight, 2002; Galley et al., 2004; Zollmann and Venugopal, 2006), while most previous studies did not deal with the second problem. Since the lexical translation also affects the reordering based on a lexicalized reordering model and an n-gram language model, considering both problems is important for an overall SMT system. The goal of this work is to improve the Japanese-to-English patent SMT by tackling both problems at the same time. The domain-specific words have important roles in the patents and should be translated carefully for meaningful translations. We propose a novel domain adaptation method for the word segmentation, using effective features derived from a large-scale patent corpus. We also incorporate machine transliteration for the unknown Japanese words written in katakana (Japanese phonograms), bootstrapped from the parallel corpus (Sajjad et al., 2012; Sudoh et al., 2013a).

Our SMT system integrates these techniques with a post-ordering framework (Sudoh et al., 2013b), which divides the SMT problem explicitly into two sub-problems of the lexical translation and the reordering. In the post-ordering framework, the lexical translation precedes the reordering, different from pre-ordering in which the reordering precedes the lexical translation (Xia and McCord, 2004; Isozaki et al., 2010). An advantage of the post-ordering is that it is easy to integrate the domain-adapted word segmentation and the unknown word transliteration in its lexical translation step and that the reordering can use the improved lexical translation results. If we are to do the same thing in the pre-ordering, we need domain adaptation of its Japanese syntactic parser in addition to the word segmenter, and have to integrate the transliteration process with the SMT decoder as Durrani et al. (2014). Our system shows better translation accuracy in BLEU and TER than baseline methods in Japanese-to-English patent translation experiments.

2 Related Work

The patent MT between Japanese and English has been studied actively on shared tasks in NTCIR (Fuji et al., 2008, 2010; Goto et al., 2011, 2013). Recent important achievements in these studies are on the reordering problem especially in English-to-Japanese direction. Isozaki et al. (2010) proposed a very simple but effective rule-based syntactic pre-ordering method called Head Finalization. It is very effective for the long distance reordering. On the other hand, the Japanese-to-English direction is more difficult due to the lack of such simple rules. Hoshino et al. (2013) proposed an effective rule-based syntactic pre-ordering based on predicate-argument structures. Sudoh et al. (2013b) proposed a different approach called post-ordering for the Japanese-to-English patent MT, and achieved high translation performance by an efficient syntax-based translation. Our system uses the latter approach based on English syntax rather than the former one based on Japanese syntax. This is because our word segmentation adaptation can be applied directly to it without the Japanese parser adaptation as described earlier. General-purpose Japanese parsers do not work well in the patent domain, and their domain adaptation is not easy without a treebank in the patent domain. In this work, we use an English parser Enju¹ that includes a parsing model for biomedical articles and works relatively well in the patent domain. The domain adaptation of syntactic parsers is another important problem for further studies, but it is beyond the scope of this paper.

The problem of word segmentation has not been addressed in previous studies on the patent MT, and general-purpose Japanese morphological analyzers have been used in common. In Chi-

¹http://www.nactem.ac.uk/tsujii/enju/index.html
Chinese, domain adaptation of the word segmentation to the patent domain has been studied (Guo et al., 2012). Their domain adaptation introduces features extracted from a large number of unlabeled patent data into a supervised word segmentation framework based on a labeled corpora in a different (newspaper) domain. They reported the improvement in word segmentation, and did not report its effect on the patent MT. Their work can be seen an application of a semi-supervised learning method (Sun and Xu, 2011) to the domain adaptation. Such an approach is appropriate for the patent domain where a huge number of patent documents are publicly available. We extend their domain adaptation by more effective and easy-to-use features, and also incorporate additional Japanese-oriented features to improve the Japanese word segmentation in the patent domain.

With respect to the relation between word segmentation and SMT, Chang et al. (2008) reported consistency and granularity of word segmentation is important in Chinese-to-English MT and modified their Chinese word segmenter to optimize the translation performance. Dyer et al. (2008) and Zhang et al. (2008) used multiple word segmentation results to overcome the problem of different word segmentation standards. Xu et al. (2008) optimized Chinese word segmentation for Chinese-to-English SMT using an extended Bayesian word segmentation method with bilingual correspondence. These studies aim to optimize word segmentation using bilingual correspondence and are different from the domain adaptation.

Machine transliteration is an important problem for translating names and other imported words (Knight and Graehl, 1998). Conventional methods need to prepare parallel transliteration pairs for training. Sajjad et al. (2012) proposed an unsupervised transliteration mining from standard parallel corpora for bootstrapping machine transliteration from the parallel corpora. Durrani et al. (2014) integrated it with a SMT framework. The transliteration process in our SMT system is a character-based SMT basically same as Durrani et al. (2014), but uses an extended transliteration mining method for Japanese compound words (Sudoh et al., 2013a).

3 System Overview

Our Japanese-to-English patent SMT is based on large-scale language resources in the patent domain. This work uses NTCIR PatentMT dataset (Goto et al., 2011, 2013) including a Japanese-English parallel corpus of 3.2 million sentences and monolingual corpora of more than 300 million sentences of Japanese and English. The parallel corpus was developed by an automatic sentence alignment over patent documents in the Japan Patent Office and the United States Patent and Trademark Office (Utiiyama and Isahara, 2007). The workflow of our SMT system is illustrated in Figure 1. The translation is divided into the following four processes.

1. Japanese word segmentation using a patent-adapted word segmentation model
2. Translation into an intermediate language, Head Final English (HFE), by a monotone phrase-based SMT
3. Transliteration of untranslated Japanese katakana words (i.e. unknown words in the previous process) into English words, by a monotone phrase-based SMT in the character level
4. Post-ordering into English by a syntax-based SMT

Here, HFE is Japanese-ordered English, which was proposed by Isozaki et al. (2010) for English-to-Japanese translation. Figure 2 shows an example of a HFE sentence and corresponding English and Japanese sentences. The word order of the HFE sentence is almost the same as the Japanese one, while that of the original English sentence is different from them largely in the verb phrase. Using HFE as the intermediate language, Japanese-to-English SMT is decomposed into two sub-problems: monotone lexical translation and reordering. For these two sub-problems, we basically follow the work by Sudoh et al. (2013b). They used phrase-based
SMT for the first lexical translation problem to utilize its advantage on the use of phrasal contexts, and a syntax-based SMT with target language syntax for the second reordering problem to utilize its advantage on the long distance reordering. The transilliteration process gives lexical translation of the unknown katakana words as the monotone character-level SMT. Katakana is often used for transcribing imported words (especially from Western languages) in Japanese, so the transliteration helps to reduce unknown words in the following HFE-to-English SMT.

The models are trained as follows. The word segmentation model is trained using a general domain labeled (word segmented) corpus and a patent unlabeled (not word segmented) corpus, by a semi-supervised learning described later in the next section. The transilliteration model is trained using transilliteration pairs mined from the parallel corpus by the method of Sudoh et al. (2013a). The translation models used in the monotone phrase-based SMT and syntax-based SMT are trained using the parallel corpora. Since HFE can be generated automatically by the Head Finalization rules, we can easily obtain the parallel corpus of three languages: Japanese, English, and HFE. The language models are trained using the monolingual corpora.
Domain Adaptation of Japanese Word Segmentation for Patents

We aim to improve the Japanese-to-English translation performance further by the word segmentation adaptation for patent-specific words and technical terms. We use the large-scale monolingual Japanese patent corpora for the domain adaptation, by the semi-supervised approach as Sun and Xu (2011) and Guo et al. (2012). There are also active learning-based supervised domain adaptation (Tsuboi et al., 2008; Neubig et al., 2011) and unsupervised word segmentation (Kempe, 1999; Kubota Ando and Lee, 2003; Goldwater et al., 2006, and many others) approaches, but the semi-supervised approach is expected to be effective; the active learning method is not easy to utilize for such large-scale corpora and the unsupervised method is not so accurate as existing supervised word segmenters.

4.1 Baseline Word Segmentation based on Conditional Random Fields

We use a character-based word segmenter based on CRFs (Peng et al., 2004; Tseng et al., 2005). It solves a character-based sequential labeling problem. In this work we employ four classes B, M, E (beginning/middle/end of a word), and S (single-character word)\(^2\), as Sun and Xu (2011).

Our baseline features follow the work of Japanese word segmentation by Neubig et al. (2011): label bigrams, character n-grams (n=1, 2), and character type n-grams (n=1, 2, 3). We use the n-gram features within \([i-2, i+2]\) for classifying the word at the position \(i\). The character types are kanji, katakana, hiragana, digits, roman characters, and others.

4.2 Conventional Method: Word Segmentation Adaptation using Accessor Variety

Sun and Xu (2011) and Guo et al. (2012) used Accessor Variety (AV) (Feng et al., 2004) derived from unlabeled corpora as word segmentation features. AV is a word extraction criterion from un-segmented corpora, focusing on the number of distinct characters appearing around a string. The AV of a string \(x_n\) is defined as

\[
AV(x_n) = \min \{AV_L(x_n), AV_R(x_n)\},
\]

where \(AV_L(x_n)\) is the left AV (the number of distinct predecessor characters) and \(AV_R(x_n)\) is the right AV (the number of distinct successor characters). The AV-based word extraction is based on an intuitive assumption; a word appears in many different context so that there is a large variation of its accessor characters. Intuitively this assumption seems true. Figure 3 shows an example of the AV calculation for a character “前”. If the character is a word by itself, it is expected to appear in many different context so that the AV values become large by different accessor characters. This kind of information benefits the supervised word segmentation because large-scale corpora derive word boundary clues for many different character sequences that are not included in the labeled training corpus. Many technical terms in the patent domain do not appear in the general-domain labeled corpus but can be segmented using this kind of information as features in the CRF-based word segmentation. To obtain reliable AV values for many different technical terms, we need to use unlabeled corpora in the patent domain as large as possible. Here note that the AV values are frequency-based and proportional to the corpus size in general. Previous studies use several frequency classes with corresponding threshold values tuned according to the corpus, but it is not straightforward to determine appropriate classes and threshold values.

Sun and Xu (2011) used the following features based on the left and right AVs of character n-grams for classifying \(x_i\), which imply word boundaries around \(x_i\), as illustrated in Figure 4.

- Left AV of n-gram starting from \(x_i\): \(AV_L(x_i, \ldots, x_{i+n-1})\)

\(^2\)Guo et al. (2012) used six classes including B2, B3 (second and third character in a word) proposed by Zhao et al. (2006) for Chinese word segmentation. This paper uses the four classes, because the six classes did not improve the word segmentation accuracy in our pilot test.
character to be classified

Figure 3: Example of accessor variety (AV) and branching entropy (BE) for a character “前”.

- Right AV of n-gram ending with $x_{i-1}$: $AV_R(x_{i-n}, \ldots, x_{i-1})$
- Left AV of n-gram starting from $x_{i+1}$: $AV_L(x_{i+1}, \ldots, x_{i+n})$
- Right AV of n-gram ending with $x_i$: $AV_R(x_{i-n+1}, \ldots, x_i)$

Sun and Xu (2011) classified the AV values into frequency classes by frequency thresholds, and used them as binary bucket features. Guo et al. (2012) used different relative frequency classes: H, M, L for top 5%, between top 5% and 20%, below top 20%. This kind of frequency-based grouping quantizes the AV values. The thresholds and the number of the classes are tuned using some held-out data (Sun and Xu, 2011) or chosen empirically (Guo et al., 2012). Such a tuning is not easy in general, especially with a large number of the classes. We follow the relative frequency classes of Guo et al. (2012) in the following experiments.

4.3 Proposed Word Segmentation Adaptation Method

We propose a word segmentation adaptation method using two additional novel types of features: branching entropy (BE) features and pseudo-dictionary (PD) features. Our semi-supervised learning framework is the same as Sun and Xu (2011) and Guo et al. (2012). The BE features are practically useful because of the probabilistic attribute of the BE, and the PD features reflect characteristics of Japanese compound words.

4.3.1 Branching Entropy Features

The BE (Jin and Tanaka-Ishii, 2006) is a different word boundary clue based on probabilistic uncertainty of accessor characters. Jin and Tanaka-Ishii (2006) used the BE for unsupervised Chinese word segmentation. Their approach is based on an intuitive assumption; the uncertainty of successive characters is large at a word boundary. The uncertainty of the successive character $X$ after a given string $x_n = x_1 \ldots x_n$ of the length $n$ can be measured by the BE as the local conditional entropy of $X$ with $X_{n}$ instantiated:

$$H(X|X_n = x_n) = - \sum_{x \in V_x} P(x|x_n) \log P(x|x_n),$$

where $X_n$ is the context of the length $n$, $V_x$ is a set of characters. Jin and Tanaka-Ishii (2006) used the BE around character n-grams: left BE $H_L(x_n)$ for predecessor characters and right
BE \( H_R(x_n) \) for successor characters. Figure 3 also shows an example of the BE calculation. The left and right BE values are slightly different due to the different distributions of predecessor and successor characters. Even if the number of distinct accessor characters is large, the probabilistic certainty varies with their variance and is not necessarily large. Another important advantage of the BE is its probabilistic attribute. The uncertainty of accessor characters represented by a certain BE value is basically the same even for different corpus sizes, while the AV values increase with the corpus size in general.

The BE features are binary bucket features based on rounded integer values of the left and right BEs of character n-grams, similarly defined as the AV features illustrated in Figure 4. This simple quantization is motivated by the probabilistic attribute of the BE.

- Left BE of \( n \)-gram starting from \( x_i \): \( H_L(x_i, \ldots, x_{i+n-1}) \)
- Right BE of \( n \)-gram ending with \( x_{i-1} \): \( H_R(x_{i-n}, \ldots, x_{i-1}) \)
- Left BE of \( n \)-gram starting from \( x_{i+1} \): \( H_L(x_{i+1}, \ldots, x_{i+n}) \)
- Right BE of \( n \)-gram ending with \( x_i \): \( H_R(x_{i-n+1}, \ldots, x_i) \)

### 4.3.2 Pseudo-dictionary Features

We additionally use Japanese-oriented heuristic word boundary clues, based on characteristics of Japanese compound words. Compound words in Japanese patents are usually written in kanji (for Japanese- or Chinese-origin words) or katakana (for imported words from Western languages). Most of their component words are also used individually and in different compound words. For example, a katakana word “εςϜ” (stem) is used in many compound words such as “ΩʔεςϜ” (key stem) and “εςϜηϧ” (stem cell). Appearance of a distinct katakana sequence “εςϜ” implies word boundaries between “Ωʔ” and “εςϜ” and between “ες” and “ηϧ”. Such a word boundary clue may help to identify component words appearing in different contexts. The motivation of these intuitive word boundary clues based on the character type is similar to the use of punctuations as reliable word boundaries by Sun and Xu (2011).

To include such information, we use distinct kanji and katakana sequences as pseudo-dictionary entries. The definition of the pseudo-dictionary features follows the dictionary word features used in Japanese morphological analyzer KyTea\(^3\): whether or not the character is in the beginning/middle/end of one of the dictionary words of a certain length. An example of the pseudo-dictionary features is shown in Figure 5. The character “ス” in the example has a feature “L3_katakana”, representing the character is located at the leftmost position of a matched katakana pseudo-dictionary word of the length of three characters “ステム”. We use two distinguished pseudo-dictionaries for kanji and katakana for the PD features. Short sequences whose length is shorter than 2 characters for kanji and 3 characters for katakana are excluded from the pseudo-dictionaries. The length of the long pseudo-dictionary words exceeding five characters is labeled as “5+” to mitigate feature sparseness.

\(^3\)http://www.phontron.com/kytea/method.html
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Type</th>
<th>#sentences</th>
<th>#Ja characters</th>
<th>#words</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCCWJ</td>
<td>Labeled (Training)</td>
<td>53,899</td>
<td>1,810,675</td>
<td>1,242,137</td>
</tr>
<tr>
<td>BCCWJ(_{UL})</td>
<td>Unlabeled</td>
<td>6,017,627</td>
<td>185,289,168</td>
<td>n/a</td>
</tr>
<tr>
<td>NTCIR</td>
<td>Unlabeled</td>
<td>3,191,228</td>
<td>214,963,715</td>
<td>n/a</td>
</tr>
<tr>
<td>PatentJP</td>
<td>Unlabeled</td>
<td>537,494,485</td>
<td>42,175,165,488</td>
<td>n/a</td>
</tr>
<tr>
<td>Test(_{Patent})</td>
<td>Labeled (Test)</td>
<td>2,000</td>
<td>127,825</td>
<td>81,481</td>
</tr>
<tr>
<td>Test(_{BCCWJ})</td>
<td>Labeled (Test)</td>
<td>6,406</td>
<td>201,080</td>
<td>135,664</td>
</tr>
</tbody>
</table>

Table 1: Corpus statistics for word segmentation experiments.

5 Evaluation

We conducted experiments using the NTCIR PatentMT data to investigate the performance of our Japanese-to-English patent SMT system. We evaluated the word segmentation itself in addition to the overall Japanese-to-English translation, to see the effect of the proposed word segmentation adaptation on the translation results.

5.1 Evaluation of Word Segmentation

We evaluated the word segmentation accuracy by the proposed word segmentation adaptation and compared it with those by other methods.

5.1.1 Setup

We implemented a word segmenter using the features described in the previous section, with CRFsuite\(^4\) and its default hyperparameters. We used CORE data of Balanced Corpus of Contemporary Written Japanese (Maekawa, 2007) as the labeled general domain corpus for training the word segmentation model\(^5\), and split them for training (BCCWJ) and test (Test\(_{BCCWJ}\)) sets by about 9:1. For unlabeled corpus, we used its non-CORE portion (BCCWJ\(_{UL}\)), the Japanese portion of NTCIR-9 PatentMT (Goto et al., 2011) Japanese-English bitext (NTCIR), and Japanese monolingual patent corpus provided for NTCIR-9 PatentMT (PatentJP). The test set in the patent domain (Test\(_{Patent}\)) was in-house 2,000 sentences in which the word segmentation was manually annotated by the same word segmentation standards as the other labeled data. Corpus statistics are shown in Table 1.

5.1.2 Compared Methods

We compared the following word segmentation features in the word segmentation experiments.

- Baseline: only the baseline features described in 4.1
- +AV: the AV (n=2,3,4,5) and baseline features
- +BE: the BE (n=1,2,3,4,5) and baseline features
- +PD: the PD and baseline features
- +BE +PD: the BE (n=1,2,3,4,5), PD, and baseline features

To investigate the impact of the unlabeled corpus size in the semi-supervised approach, we compared two different conditions, mid-scale and large-scale; BCCWJ\(_{UL}\) and NTCIR were used in the mid-scale condition, and BCCWJ\(_{UL}\) and PatentJP\(^6\) in the large-scale condition. Here, the pseudo-dictionaries of kanji and katakana sequences were composed by kanji and katakana sequences found in the unlabeled data. We also compared the word segmenters with a publicly available word segmenter MeCab\(^7\) with a Japanese dictionary UniDic\(^8\), for reference.

\(^4\)http://www.chokkan.org/software/crfsuite/

\(^5\)We replaced kanji numbers with digits for consistency with the patent corpus.

\(^6\)The patent sentences in NTCIR is also included in PatentJP.

\(^7\)https://code.google.com/p/mecab/

\(^8\)http://sourceforge.jp/projects/unidic/
Table 2: Word segmentation F-measures (%) for the patent and original domains and OOV recalls (%) in the patent domain. $A$, $B$, and $P$ indicate significantly better results than +AV, +BE, +PD (in the same group), $M$ and $L$ indicate significantly better results than mid- and large-scale. PD$_m$ means the PD features derived from the mid-scale unlabeled corpora.

<table>
<thead>
<tr>
<th>Condition</th>
<th>Feature</th>
<th>Patent F-measure</th>
<th>OOV Recall</th>
<th>BCCWJ F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Labeled</td>
<td>Baseline</td>
<td>96.87</td>
<td>87.94</td>
<td>97.85</td>
</tr>
<tr>
<td>Unlabeled (Mid-scale)</td>
<td>+AV</td>
<td>$98.08^L$</td>
<td>91.25</td>
<td>98.27</td>
</tr>
<tr>
<td></td>
<td>+BE</td>
<td>$98.25^A$</td>
<td>91.38</td>
<td>98.38</td>
</tr>
<tr>
<td></td>
<td>+PD</td>
<td>$97.85^L$</td>
<td>91.18</td>
<td>98.08</td>
</tr>
<tr>
<td></td>
<td>+BE +PD</td>
<td>$98.32^A$</td>
<td>92.09</td>
<td>98.39</td>
</tr>
<tr>
<td>Unlabeled (Large-scale)</td>
<td>+AV</td>
<td>$97.80^P$</td>
<td>90.79</td>
<td>98.26</td>
</tr>
<tr>
<td></td>
<td>+BE</td>
<td>$98.34^A,P,M$</td>
<td>91.62</td>
<td>98.33</td>
</tr>
<tr>
<td></td>
<td>+PD</td>
<td>97.12</td>
<td>89.32</td>
<td>98.36</td>
</tr>
<tr>
<td></td>
<td>+BE +PD</td>
<td>$98.32^A,P,M$</td>
<td>92.33</td>
<td>98.37</td>
</tr>
<tr>
<td></td>
<td>+BE +PD$_m$</td>
<td>$98.36^A,P,M$</td>
<td>92.61</td>
<td>98.37</td>
</tr>
<tr>
<td>MeCab</td>
<td></td>
<td>97.73</td>
<td>86.94</td>
<td>98.35</td>
</tr>
</tbody>
</table>

5.1.3 Results

Table 2 shows word segmentation results in F-measures in the patent and general (BCCWJ) domains, and recalls of out-of-vocabulary words (OOV recall) in the patent domain focusing on domain-specific words not included in the general domain corpus. All the additional features showed better results in the patent domain than the baseline features and MeCab, which were statistically significant (p=0.05) by bootstrap resampling tests.

The AV and BE features helped to outperform MeCab in the patent domain especially in the OOV recall while the baseline performance was much worse. The BE features worked consistently with the different corpus sizes. The AV features with the large-scale data showed obviously worse results than with the mid-scale data; this indicates instability of the AV features with different corpus sizes. The PD features showed good performance especially in OOV recall, but those from the large-scale corpora did not work so well. This is possibly due to inappropriate pseudo-dictionary entries extracted around typographical errors, which sometimes occur between characters with similar type faces. Thus we additionally tested the combination of the PD features from the mid-scale data and the large-scale BE features, and that showed the best results. This indicates our domain adaptation is very effective for domain-specific words.

5.2 Evaluation of Translation

We finally conducted MT experiments to investigate the performance of our patent SMT system and the effect of each technique.

5.2.1 Setup

We used Japanese-to-English patent translation dataset used in NTCIR-9 (Goto et al., 2011) and NTCIR-10 (Goto et al., 2013) PatentMT. They shared the same training and development sets and used different test sets. Corpus statistics are shown in Table 3. English sentences were tokenized and parsed by an English syntactic parser Enju with its “GENIA” models for biomedical articles, and then lowercased. Japanese sentences were tokenized by the different tokenizers described later. Here in the training set, long sentences exceeding 64 words in either Japanese or English were filtered out. For the long sentence filtering, we used the segmentation results by KyTea because it is based on a short word unit and resulted in the largest number of segmented words. Note that the sentence set was the same for all Japanese segmenters.
Table 3: Corpus statistics in the number of words for translation experiments.

<table>
<thead>
<tr>
<th></th>
<th>Training</th>
<th>Development</th>
<th>Test9</th>
<th>Test10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tokenizer</td>
<td>(2,862,022 sents.)</td>
<td>(2,000 sents.)</td>
<td>(2,000 sents.)</td>
<td>(2,300 sents.)</td>
</tr>
<tr>
<td>Proposed</td>
<td>95,465,533</td>
<td>75,020</td>
<td>75,962</td>
<td>101,309</td>
</tr>
<tr>
<td>Baseline</td>
<td>94,914,460</td>
<td>74,627</td>
<td>75,504</td>
<td>100,589</td>
</tr>
<tr>
<td>KyTea</td>
<td>101,718,532</td>
<td>80,025</td>
<td>80,842</td>
<td>107,405</td>
</tr>
<tr>
<td>McCab</td>
<td>93,030,977</td>
<td>73,263</td>
<td>74,066</td>
<td>99,163</td>
</tr>
<tr>
<td>JUMAN</td>
<td>91,052,206</td>
<td>71,707</td>
<td>72,515</td>
<td>97,205</td>
</tr>
<tr>
<td>English</td>
<td>88,192,234</td>
<td>68,854</td>
<td>69,806</td>
<td>94,906</td>
</tr>
</tbody>
</table>

The Japanese-to-HFE monotone PBMT was implemented with Moses and trained using Japanese-HFE parallel sentences. The distortion limit of the PBMT was set to zero, but a standard lexicalized reordering model (wbe-msd-bidirectional-fe) was used to constrain adjacent phrase translations. The HFE-to-English SAMT was implemented with Moses-chart and trained using the HFE sentences and the corresponding English parse trees. Its reordering parameter max-chart-span was set to 200 to allow arbitrary distance reordering for accurate Japanese-to-English translation\(^9\). The search space parameter cube-pruning-pop-limit was set to 32 for efficiency, according to Sudoh et al. (2013b). Their language models were word 6-gram models trained using a large-scale English patent corpus with more than 300 million sentences. Model weights were optimized in BLEU (Papineni et al., 2002) using Minimum Error Rate Training (MERT) (Och, 2003). We chose the best weights among ten individual runs of MERT.

The katakana transliteration was implemented as a Moses-based monotone PBMT in the character level, trained using transliteration pairs mined from the Japanese-English phrase table entries whose Japanese part consisted of katakana only. Its character-level language model was character 9-gram models trained using the large-scale English patent corpus which is used for the word-level language models described above. It was used to replace katakana words remained in the intermediate results in HFE with their transliteration results.

5.2.2 Compared Methods

We compared following segmenters for the translation experiments.

- **Baseline**: the baseline word segmenter same as the word segmentation experiments above
- **Proposed**: the patent-adapted segmenter using the labeled general-domain corpus and the large-scale unlabeled patent corpus with the BE and PD features
- **KyTea, McCab, and JUMAN**: publicly available Japanese morphological analyzers

We also compared the results by the post-ordering with those by standard SAMT and PBMT. The search space parameters of the standard SAMT were set to the same value as the HFE-to-English SAMT, to compare the performance with similar computation time\(^11\).

5.2.3 Results and Discussion

Table 4 shows the translation performance in BLEU and TER (Snover et al., 2006) with the results of statistical significance tests (p=0.05) by bootstrap resampling (Koehn, 2004), in which our overall system resulted in the best. The table also shows the results of intermediate Japanese-to-HFE translation. The advantage of our system can be attributed to three techniques included in the system: domain adaption of word segmentation, katakana unknown word transliteration, and post-ordering.

\(^9\) It exceeded the maximum sentence length in the development and test sets.

\(^10\) http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?JUMAN

\(^11\) Actually the post-ordering needs the time for the first monotone PBMT but it ran very fast and did not affect so much (Sudoh et al., 2013b).
Table 4: Results of overall Japanese-to-English translation and intermediate Japanese-to-HFE translation in BLEU and TER. + indicates the difference from the results without transliteration is statistically significant. * indicates the difference from Proposed in the same group is statistically significant.

First, the post-ordering contributed the largest and significant improvements compared with the standard SAMT and PBMT, by about 1-2 points in BLEU and 2-3 points in TER. They basically followed the results by Sudoh et al. (2013b).

Second, the proposed word segmentation showed significant improvements in most cases, by the better intermediate translation results shown at the bottom of Table 4. Although the absolute improvement was not so large, the domain adaptation worked consistently. These results suggest that the domain adaptation of word segmentation actually worked for the patent SMT. We also analyzed the advantage of the patent-adapted word segmentation by the number of unknown words in translation. Table 5 shows the numbers of unknown kanji and katakana words that were not translated in the monotone PBMT, by the five word segmenters in the experiments. These values reflect the consistency and granularity problem in word segmentation (Chang et al., 2008). If the word segmentation is consistent and have relatively small granularity (choosing shorter words), the number of the unknown words becomes small. The granularity is closely related to the problem of compound words in this work; the translation of compound words becomes easy if they are segmented to short and appropriate component words. MeCab and JUMAN are dictionary-based word segmenters that have an advantage on precise segmentation of in-vocabulary words. JUMAN used a large-scale dictionary collected from web texts covering many domain-specific words, and resulted in a smaller number of unknown words than MeCab. KyTea and this paper’s segmenter are character-based ones that have an advantage on identifying out-of-vocabulary words (as shown in Table 2). KyTea worked well on kanji words, but derived a large number of katakana unknown words. It was probably due to the difference of embedded information between ideogram (kanji) and phonogram (katakana). Katakana compound words are usually difficult to segment only by their poor character-based information. The proposed method used reliable word boundary clues derived from the large-scale corpora and achieved consistent word segmentation of katakana compound words with
Table 5: Statistics of unknown kanji and katakana words (non-translated words by monotone PBMT). The numbers in parentheses are the number of unique unknown words.

<table>
<thead>
<tr>
<th>Ja word segmenter</th>
<th>test9</th>
<th>test10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>kanji</td>
<td>katakana</td>
</tr>
<tr>
<td>Proposed</td>
<td>18 (18)</td>
<td>30 (20)</td>
</tr>
<tr>
<td>Baseline</td>
<td>54 (43)</td>
<td>87 (59)</td>
</tr>
<tr>
<td>KyTea</td>
<td>10 (10)</td>
<td>108 (79)</td>
</tr>
<tr>
<td>MeCab</td>
<td>48 (39)</td>
<td>68 (50)</td>
</tr>
<tr>
<td>JUMAN</td>
<td>2 (2)</td>
<td>48 (41)</td>
</tr>
</tbody>
</table>

Table 6: Transliteration accuracy in sample-wise correctness (ACC) in the proposed system.

<table>
<thead>
<tr>
<th></th>
<th>Test9</th>
<th>Test10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed</td>
<td>53.33 (16/30)</td>
<td>59.37 (19/32)</td>
</tr>
</tbody>
</table>

Figure 6: Examples of small granularity segmentation for out-of-vocabulary words by JUMAN.

more appropriate granularity than others, as suggested by the smallest number of katakana unknown words in Table 5. Such an advantage was not found in kanji words compared to KyTea and JUMAN. However, JUMAN tended to choose small granularity segmentations for out-of-vocabulary words as shown in the examples in Figure 6, so these results may not indicate directly the disadvantage of the proposed method.

Finally, the transliteration itself did not improve BLEU and TER significantly in our system, although some significant improvements were found in the results by the other segmenters because of their many unknown katakana words. Its effect was limited only on the unknown katakana words and their context words (related to the word n-gram language model and the post-ordering) and did not contribute well to BLEU and TER with a small number of the unknown katakana words. We analyzed the transliteration accuracy in the intermediate HFE results with the transliteration as shown in Table 6. About a half of the unknown katakana words were transliterated correctly. This improvement is practically important for the assimilation.

6 Conclusion

This paper presented our Japanese-to-English SMT system specialized for patent translation, including the effective word segmentation by our domain adaptation method, the unknown katakana word transliteration, and the efficient syntax-based post-ordering. We achieved better translation performance by the system than by other existing Japanese word segmenters and standard SMT methods.

Domain adaptation is also expected to be effective in other components such as syntactic parsing, translation models, and language models. The application of monolingual and bilingual knowledge of other domains to the patent domain especially for named entities, and the use of patent MT knowledge in MT for other domains are also practically important.
References


A Discriminative Framework of Integrating Translation Memory Features into SMT

Liangyou Li
liangyouli@computing.dcu.ie
Andy Way
away@computing.dcu.ie
Qun Liu
qliu@computing.dcu.ie

CNGL Centre for Global Intelligent Content,
School of Computing, Dublin City University, Ireland

Abstract
Combining Translation Memory (TM) with Statistical Machine Translation (SMT) together has been demonstrated to be beneficial. In this paper, we present a discriminative framework which can integrate TM into SMT by incorporating TM-related feature functions. Experiments on English–Chinese and English–French tasks show that our system using TM feature functions only from the best fuzzy match performs significantly better than the baseline phrase-based system on both tasks, and our discriminative model achieves comparable results to those of an effective generative model which uses similar features. Furthermore, with the capacity of handling a large amount of features in the discriminative framework, we propose a method to efficiently use multiple fuzzy matches which brings more feature functions and further significantly improves our system.

1 Introduction
Translation Memory (TM) has been widely used to assist human translators. It provides the most similar source sentence in the database together with the target translation as the reference to a human for post-editing. As TM stores legacy translations, it can give high quality and consistent translations for repetitive materials. However, it performs badly when there are no highly similar matches in TM.

In contrast, Statistical Machine Translation (SMT) automatically learns several models, such as the translation model (from parallel data) and language model (from the target side of the parallel corpus as well as other monolingual data), and uses them to translate a new sentence. The translation is produced by maximizing a weighted combination of these models. Given a large amount of data, SMT can generate better results for unseen sentences than TM. However, unless sentence-caching is utilised, it treats a seen sentence (such as a sentence in the training data) as unseen.

Clearly, TM and SMT complement one another on matched and unmatched segments, so both are receiving increasing attention from translators and researchers, who would like to combine TM and SMT together to obtain better translation quality with methods such as system recommendation (He et al., 2010a,b) or using fragments from TM in SMT (Biçici and Dymetman, 2008; Koehn and Senellart, 2010; Ma et al., 2011; Wang et al., 2013)

This paper is focused on integrating TM into SMT to improve translation quality. We present a discriminative framework which directly integrates TM-related feature functions into SMT. In this paper, we change features extracted from TM which are defined in a generative
model (Wang et al., 2013) to feature functions and add them into the phrase-based translation model. Experiments on English–Chinese and English–French tasks show that our method achieves comparable results with Wang et al. (2013), and is significantly better than the baseline phrase-based system. In addition, we present a method to incorporate multiple fuzzy matches into our system, which brings further significant improvement.

In the rest of this paper, we first introduce related work on TM and SMT combination (Section 2). Then Section 3 details our discriminative framework, TM features and the approach of using multiple fuzzy matches. Then, we provide experiments to examine our method (Section 4) and give a conclusion together with avenues for future work in Section 5.

2 Related Work

As shown in experiments (e.g. Koehn and Senellart (2010) and Wang et al. (2013)), TM can give better translation than SMT for highly matched segments; SMT is more reliable than TM for other segments. Because of such complementariness, combining TM and SMT together has been explored by some researchers in recent years.

He et al. (2010a) present a recommendation system which uses an SVM (Cortes and Vapnik, 1995) binary classifier to select a translation from the outputs of TM and SMT with the selected translation being more suitable to post-editing. They take TER (Snover et al., 2006) score as the measure of post-editing effort and use it to create training instances for SVM. He et al. (2010b) extend this work by re-ranking the N-best list of SMT and TM. However, these works are focused on sentence-level selection and thus the matched phrases in TM are not used so well.

For an input sentence, even though it does not have an exact match in the TM, there are some matched phrases which could provide useful hints for translation. Bicici and Dymetman (2008) present a dynamic TM approach which dynamically adds the longest matched non-continuous phrase and its translation in the TM to the phrase table. They show a significant improvement over both SMT and TM. However their baseline SMT system seems to perform badly (Koehn and Senellart, 2010), in which case their claims need to be considered with caution. Koehn and Senellart (2010) and Ma et al. (2011) use TM in a pipeline manner: first, identifying the matched part from the best match in the TM and merging their translation with the input; then, forcing SMT to translate the unmatched part of the input sentence. One drawback of these methods is that they do not distinguish whether a match is good or not at phrase-level.

Wang et al. (2013) propose a deep integration method by using TM information during decoding. For a phrase pair applied to an input sentence, this method extracts features from the best match in the TM, and uses pre-trained generative models to estimate one or more probabilities, and then adds them into the phrase-based system for scoring a translation. These pre-trained models are built using a factored language model (Bilmes and Kirchhoff, 2003) over sequences of features. Their experiments show significant improvement over TM, SMT and pipeline approaches. However, their work requires a rather complex process to obtain training instances for these pre-trained models, and needs to define the generative relation between different features.

3 Our Method

In this section, we present a generalized discriminative framework which can integrate TM into SMT at decoding time. Under this framework, we add features from Wang et al. (2013) into the phrase-based model as TM feature functions. In addition, we describe how to use multiple fuzzy matches efficiently to improve translation quality.
3.1 Discriminative Framework

Generally, in a state-of-the-art statistical translation framework like Moses (Koehn et al., 2007), the direct translation probability is given by a discriminative framework, as shown in Equation (1):

$$P(e \mid f) = \frac{\exp\{\sum_{m=1}^{M} \lambda_m h_m(e, f)\}}{\sum_{e'} \exp\{\sum_{m=1}^{M} \lambda_m h_m(e', f)\}}$$

where $h_m(e, f)$ denotes the $m$th feature function for target $e$ and source $f$, $\lambda_m$ is the weight of this feature function, and $M$ is the number of feature functions considered.

This framework works well on pre-defined features, such as the translation model features and language model features, which are based on target $e$ and source $f$. However, as is well-known, once these features have been induced, the training data (which can be a data) is disregarded in decoding. In our work, we want to maintain the possibility of consulting such TM source-target segments (with exact and fuzzy matches) at runtime.

In this paper, we argue that given a foreign sentence $f$, the probability of its translation $e$ is conditioned on foreign sentence $f$ and TM $D$: $P(e \mid f, D)$. When $D$ is unavailable, it falls back to $P(e \mid f)$. Thus the discriminative model in Equation (1) could be generalized to Equation (2):

$$P(e \mid f, D) = \frac{\exp\{\sum_{m=1}^{M} \lambda_m h_m(e, f, D)\}}{\sum_{e'} \exp\{\sum_{m=1}^{M} \lambda_m h_m(e', f, D)\}}$$

From this, we obtain the rule in Equation (3):

$$e = \arg\max_{e'} \{P(e' \mid f, D)\}$$

$$\simeq \arg\max_{e'} \{P(e' \mid f, D_f)\}$$

$$\simeq \arg\max_{e'} \{\sum_{m=1}^{M} \lambda_m h_m(e', f, D_f)\}$$

When $h_m(e', f, D_f) = \log p(e')$, this is known as the language model feature; and when $h_m(e', f, D_f) = \log p(f \mid e)$, this is known as the translation model feature. From Equation (3) we can see that, for an input sentence $f$, instead of using the whole TM $D$, we only use one or more of the matches $D_f$ in $D$.

In this paper, we integrate TM into a phrase-based SMT model. In decoding, the foreign input sentence $f$ is segmented into a sequence of $I$ phrases $f_i^f$, and each foreign phrase $f_i^f$ is translated into a target phrase $f_i^e$. Thus, a TM-related feature function can be seen as the sum of $I$ feature functions which are based on phrase pairs, as in Equation (4):

$$h(e, f, D_f) = h(\tau_i^e, \tau_i^f, D_{\tau_i})$$

$$\simeq \sum_{i=1}^{I} h(\tau_i^e, \tau_i^f, D_{\tau_i})$$

where $h(\tau_i^e, \tau_i^f, D_{\tau_i})$ gives a value measured on the phrase pair $(\tau_i^e, \tau_i^f)$ and TM matches $D_{\tau_i}$.

3.2 Fuzzy Matching

In this paper, TM-related features are extracted from the matches in the TM. For retrieving matches, we use a word-based string edit distance (Koehn and Senellart, 2010) to measure the
similarity between the input sentence and a TM instance, as in Equation (5):

$$FMS = 1 - \frac{\text{edi.distance}(\text{input}, \text{tm.source})}{\max(|\text{input}|, |\text{tm.source}|)}$$  (5)

During the calculation of the fuzzy match score, we also obtain a sequence of operations, including insertion, match, substitution and deletion, to convert the input sentence into a TM instance. Such operations are useful for finding the TM correspondence of a source phrase.

### 3.3 Translation Memory Features

In this paper, we change features from Wang et al. (2013) to TM feature functions, and add them into our phrase-based system. The value of each feature function on a sentence pair is the sum of values from features extracted on phrase pairs, as in Equation (4).

Given an input sentence $f$ and its best match $(tmf, tme)$ in the TM, for each phrase pair $(pf, pe)$ applied to $f$, we first find its corresponding TM source phrase $ptmf$ in $tmf$ based on operations for calculating edit-distance. Then with the help of word alignment between $tmf$ and $tme$, we identify one or more TM target phrases $ptme_i$ in $tme$ by extending them with unaligned words. Then we extract the following features for the phrase pair $(pf, pe)$. Figure 1 shows an example:

- Feature set $Z_{i,j}$ indicates which match in the TM is used for source phrase $pf$. We split fuzzy match score into 11 bins: $[0, 0.1), [0.1, 0.2), [0.2, 0.3), [0.3, 0.4), [0.4, 0.5), [0.5, 0.6), [0.6, 0.7), [0.7, 0.8), [0.8, 0.9), [0.9, 1.0], [1.0]$, which correspond to 11 features: $Z_{0} \cdots Z_{10}$. For example, in Figure 1, $FMS(f, tmf) = 0.818$, so it goes into bin $Z_{8}$, and we add a value 1 to the feature $Z_{8}$.

![Figure 1: An example of extracting TM features. Target Chinese words are replaced by their corresponding Latin characters. The italic words in parentheses are the notions used in Section 3.3.](image-url)
• Feature set $SCM_s$ represents the matching status between $pf$ and $ptmf$. If $ptmf$ is unavailable, we add the value 1 to the feature $SCM_{non}$; if $FMS(pf,ptmf) < 0.5$, we add the value 1 to the feature $SCM_{low}$; if $FMS(pf,ptmf) > 0.5$, we add the value 1 to the feature $SCM_{high}$; and if $FMS(pf,ptmf) = 0.5$, we add the value 1 to the feature $SCM_{medium}$.

• Feature set $SPL_i$ measures the length of $pf$. For example, if $\text{length}(pf) = 4$, we add the value 1 to the feature $SPL_4$. In this paper, we set maximum phrase length 7 in our system, so there are 7 features in this set.

• Feature set $SEP_b$ is the indicator of whether $pf$ is the punctuation at the end of sentence $f$ or not. If yes, we add the value 1 to the feature $SEP_Y$; otherwise, we add the value 1 to the feature $SEP_N$.

• Feature set $TCM_s$ is the matching status between $pe$ and $ptme_i$. If $ptme_i$ is unavailable, we add the value 1 to the feature $TCM_{non}$; otherwise, for each $ptme_i \in ptme^l_i$: if $FMS(pe,ptme_i) < 0.5$, we add the value 1 to the feature $TCM_{low}$; if $FMS(pe,ptme_i) > 0.5$, we add the value 1 to the feature $TCM_{high}$; and if $FMS(pe,ptme_i) = 0.5$, we add the value 1 to the feature $TCM_{medium}$.

• Feature set $NLN_{x,y}$ models the matching status of context between $pf$ and $ptmf$, where $x$ denotes the number of matched source neighbours (left and right words) and $y$ denotes how many of those neighbours are aligned to target words. If $ptmf$ is unavailable, we just add the value 1 to the feature $NLN_{non}$. Taking Figure 1 as an example, the left words of source phrase “that you want to delete” and TM source phrase “you want to edit” are the same and their right words are also the same, so $x = 2$. As both left and right words are aligned to target words, $y = 2$, so we add the value 1 to the feature $NLN_{2,2}$. In total, there are 6 different $<x,y>$ tuples.

• Feature set $CSS_s$ describes the status of $ptme^l_i$. If $ptme^l_i$ is unavailable, we add the value 1 to the feature $CSS_{non}$; if $J = 1$, we add the value 1 to the feature $CSS_{single}$; if $J > 1$ and all phrases in $ptme^l_i$ are generated by extending only the left side, we add the value 1 to the feature $CSS_{left}$; if $J > 1$ and all phrases in $ptme^r_i$ are generated by extending only the right side, we add the value 1 to the feature $CSS_{right}$; if $J > 1$ and phrases in $ptme^l_i$ are generated by extending both sides, we add the value 1 to the feature $CSS_{both}$.

• Feature set $LTC_s$ is the indicator of whether a phrase $ptme_i$ in $ptme^l_i$ is the longest or not. If $ptme^l_i$ is unavailable, we add the value 1 to the feature $LTC_{non}$; if $ptme_i$ is the phrase without being extended by unaligned words, we add the value 1 to the feature $LTC_{original}$; if $ptme_i$ is only extended on its left side and has the longest left side, we add the value 1 to the feature $LTC_{left}$; if $ptme_i$ is only extended on its right side and has the longest right side, we add the value 1 to the feature $LTC_{right}$; if $ptme_i$ is extended on both sides and is the longest on both sides, we add the value 1 to the feature $LTC_{both}$; if $ptme_i$ is the one extended but not the longest one, we add the value 1 to the feature $LTC_{medium}$.

• Feature set $CPM_s$ models the reordering information. If $ptmf$ is unavailable, we add the value 1 to the feature $CPM_{non}$. Otherwise, let $(pf, pe)$ denote the last phrase pair applied to sentence $f$ and assume the translation is generated from left-to-right. Furthermore, let $(ptmf, ptme_i)$ denote the matched TM phrase pair for $(pf, pe)$. When both $ptme_i$ and $ptmf$ are available:
if \( ptme_j \) is on the right of and adjacent to \( ptme_i \),

- if the left boundary words of \( pe \) and \( ptme_j \) are the same and the right boundary words of \( pe \) and \( ptme_i \) are also the same, we add the value 1 to the feature \( CPM_{AdjacentSame} \).
- otherwise, we add the value 1 to the feature \( CPM_{AdjacentSubstitute} \).

- if \( ptme_j \) is on the right of but not adjacent to \( ptme_i \), we add the value 1 to the feature \( CPM_{LinkedInterlived} \).

- if \( ptme_j \) is not on the right of \( ptme_i \),

  - if \( ptme_j \) and \( ptme_i \) overlap, we add the value 1 to the feature \( CPM_{LinkedCross} \).
  - otherwise, we add the value 1 to the feature \( CPM_{LinkedReversed} \).

When \( ptme_i \) is unavailable and \( ptme_j \) is available, we need to find the last available TM phrase pair used in the input, let it be \( (ptmf_i, ptme_{N1}) \), for phrase \( ptme_{N1} \) in \( ptme \):

- if \( ptme_j \) is on the right of \( ptme_{N1} \), we add the value 1 to the feature \( CPM_{SkipForward} \).
- if \( ptme_j \) is not on the right of \( ptme_{N1} \),

  - if \( ptme_j \) and \( ptme_{N1} \) overlap, we add the value 1 to the feature \( CPM_{SkipCross} \).
  - otherwise, we add the value 1 to the feature \( CPM_{SkipReversed} \).

In Figure 1, the previous phrase pair is <“select”,“Xuanze”>, and its corresponding phrase pair in the TM is indicated by a rectangle. Taking TM target phrase 1 as an example, it is to the right of and adjacent to the previous TM target phrase “Xuanze” and has the same left boundary word with the target phrase “Yao Shanchu”. Furthermore, the right boundary words of the previous target phrase “Xuanze” and previous TM target phrase “Xuanze” are the same, so we use the feature \( CPM_{AdjacentSame} \).

### 3.4 Multiple Fuzzy Matches

In Section 3.3, only the best fuzzy match is used to extract features. Although we were able to find a correspondence in the TM for each source phrase, sometimes this correspondence is actually not the same as the source phrase, as shown in Figure 1. Thus we propose a method to use multiple fuzzy matches to cover as many source phrases as possible.

In this paper, besides the best match, for each source phrase we also find a TM instance which contains this phrase and has the highest fuzzy match score with the input sentence. We call such a TM instance span-match. Figure 2 shows an example of finding multiple matches.

Different to the best match which is estimated over the whole sentence and thus does not bias to any particular source phrase, span-match provides us with information about how a specific source phrase is used and thus may be helpful in selecting the proper target candidate. In addition, note that for a source sentence, the number of span-matches used is not fixed and has no limitation, so our method does not need to be optimized on such parameters.

When multiple fuzzy matches are considered, for each phrase pair applied to the input sentence during decoding, we extract features for it not only from the best match but also from the span-match of the source phrase. Features from span-match are the same as those defined in Section 3.3, except \( SPL_j \) and \( SEP_s \) are excluded as they are the same as features from the best match. In addition, \( CPM_s \) are not used on span-match as the current source phrase may be not using the same span-match as the last phrase. We distinguish features from the best match and the span-match by adding additional information, such as feature...
Source: click to select the policy that you want to delete.

TM Source 1: click to select the policy you want to edit.

TM Source 2: click to select the existing policy that you want have replaced.

TM Source 3: in the policies pane, click the specific policy that you want to delete.

Figure 2: An example of finding multiple matches.

<table>
<thead>
<tr>
<th></th>
<th>EN-ZH sentences</th>
<th>words (EN)</th>
<th>words (ZH)</th>
</tr>
</thead>
<tbody>
<tr>
<td>train</td>
<td>86,602</td>
<td>1,148,126</td>
<td>1,171,313</td>
</tr>
<tr>
<td>dev</td>
<td>762</td>
<td>10,599</td>
<td>10,791</td>
</tr>
<tr>
<td>test</td>
<td>943</td>
<td>16,366</td>
<td>16,375</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>EN-FR sentences</th>
<th>words (EN)</th>
<th>words (FR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>train</td>
<td>765,922</td>
<td>20,604,865</td>
<td>22,401,839</td>
</tr>
<tr>
<td>dev</td>
<td>1,902</td>
<td>67,403</td>
<td>73,743</td>
</tr>
<tr>
<td>test</td>
<td>1,919</td>
<td>71,228</td>
<td>78,177</td>
</tr>
</tbody>
</table>

Table 1: Summary of English–Chinese (EN-ZH) and English–French (EN-FR) corpus

BFM, SCM, high, which is from the best match, and SPAN, SCM, high, which is from the span-match. In addition, we also define two more features:

- Feature NO_SPAN_MATCH means we cannot find a span-match for current source phrase.
- Feature IS_SPAN_BEST means this span match is equal (the same fuzzy match score) to the best match.

4 Experiment

4.1 Data

Our English-Chinese data set is a translation memory from Symantec, as shown in Table 1. Our English–French data is from the publicly available JRC-Acquis corpus. Sentences are tokenized with scripts in Moses. We randomly select 3000 sentence pairs as dev data and 3000 as test data. We filter sentence pairs longer than 80 words in the training data and 100 words in the dev and test data. We also keep the length ratio less than or equal to 3 in all data sets. Table 1 also shows a summary of English–French corpus.

4.2 Baseline

On both language-pairs, we take the phrase-based model in Moses with default settings as our baseline. Word alignment is performed by GIZA++ (Och and Ney, 2003), with heuristic function grow-diag-final-and (Koehn et al., 2003). We use SRILM (Stolcke, 2002) to train a 5-gram language model on the target side of the training data with modified Kneser-Ney

discounting (Chen and Goodman, 1996). Minimum Error Rate Training (MERT) (Och, 2003) is used to tune weights. However, when TM features are incorporated, the number of features grows to more than 50 (Table 2 show the features used in our system when only best match is considered). As MERT is known to be weak when the number of features grows (Durrani et al., 2013), we use MIRA (Cherry and Foster, 2012) instead to tune weights in this case. We set the maximum iteration of MIRA to be 25. Case-insensitive BLEU (Papineni et al., 2002) is used to evaluate the translation results. Bootstrap resampling (Koehn, 2004) is also performed to compute statistical significance with 1000 iterations.

We implement Wang et al. (2013)'s method in Moses for comparison. This method needs first to train three models with the factored language model toolkit (Kirchhoff et al., 2007) over the feature sequence of phrase pairs. To obtain such phrase pairs for training, we do cross-folder translation on two language pairs. For the English–Chinese task, we split the training data into 50 parts and build 50 systems with the above settings by taking each part as test data and the rest as training data. Systems are tuned via the devset for the task. For the English–French task, we do 10-cross folder training. After training the systems, forced decoding (Schwartz, 2008) is used to generate the corresponding phrase segmentation on the test data. Then features are extracted on those phrase correspondences.

We also implement our method in Moses. In this paper, training data is taken as the TM data, so phrase rules from the TM are already included during translation. After the SMT models are trained, word alignment of the TM is also produced as a by-product.

### 4.3 Experiment Results

Table 3 shows our experiment results on two language pairs. We found that our system with TM features achieves comparable results (+0.24/+0.31 on the dev set and +0.17/-0.01 on the test set) with Wang et al. (2013) and both systems are significantly better than the baseline. After

---

<table>
<thead>
<tr>
<th>Feature Set</th>
<th>Feature name</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Z_i)</td>
<td>(Z_0, Z_1, Z_2, Z_3, Z_4, Z_5, Z_6, Z_7, Z_8, Z_9, Z_{10})</td>
</tr>
<tr>
<td>SCM(_s)</td>
<td>SCM(<em>{non}), SCM(</em>{high}), SCM(<em>{low}), SCM(</em>{medium})</td>
</tr>
<tr>
<td>SPL(_i)</td>
<td>SPL(_1), SPL(_2), SPL(_3), SPL(_4), SPL(_5), SPL(_6), SPL(_7)</td>
</tr>
<tr>
<td>SEP(_e)</td>
<td>SEP(_Y), SEP(_N)</td>
</tr>
<tr>
<td>TCM(_s)</td>
<td>TCM(<em>{non}), TCM(</em>{high}), TCM(<em>{low}), TCM(</em>{medium})</td>
</tr>
<tr>
<td>NLN(_{x,y})</td>
<td>NLN(<em>{2,2}), NLN(</em>{2,1}), NLN(<em>{2,0}), NLN(</em>{1,1}), NLN(<em>{1,0}), NLN(</em>{0,0})</td>
</tr>
<tr>
<td>CSS(_s)</td>
<td>CSS(<em>{non}), CSS(</em>{single}), CSS(<em>{left}), CSS(</em>{right}), CSS(_{both})</td>
</tr>
<tr>
<td>LTC(_s)</td>
<td>LTC(<em>{non}), LTC(</em>{original}), LTC(<em>{left}), LTC(</em>{right}), LTC(<em>{both}), LTC(</em>{medium})</td>
</tr>
<tr>
<td>CPM(_s)</td>
<td>CPM(<em>{AdjacentSame}), CPM(</em>{AdjacentSubstitute}), CPM(<em>{LinkedInterlived}), CPM(</em>{LinkedCross}), CPM(<em>{LinkedReversed}), CPM(</em>{SkipForward}), CPM(_{SkipReversed})</td>
</tr>
</tbody>
</table>

Table 2: The list of TM features extracted on the best match in our system.
Table 3: BLEU [%] on English–Chinese (EN-ZH) and English–French (EN-FR) data. Bold figures mean that the result is significantly better than the baseline phrase-based model at \( p \leq 0.01 \) level. * indicates that multiple fuzzy matches significantly improves the system with TM features at \( p \leq 0.01 \) level.

<table>
<thead>
<tr>
<th>Systems</th>
<th>EN-ZH</th>
<th>EN-FR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>dev</td>
<td>test</td>
</tr>
<tr>
<td>Phrase-based SMT</td>
<td>52.88</td>
<td>44.63</td>
</tr>
<tr>
<td>+Wang’s model</td>
<td>54.47</td>
<td>45.72</td>
</tr>
<tr>
<td>+TM feature</td>
<td>54.71</td>
<td>45.89</td>
</tr>
<tr>
<td>+multiple fuzzy matches</td>
<td>55.48*</td>
<td>46.75*</td>
</tr>
</tbody>
</table>

Table 4: Composition of test subsets based on fuzzy match scores on English–Chinese and English–French data.

(a) English–Chinese

<table>
<thead>
<tr>
<th>Ranges</th>
<th>Sentence</th>
<th>Words(EN)</th>
<th>Words/Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0.8, 1.0)</td>
<td>198</td>
<td>3,239</td>
<td>16.4</td>
</tr>
<tr>
<td>[0.6, 0.8)</td>
<td>195</td>
<td>2,876</td>
<td>14.7</td>
</tr>
<tr>
<td>[0.4, 0.6)</td>
<td>318</td>
<td>5,358</td>
<td>16.8</td>
</tr>
<tr>
<td>(0.0, 0.4)</td>
<td>223</td>
<td>4,784</td>
<td>21.5</td>
</tr>
</tbody>
</table>

(b) English–French

<table>
<thead>
<tr>
<th>Ranges</th>
<th>Sentence</th>
<th>Words(EN)</th>
<th>Words/Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0.9, 1.0)</td>
<td>313</td>
<td>10,166</td>
<td>32.5</td>
</tr>
<tr>
<td>[0.8, 0.9)</td>
<td>258</td>
<td>7,297</td>
<td>28.3</td>
</tr>
<tr>
<td>[0.7, 0.8)</td>
<td>216</td>
<td>6,128</td>
<td>28.4</td>
</tr>
<tr>
<td>[0.6, 0.7)</td>
<td>156</td>
<td>5,195</td>
<td>33.3</td>
</tr>
<tr>
<td>[0.5, 0.6)</td>
<td>171</td>
<td>5,832</td>
<td>34.1</td>
</tr>
<tr>
<td>[0.4, 0.5)</td>
<td>168</td>
<td>5,754</td>
<td>34.3</td>
</tr>
<tr>
<td>[0.3, 0.4)</td>
<td>277</td>
<td>11,157</td>
<td>40.3</td>
</tr>
<tr>
<td>(0.0, 0.3)</td>
<td>360</td>
<td>19,699</td>
<td>54.7</td>
</tr>
</tbody>
</table>

multiple fuzzy matches are incorporated, our system shows further significant improvement (+0.76/+0.62 on dev and +0.86/+0.67 on test).

In addition, we are also interested in the performance of the systems on different fuzzy match ranges. Table 4 shows statistics on subsets of test data based on fuzzy match ranges on English–Chinese and English–French data. We see that sentences with a lower fuzzy match score (0.0-0.4) are longer.

The BLEU scores [%] for different fuzzy match ranges are shown in Figure 3. It is easy to see that our system with multiple fuzzy matches achieves best performance over most ranges. Especially on the English–Chinese task, when both Wang’s model and the TM features are ineffective on the range (0.0,0.4) and [0.4,0.6), multiple fuzzy matches improve the system to give the best translation on both language pairs. However, in the highest range, Wang et al. (2013)’s method gives the best results. It seems that our system does not bias to high-scoring fuzzy match range and treat all ranges fairly.
Figure 3: BLEU [%] for different fuzzy match ranges on two language pairs. The baseline is the phrase-based SMT system. The other three systems integrate different TM information into the baseline.
5 Conclusion

In this paper, we present a discriminative framework which can integrate TM into SMT. Under this framework, we add TM feature functions, which model the relation between the source sentence and TM instances, into a phrase-based SMT. In experiments on English–Chinese and English–French tasks, our method performs significantly better than the baseline phrase-based system. Furthermore, we present a method to efficiently use multiple fuzzy matches. Experiments show that this addition significantly improves our system.

Although in this paper most features are from Wang et al. (2013), our method is much simpler yet shows comparable results to their work. In addition, our method can be more easily extended with further features and integrated into other translation models, such as hierarchical phrase-based and syntax-based models. These are avenues for future work. Furthermore, as our method is SMT-centric, in the future we would also like to extend it to get the best of both worlds (SMT and TM).

Acknowledgements

This research has received funding from the People Programme (Marie Curie Actions) of the European Unions Seventh Framework Programme FP7/2007-2013/ under REA grant agreement no. 317471. This research is also supported by the Science Foundation Ireland (Grant 12/CE/12267) as part of the Centre for Next Generation Localisation at Dublin City University. The authors of this paper also thank Kun Wang and Xiaofeng Wu for their help on our experiments and thank the reviewers for helping to improve this paper.

References


Assessing the Impact of Speech Recognition Errors on Machine Translation Quality

Nicholas Ruiz  
Marcello Federico  
Fondazione Bruno Kessler, Trento, 38122, Italy

Abstract

In spoken language translation, it is crucial that an automatic speech recognition (ASR) system produces outputs that can be adequately translated by a statistical machine translation (SMT) system. While word error rate (WER) is the standard metric of ASR quality, the assumption that each ASR error type is weighted equally is violated in a SMT system that relies on structured input. In this paper, we outline a statistical framework for analyzing the impact of specific ASR error types on translation quality in a speech translation pipeline. Our approach is based on linear mixed-effects models, which allow the analysis of ASR errors on a translation quality metric. The mixed-effects models take into account the variability of ASR systems and the difficulty of each speech utterance being translated in a specific experimental setting. We use mixed-effects models to verify that the ASR errors that compose the WER metric do not contribute equally to translation quality and that interactions exist between ASR errors that cumulatively affect a SMT system’s ability to translate an utterance. Our experiments are carried out on the English to French language pair using eight ASR systems and seven post-edited machine translation references from the IWSLT 2013 evaluation campaign. We report significant findings that demonstrate differences in the contributions of specific ASR error types toward speech translation quality and suggest further error types that may contribute to translation difficulty.

1 Introduction

Spoken language translation (SLT) systems are composed with an automatic speech recognition (ASR) system that transcribes source language utterances and a machine translation (MT) system that translates the transcripts into a target language. While there is growing interest in constructing tightly-coupled ASR and MT systems that leverage joint training and optimization (He and Deng, 2012, 2013), the dominant approach is to construct a pipeline consisting of a MT system that decodes one or more ASR hypotheses (Ney, 1999; Matusov et al., 2006; Bertoldi et al., 2007; Casacuberta et al., 2008). The individual SLT components are trained and evaluated independently against local optimization metrics that fit each model to its local task, but they do not generalize to overall SLT quality. In particular, the de-facto automatic evaluation metric for speech recognition is Word Error Rate (WER), which classifies ASR errors into three categories corresponding to Levenshtein distance alignments (i.e. insertions, substitutions, and deletions) between a hypothesis and its reference. The linguistic features of the erroneous words and their relative positions in an utterance are not taken into account.

In this paper, we investigate the impact of ASR errors on speech translation quality. In particular, does each type of ASR error contribute equally to the performance degradation of MT
outputs, or do specific classes of ASR errors more greatly inhibit the capacity of the translation model and language model to provide adequate translations? Using the results of the International Workshop on Spoken Language Translation (IWSLT)'s ASR and MT tracks in 2013 (Cettolo et al., 2013), we analyze the impact of ASR errors on the translation quality of TED talks when translating with a standard phrase-based statistical machine translation (SMT) system trained on talk transcripts. We measure the increase of translation errors due to ASR errors over the errors associated with translating well-formed speech transcripts. We further analyze the impacts of ASR errors by performing analyses with linear mixed-effects regression models (Searle, 1973): a generalization of linear regression models suited to model responses with fixed and random effects. ASR errors are categorized based on their Levenshtein alignments to the reference transcript. Experiments are performed on data covering eight ASR systems and 580 utterances in the English to French translation direction. We find that certain types of ASR errors inhibit a translation model’s ability to model and accurately translate longer phrases more than others, resulting in disjoint translation hypotheses that are difficult to score by the language model.

In Section 2, we describe related work on ASR and MT error analysis. In Section 3, we describe our experimental setup and outline the research questions used to test for differences between the effects of specific ASR error types on translation errors. In Section 4, we measure the correlation between ASR and MT errors; in Section 5 we test the assumption that translation quality is dependent on the word error rate of ASR hypotheses in the SLT pipeline. In Section 6, we analyze the effects of insertion, deletion, and substitution ASR error types on translation quality and test if each error type equally contributes to the increase in translation errors. We confirm our results by testing for interactions between error types, as well as linguistic properties of the ASR errors that may explain an increase in translation errors. Finally, Section 7 provides concluding remarks and suggestions about the utility of our findings.

2 Previous Work

Error analysis has been successfully used in the ASR and MT communities to improve the quality of each task in isolation. One of the pioneering works of error analysis in MT and ASR is that of Vilar et al. (2006), who categorize MT errors into general categories covering missing words, word order, incorrect words, unknown words, and punctuation errors. Each error type is broken down into specific subtypes covering lexical, syntactic, and semantic properties. Certain error types emerge as frequent issues in translation quality, depending on the language pair. Their analysis on the impact of ASR errors on MT shows that over 50% of the MT errors are associated with substitution errors, many of which are morphological errors that otherwise capture the meaning of the source sentence. While they show the distribution of error types in machine translation outputs, they do not elaborate on the impact of each ASR error type on MT metrics.

He et al. (2011) show that the WER score of an ASR output poorly correlates with its BLEU score of the final SLT output. They demonstrate that optimizing ASR feature weights such as the language model and word insertion penalty to minimize WER can lead to suboptimal translations and instead suggest that discriminative training approaches that optimize WER should be replaced with joint ASR-MT log-linear models that directly optimize ASR and MT features on BLEU. They provide examples of ASR errors related to normalization and speaker disfluencies to show that minimizing WER does not necessarily yield optimal translation scores. While the results suggest that WER minimization in ASR is suboptimal in spoken language translation, they do not identify the contribution of the types of ASR errors on translation errors.

In the ASR community, Goldwater et al. (2010) use a statistical analysis framework based
on mixed-effects models to analyze the effects of lexical, prosodic, contextual, and disfluency features of individual words on WER in two state-of-the-art ASR systems and show that their effects on ASR quality are dependent on position and context. For example, they show that while disfluencies account for most ASR errors, only fragments, non-final repetitions, and words preceding fragments have a significant impact. We use a similar experimental setup to measure the influence of different ASR error types, expressed as continuous fixed effects, on the increase in machine translation errors over the translations of perfectly recognized utterances.

A number of works have been proposed to mitigate the contextual effects of ASR errors on MT quality by adapting the SMT phrase table. Ananthakrishnan et al. (2013) use attention-shift decoding for ASR (Kumaran et al., 2007) to identify reliable islands and unreliable gaps in an ASR hypothesis. The SMT decoder penalizes phrase translation pairs whose source phrases span across island-gap boundaries. Likewise, the language model penalizes target language n-grams that cover the island-gap boundary in the source phrase. Tsvetkov et al. (2014) augment phrase-based MT translation models with synthetic phrases by identifying word contexts in ASR outputs that contain acoustically confusable phoneme sequences. The source phrase for each bilingual phrase pair is checked for alternative word sequences that have acoustically similar phoneme sequences. Source-side matches are added to the phrase table with the same target language phrase and new phrase table features are added to measure their fluency.

3 Research methodology

Our goal is to analyze the impact of ASR errors on machine translation quality. Using WER as a metric for ASR quality, how do errors in recognizing speech utterances affect the accuracy of a machine translation system that assumes that each source sentence is clean and well-formed?

We perform our experiments on an intersection of the ASR and MT results of the IWSLT 2013 evaluation campaign (Cettolo et al., 2013), which focused on the translation of TED talks. We collect each submitter’s English ASR hypotheses on the tst2012 test set and take the subset of the ASR hypotheses that correspond to the reference set for the English-French MT track. A subset of the MT outputs of each system in the MT track was manually post-edited by professionals and served as multiple human references for automatic evaluation. Using these post-edited translations, we construct 3-way data consisting of eight English ASR hypotheses for 580 utterances, a single unpunctuated reference transcript from the ASR track, and the human post-edited translations from the English-French MT track.

We will use Translation Edit Rate (Snover et al., 2006) as a sentence-level MT quality metric, as it models the original post-editing scenario of the evaluation campaign by estimating the amount of effort required to correct machine translation output. In order to analyze the impact of ASR errors on MT quality, we construct experiments to address the following questions:

- Does ASR’s WER correlate with SMT’s automatic quality metrics (e.g. TER)?
- Do higher WER scores cause a degradation in MT quality with respect to translations on perfectly recognized utterances (\(\Delta\)TER)?
- Which types of ASR errors have the strongest impact on translation quality?

In Section 3.1, we discuss the preprocessing steps for each ASR hypothesis and in Section 3.2, we discuss how machine translation outputs are generated for each ASR hypothesis.

3.1 ASR data processing

IWSLT’s ASR submissions are in lowercase, lack punctuation, and do not have embedded segmentation. We use the segmentation file provided in the SLT track to induce segmentation. After segmentation, we use the documentation provided in the IWSLT evaluation campaign to
Prior to evaluating hypotheses from ASR systems, the DARPA Hub-4 evaluation plan (Pallett et al., 1998) and subsequent ASR evaluations such as NIST’s Rich Transcription tasks (Garofolo et al., 2002) used an evolving normalization script to prevent penalization for minor orthographic variations such as multiple spellings (e.g. British vs. American English), compound words (e.g. “storyline” vs. “story line”), and contractions (e.g. “it’s” vs. “it is”). Assuming that a phrase-based SMT system in the SLT pipeline is trained on ASR reference transcripts, orthographic variances in ASR outputs can result in out-of-vocabulary words or under-represented source language n-grams in the translation model, further degrading machine translation quality. Although both the ASR hypotheses and the reference transcripts were normalized in prior evaluations, our experiments require the ASR reference to remain unmodified in order to properly evaluate the translation of ASR outputs against translation of the original TED transcripts. Instead, we wrote a supervised word compounding script that splits or compounds words, depending on the word form in the reference transcript. Afterward we apply a bare-bones version of the normalization file provided by IWSLT which only maps British English to American English, since we observed anomalies including inconsistent mappings in the filters used for previous evaluations.

We observed a $\pm 0.3\%$ absolute difference between our WER measures after normalization and the scores reported in the official evaluation task (Cettolo et al., 2013). The rankings of each system remained consistent. In Table 1 we report the performance of each ASR system, before and after orthographic normalization. Note that 5% of the errors for each system are attributed to normalization issues of compounding or word form (e.g. British English instead of American English). The majority of the errors are related to word compounding. The left-hand side of Fig. 1 shows a system-by-system comparison of ASR error distributions. Only a couple of ASR systems have significantly different error distributions from one another.

Figure 1: Boxplots describing the distribution of ASR errors (WER) and their impact on translation errors (TER) by ASR system and utterance.
Table 1: ASR outputs used as English-French MT evaluation input data on the human evaluation task of IWSLT 2013. ASR outputs are evaluated with no additional normalization, oracle word compounding (COMP), or compounding with word form normalization (NORM). Translated ASR outputs are tokenized and evaluated against the reference translation (Auto) and a combination of the human post-edited sentences from the MT task (Post-edit).

<table>
<thead>
<tr>
<th>ASR System</th>
<th>Norm</th>
<th>ASR WER % ↓</th>
<th>MT TER % ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>All</td>
<td>S</td>
<td>D</td>
</tr>
<tr>
<td>fbk</td>
<td>none</td>
<td>21.4</td>
<td>13.3</td>
</tr>
<tr>
<td></td>
<td>+COMP</td>
<td>16.8</td>
<td>10.8</td>
</tr>
<tr>
<td></td>
<td>+NORM</td>
<td>16.5</td>
<td>10.5</td>
</tr>
<tr>
<td>kit</td>
<td>none</td>
<td>15.3</td>
<td>9.2</td>
</tr>
<tr>
<td></td>
<td>COMP</td>
<td>10.4</td>
<td>6.6</td>
</tr>
<tr>
<td></td>
<td>+NORM</td>
<td>10.1</td>
<td>6.3</td>
</tr>
<tr>
<td>mitll</td>
<td>none</td>
<td>16.4</td>
<td>9.6</td>
</tr>
<tr>
<td></td>
<td>COMP</td>
<td>11.6</td>
<td>7.0</td>
</tr>
<tr>
<td></td>
<td>+NORM</td>
<td>11.4</td>
<td>6.8</td>
</tr>
<tr>
<td>naist</td>
<td>none</td>
<td>15.7</td>
<td>9.1</td>
</tr>
<tr>
<td></td>
<td>COMP</td>
<td>10.9</td>
<td>6.5</td>
</tr>
<tr>
<td></td>
<td>+NORM</td>
<td>10.6</td>
<td>6.3</td>
</tr>
<tr>
<td>nict</td>
<td>none</td>
<td>14.5</td>
<td>8.7</td>
</tr>
<tr>
<td></td>
<td>COMP</td>
<td>9.5</td>
<td>6.0</td>
</tr>
<tr>
<td></td>
<td>+NORM</td>
<td>9.2</td>
<td>5.8</td>
</tr>
<tr>
<td>prke</td>
<td>none</td>
<td>21.3</td>
<td>13.2</td>
</tr>
<tr>
<td></td>
<td>COMP</td>
<td>16.9</td>
<td>10.8</td>
</tr>
<tr>
<td></td>
<td>+NORM</td>
<td>16.6</td>
<td>10.6</td>
</tr>
<tr>
<td>rwth</td>
<td>none</td>
<td>16.5</td>
<td>10.1</td>
</tr>
<tr>
<td></td>
<td>COMP</td>
<td>11.9</td>
<td>7.7</td>
</tr>
<tr>
<td></td>
<td>+NORM</td>
<td>11.7</td>
<td>7.5</td>
</tr>
<tr>
<td>uedin</td>
<td>none</td>
<td>17.2</td>
<td>10.2</td>
</tr>
<tr>
<td></td>
<td>COMP</td>
<td>12.6</td>
<td>7.7</td>
</tr>
<tr>
<td></td>
<td>+NORM</td>
<td>12.3</td>
<td>7.4</td>
</tr>
<tr>
<td>gold</td>
<td>none</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

3.2 MT data processing

Since we are evaluating the impact of ASR errors on translation quality, we use a fixed SMT system trained on TED talk transcripts, which correspond to the reference transcripts in the ASR track, with the addition of punctuation. We use FBK’s primary phrase-based SMT system used in the English-French MT track (Bertoldi et al., 2013). The normalized ASR hypotheses are post-processed by inserting punctuation and applying recasing. We insert the punctuation as closely as possible to the position dictated in the reference in order to control the impact of punctuation on translation output. This is done by computing the Levenshtein alignments between the unpunctuated TED transcripts and each ASR hypothesis, using SCLITE\(^1\). We train and apply a recaser model using the standard Moses tools (Koehn et al., 2007) with IWSLT 2013’s TED training data to all of the newly-punctuated ASR outputs.

After introducing punctuation and recasing the ASR output, we translate each ASR output and evaluate the results using TER over the seven human post-edited translations. Translation

\(^1\)http://www1.icsi.berkeley.edu/Speech/docs/sctk-1.2/sclite.htm
results are contrasted with FBK’s primary MT submission on the right-hand side of Table 1. We observe over a 4% absolute increase in TER for each of the translations of ASR hypotheses against the reference translation and likewise over a 6% absolute increase against the post-edited references. Most of the ASR transcripts’ translations yield a TER score around 30% against the post-edited references. Turchi et al. (2013) empirically determine that translations with a TER score above 40% against a set of post-edited references require the translator to re-translate the source sentence from scratch, while lower scores imply that it is productive for the translator to post-edit the MT output. Likewise, our reported TER scores suggest that the translations of the ASR hypotheses are of good enough quality to be used in a post-editing scenario. The right-hand side of Fig. 1 shows a system-by-system comparison of SLT error distributions, measured in TER. In particular, we observe less variance among ASR systems as their hypotheses are translated by the SMT system.

4 Do ASR errors correlate with SMT errors?

Using the WER metric, how do ASR errors correlate with SMT errors? We split this question into two related questions. First, is there any relation between an ASR system’s difficulty to recognize a speech utterance and the difficulty of translating the utterance, assuming it was recognized perfectly? The answer may seem obvious, since an ASR model could be trained poorly and generate hypotheses that have no bearing with their references. However, as described in the previous section, each of the ASR systems used in the IWSLT evaluation are capable of producing translations that can be efficiently post-edited by a professional translator. Second, do ASR errors correspond directly with translation quality? In other words, does the increase or decrease in WER correlate with the number of translation errors in the speech translation pipeline? We address these questions by analyzing the correlation between the independent variable (WER) and the dependent variables (TER on translations of ASR references and ASR hypotheses, respectively) in Section 4.1, followed by constructing linear regression models to test for statistical significance in Section 4.2.

4.1 Correlation

We first measure the correlation between the WER scores of each ASR system and the TER acquired by translating each corresponding ASR reference. The Pearson correlation coefficient, $r$, measures the linear dependence between two variables. For our experiments, we control the effects of sentence length by binning the ASR hypotheses from each system into buckets corresponding to the quartiles of the reference length. Since much of the skewness of ASR errors shown in Fig. 1 is related to ASR reference length, we take correlation measurements on the 2nd and 3rd length quartiles, corresponding to reference lengths of 9-15 and 16-22. Using all ASR systems, we observe $r$ values of 0.039 and 0.091, on the respective reference lengths, implying no correlation. Using only the observations of NICT’s primary system (which had the lowest WER in the ASR evaluation track), we observe $r$ values of -0.031 and 0.049, respectively.

We repeat the experiment, this time comparing ASR errors to their corresponding translation errors. Using all ASR systems, we observe $r$ values of 0.672 and 0.632, respectively, implying strong correlation. We observe a similar trend when considering NICT’s system alone. Again, these results are not surprising, since a machine translation system depends on the speech recognition output in order to generate a translation. It is important to note that while there is naturally a strong correlation between ASR outputs and the quality of their translations, translation quality is not solely dependent on ASR quality. The missing 30% includes phenomena related to the problem of transferring content from the source language (English) to the target language (French), which take into consideration the lexical, syntactic, and semantic properties of each language (Vilar et al., 2006; He et al., 2011; Ruiz and Federico, 2014).
4.2 Linear Regression

To verify whether the correlation results in the previous section imply dependence, we fit univariate linear regression models using a single ASR system to evaluate the contribution of WER to the corresponding translation’s TER score. We focus our attention on the observations of NICT’s primary system. The response variables are the TER scores computed against seven post-edited translation references. TER is computed either on the ASR references or on the translations of NICT’s ASR hypotheses. Again, WER is computed on the uncased, unpunctuated output of the ASR system. Translations are performed using FBK’s primary MT submission.

Our first experiment estimates the effects of WER on TER acquired by translating each corresponding ASR reference. As suggested by the low Pearson correlation scores in our previous experiment, WER is not a significant predictor of TER scores on the translation of ASR references: $\beta = 0.028, t(578) = 0.719, p = 0.473$, with a negative adjusted $r^2$ value.

Our second model treats the TER of the translated ASR hypotheses as the response variable. WER significantly predicts TER scores, $\beta = 0.696, t(578) = 18.42, p < 10^{-4}$ and explains a significant proportion of variance in TER scores ($r^2 = 0.369, F(1, 225) = 339.4, p < 10^{-4}$). However, much of the variance remains unexplained by the model. WER normalizes by the reference transcript’s utterance length; however, input length is an important factor that affects the search space in SMT decoding. Thus, WER cannot intrinsically anticipate the difficulty of translating the utterance. As evidence, we sample two utterances recognized by NICT’s ASR system, both with WER scores of 20% but having a different number of words in the reference (5 and 25, respectively). The TER scores of their translations are 46.7% and 28.4%, respectively. WER also assumes that each error contributes independently towards the error metric and thus does not measure interactions between multiple errors in an utterance. In phrase-based SMT, the position and density of ASR errors can hinder the translation model’s ability to select proper target phrases, as well as affect the reordering model’s ability to properly arrange the phrases in the target language.

5 Does a higher WER cause an increase in translation errors?

Our previous experiments in Section 4.1 measured the relationship between WER of ASR hypotheses and TER. While WER is a significant predictor of TER in our simple regression model, it fails to capture the variance in TER associated with the innate difficulty of translating the utterance. As shown in the correlation measurements in the first experiment, WER alone cannot provide reliable estimates of the number and types of errors in a perfectly recognized utterance. To control for the difficulty of translating an otherwise perfect speech recognition hypothesis, we use the difference between the TER associated with translating the perfect ASR reference and the TER associated with translating the ASR hypothesis, labeled as $\Delta$TER:

$$\Delta\text{TER} = \text{TER}_{\text{gold}} - \text{TER}_{\text{ASR}},$$

where $\text{TER}_{\text{gold}}$ is the TER score for a perfectly recognized utterance, and $\text{TER}_{\text{ASR}}$ is the TER score on the translation of the ASR hypothesis. By using $\Delta$TER, we assume that $\text{TER}_{\text{gold}}$ is the upper-bound on translation quality with the given SMT system. In other words, we assume that a SMT system cannot translate transcripts containing errors better than clean transcripts. We check this assumption in our observation data and note 64 violations out of a total of 4,640 observations covering the outputs of the eight ASR systems (1.4% of the time). As a sanity check, we had two native French speakers evaluate the translation quality of several scenarios where $\Delta$TER < −0.1. In all cases, the native speakers preferred the MT outputs of translated ASR references over the translations of ASR hypotheses. These violations are likely due to the
Table 2: Fixed and random effects for the null model, which measures the effect of WER on ∆TER for English-French SLT. The model is constructed with observations from all ASR systems in IWSLT 2013’s ASR Track on the left-hand side and only NICT and FBK’s ASR systems on the right. Fixed effects coefficients (β) and standard errors are reported. Random intercepts account for variances by utterance (UttID) and ASR system (SysID). Statistical significance at $p < 10^{-4}$ is marked with • and $p < 10^{-2}$ is marked with ◦.

<table>
<thead>
<tr>
<th></th>
<th>All ASR</th>
<th>NICT+FBK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>β</td>
<td>Std. Error</td>
</tr>
<tr>
<td>(Intercept)</td>
<td>8.72e-03</td>
<td>3.14e-03 ◦</td>
</tr>
<tr>
<td>WER</td>
<td>6.30e-01</td>
<td>8.55e-03 ●</td>
</tr>
<tr>
<td>Random effects</td>
<td>Variance</td>
<td>Std. Dev.</td>
</tr>
<tr>
<td>UttID (Inter)</td>
<td>4.50e-03</td>
<td>0.067</td>
</tr>
<tr>
<td>SysID (Inter)</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>Residual</td>
<td>3.74e-03</td>
<td>0.061</td>
</tr>
</tbody>
</table>

We first measure the correlation between WER and ∆TER using Pearson’s $r$. Following the same approach as Section 4.1, we observe strong correlations on the observations with reference lengths in the middle 50% length quartiles: 0.780 and 0.756 using all ASR systems for utterance lengths of 9-15 and 16-22, respectively, and scores of 0.786 and 0.778 using only NICT’s ASR system.

We next verify ∆TER’s dependence on WER using linear mixed-effects models, which have been effectively used on linguistic data by Baayen et al. (2008). Mixed-effects models allow us to take into consideration random effects caused by an ASR system and the particular features of each ASR utterance. We use the R (R Core Team, 2013) implementation of linear mixed-effects models in the lme4 library (Bates et al., 2014). As fixed effects, we enter WER into the model, which we label as Model 1. We provide random intercepts for the utterance (labeled as UttID) and ASR system (labeled as SysID). The models are fit by maximum likelihood. We use repeated observations of 580 speech utterances by eight ASR systems, yielding a total of 4,640 observations. Fixed effect coefficients and random effects variance for Model 1 are reported in Table 2.² Both WER and the intercept are observed as statistically significant. The coefficients suggest that if there are no ASR errors, TER will increase by 0.87%. However, for each percentage point of WER, the TER will further increase by roughly $0.63 \times 0.01 = 0.0063$ (0.63%). We observe a $r^2$ value of 0.840 for the model, 0.154 of which is attributed to the fixed effects.

As a random effect, SysID was not significant, as it has a standard deviation near zero. This behavior is also evident in the boxplots of Fig. 1, implying that the differences between the emitted WER scores and translation TER scores for each ASR system are not significantly different from one another. In order to verify that the random intercept associated with the ASR system is indeed insignificant, we repeat the mixed-effects analysis, using two systems with significantly different WER scores; namely NICT and FBK. Statistics on the fixed and random effects are also listed in Table 2. We again observe near-zero variance for SysID and do not observe significant differences in the fixed effects coefficients, implying that the SysID random effect has no impact on the model.

²Note that the WER and TER values in Table 1 are listed as percentages, while our regression models express the values between 0 and 1.
Table 3: Fixed and random effects for Models 2 and 3, which measure the effect of ASR error types on ∆TER for English-French SLT. Model 3 includes interactions between error types. Fixed effects coefficients (β) and standard errors are reported; statistically insignificant fixed effects are omitted. Random intercepts account for variances by utterance (UttID) and ASR system (SysID). Statistical significance at $p < 10^{-4}$ is marked with •, $p < 10^{-2}$ is marked with ◦, and $p < 0.05$ is marked with ∗.

6 Which types of ASR errors have the strongest impact on translation quality?

Now that we have verified that an increase in WER significantly increases TER, are there significant differences between the effects of individual ASR error types on translation quality? We hypothesize that not all ASR errors are treated equally when ASR hypotheses are used in the speech translation pipeline. To demonstrate this, we construct new mixed-effects models which factorize the WER metric into the components used to compute its score. Recall that the WER for an utterance is computed as:

$$WER = \frac{S + D + I}{L},$$  (2)

where $S$, $D$, and $I$ are the number of substitutions, deletions, and insertions in the Levenshtein alignment between the hypothesis and the reference, and $L$ is the ASR reference length (in words). According to (2), we factorize WER into three independent variables, corresponding to the number of occurrences of each error type, normalized by the reference length. As random effects, we continue to use the utterance ID and the ASR system ID. Our null hypothesis states that all length-normalized ASR error types ($S, D, I$) contribute equally to ∆TER, which is the same as our Model 1 specification in Section 5.

6.1 Do Levenshtein error types have differing levels of importance?

In the alternative hypothesis’s mixed-effects model, we enter $S/L$, $D/L$, and $I/L$ as fixed effects and maintain the same random effects as Model 1. To simplify the notation in our model, which we label Model 2, we refer to the length-normalized error types in shorthand form (e.g. WER.S). The coefficients of the fixed effects of the fitted model are shown in Table 3.

All error type coefficients are statistically significant at the $p < 10^{-4}$ level, with $r^2 = 0.840$. We perform a likelihood ratio test between Model 2 and the null model (Model 1), using the anova() function in the lmerTest R package. We observe marginal significance at the $p < .1$ level ($\chi^2(2) = 5.558, p = 0.062$), suggesting that Model 2 may better describe the relationship between ASR errors and ∆TER; however, a deeper analysis is required.
6.2 Are there interactions between Levenshtein error types?

While the coefficients in Model 2 listed in Table 3 seem to suggest that substitutions have a greater impact on translation quality than deletions or substitutions, the low level of significance may indicate that the model is missing a predictor. The WER metric in (2) posits that the influence of Levenshtein error types on ASR quality are independent from one another. In other words, it assumes that there are no interactions between error types. Our factorization of WER in Model 2 retains this assumption. We now test for interactions between the reference length-normalized error types in Model 2. We construct a new mixed-effects model (Model 3) that contains all pairwise interactions (e.g. \( WER.S \times WER.D \)), as well as the interaction triple \( (WER.S^2 \times WER.D \times WER.I) \). The fixed and random effects of the new model are reported on the right-hand side of Table 3. The \( WER.S \times WER.D \) interaction is reported as significant \( (p < 10^{-4}) \), while the other interactions are statistically insignificant.

We perform likelihood ratio tests between Model 3 and our previous models to verify the significance of the interaction term, again using the \texttt{anova()} function in the \textit{lmerTest} R package. We observe a significant difference between Models 1 and 3 \( (\chi^2(7) = 47.109, p < 1.78 \times 10^{-8}) \), as well as between Models 2 and 3 \( (\chi^2(4) = 41.55, p < 2.07 \times 10^{-8}) \), confirming the presence of interactions between ASR error types. The impact of substitutions in Model 3 is dependent on the number of deletions that co-occur within a sentence. For example, a sentence with a WER of 10%, solely caused by substitution errors, corresponds to approximately a 7.5% increase in TER \( (\beta_0 + \beta_S \times 0.1) \). However, a sentence with a WER of 15% with 10% as substitution errors and 5% as deletion errors would expect an increase in TER by \( \beta_0 + \beta_S \times 0.1 + \beta_D \times 0.05 - \beta_{SD} \times 0.1 \times 0.05 \approx 0.105 \) (10.5%); the interaction term reduces the increase in TER by 0.34%. Thus, we can conclude that not only do Levenshtein alignment-based ASR error types have differing levels of importance, but there also exists an interaction between substitution and deletion errors that reduces their individual impact on translation quality (in terms of \( \Delta \text{TER} \)).

6.3 Are there linguistic patterns of ASR errors that impact translation quality?

In Section 6.1, we tested the hypothesis that individual Levenshtein error types have different effects on translation quality. We showed weakly significant results, suggesting that the breakdown of WER into length-normalized Levenshtein alignment types may better model the relationship between ASR errors and translation quality (in \( \Delta \text{TER} \)). As we have seen with interactions between substitutions and deletions in Section 6.2, there are contexts in which the impact of substitution errors on translation quality may vary. In particular, we believe that linguistically informed errors may better describe how ASR errors violate the structural assumptions used to train standard statistical machine translation systems.

As a preliminary experiment, we focus our attention on misrecognized function words and content words. In particular, researchers such as Goldwater et al. (2010) identify function words (also known as closed class words) as problematic for speech recognition. Oftentimes a speaker may alter the pronunciation of high frequency function words, such as prepositions and articles, by under-articulating or dropping phonemes. While a human can predict these words with high accuracy, an ASR system relies on phoneme or triphone recognition as an intermediate step toward recognizing words. Content words (also known as open class words) are generally simpler to recognize, as they often contain more syllables and cover a larger amount of speaking time within an utterance. On the other hand, open class words might not be represented in a speech lexicon, rendering them impossible to be generated by an ASR system. Aside from the issue of out-of-vocabulary words, SMT systems have the opposite problem. Vilar et al. (2006) demonstrate that missing content words contribute more toward translation errors than missing function words.
Table 4: Model 4, which measures the effect of open and closed-class ASR error types on $\Delta$TER for English-French SLT. Coefficients ($\beta$) with confidence intervals are reported for the fixed effects. Random intercepts account for variances by utterance (UttID) and ASR system (SysID). Statistical significance at $p < 10^{-4}$ is marked with • and $p < 10^{-2}$ is marked with ◦.

Since we have already observed differences between Levenshtein error types, we now look at differences between how misrecognitions of open and closed class words affect translation outputs. We use TreeTagger (Schmid, 1994) to assign part-of-speech (POS) tags on the ASR references using the Penn Treebank (Marcus et al., 1993). Using the Levenshtein alignments between each ASR hypothesis and its reference, we annotate deletion and substitution errors with their POS tags. We do not annotate insertion errors, as an insertion error indicates that no reference word is available to tag. We manually map each POS tag associated with a substitution and deletion error to its class (open or closed).

Our new model (Model 4) extends Model 2 from Section 6.1 by separating substitution and deletion errors by their word classes. To simplify our model, we do not consider interactions between the error types. Statistics on the fixed and random effects are shown in Table 4a. Our results confirm that all word class-specific ASR error types are significant at the $p < 10^{-4}$ level. Likelihood ratio tests between Models 2 and 4 indicate that the Levenshtein error types grouped by word class better measure the impact of ASR errors on translation quality ($\chi^2(2) = 15.487, p = 4.34 \times 10^{-3}$). The fixed effect coefficients’ confidence intervals in Table 4b show that substitution errors on function words have the greatest individual impact on translation errors: every one percent of these errors increases $\Delta$TER by 0.7% over the intercept, assuming all other factors are held constant. Substitution errors on content words, however, have a significantly lower impact. Conversely, deletion errors on content words have a greater impact than those on function words. All other factors held constant, a standard phrase-based machine translation system is apparently more tolerant of ASR deletion errors on function words than towards substitution errors on function words. We hypothesize that this is most commonly due to cases where a function word is recognized as another function word from a different lexical category (e.g. a preposition recognized as a determiner).

7 Conclusion

In this paper, we focused on the contribution of Levenshtein alignment errors in ASR’s word error rate (WER) metric on translation quality in terms of Translation Edit Rate (TER). Working on the English-French translation direction in the IWSLT 2013 TED Talk spoken language...
translation task, we collected a subset of ASR hypotheses from eight systems on the 2012 test set and measured their translation quality against seven human post-edited translations. Using this data, we measured the impact of ASR errors on TER against a gold standard that measures the inherent complexity of an utterance, assuming perfect speech recognition. We measured the correlation between the WER of ASR hypotheses and the TER of the associated translations, showing that while WER strongly correlates with machine translation evaluation metrics such as TER, it does not account for the inherent complexity of a source language utterance.

We additionally constructed linear mixed-effects models to show that substitution, insertion, and deletion error types in the WER metric do not contribute uniformly to translation errors when using a statistical machine translation (SMT) system trained on clean transcripts. Our results suggest similarly to Vilar et al. (2006) and He et al. (2011) that while WER is the de-facto metric for ASR quality, WER alone is not a good indicator of translation quality due to its assumption of independent error types acting in isolation. WER fails to take into account the cumulative effects of errors, which include interactions between substitution and deletion errors.

Additionally, we provided a preliminary experiment that shows that the linguistic properties of ASR errors have ramifications on SMT quality in speech translation. We annotated substitution and deletion errors by their word class (open or closed), based on the part-of-speech tags assigned to each ASR reference transcript. Our preliminary results show differences in how speech recognition errors on open class and closed class words affect the machine translation engine, particularly due to their Levenshtein alignment types. We see that substitution errors on function words are more detrimental than deleting a function word; though this behavior was not observed for content words. The results indicate that more investigation should be done on the impact of ASR errors by lexical class on translation quality. In our next steps, we plan to train and apply a part-of-speech tagger directly on the ASR output to get a better idea of which types of part-of-speech errors are less likely to be tolerated by a standard phrase-based SMT system.

Our results suggest that additional error types should be considered when measuring the impact of ASR errors on spoken language translation quality. Thus far, our experiments have focused on the role of individual Levenshtein error types and their properties, though, as we have observed with interactions between error types, the context of each ASR error affects how it will impair the translation model and language model from generating a high quality translation. We suggest, for example, to analyze features that account for the distributional density of ASR errors, which may better describe how ASR errors violate the structural assumptions used to train standard SMT systems. For example, let us assume that we have two utterances with the same reference length and WER. If the errors in one utterance are concentrated at the beginning or end of the utterance, would its TER be greater than an utterance whose errors are uniformly distributed across the entire segment?

Finally, while our experiments have focused on MT references generated by human post-editions, we propose to perform this analysis on automatic references with metrics such as NIST (Doddington, 2002), METEOR (Banerjee and Lavie, 2005), and HMEANT (Lo and Wu, 2011). Ultimately, we believe that the analysis of ASR errors on SLT can result in deriving an error metric that better correlates with MT quality in the speech translation pipeline.

Acknowledgments

This work was partially supported by the MateCAT (grant agreement 287688) and EU-BRIDGE projects (grant agreement 287658), which are funded by the EC under the 7th Framework Programme. The authors also thank Matteo Negri and Marco Turchi for their useful suggestions.
References


Using Noun Class Information to Model Selectional Preferences for Translating Prepositions in SMT

Marion Weller$^{1,2}$
Sabine Schulte im Walde$^1$
Alexander Fraser$^2$

wellermn@ims.uni-stuttgart.de
schulte@ims.uni-stuttgart.de
fraser@cis.uni-muenchen.de

$^1$IMS, Universität Stuttgart, Germany
$^2$CIS, Ludwig-Maximilians-Universität München, Germany

Abstract

Translating prepositions is a difficult and under-studied problem in SMT. We present a novel method to improve the translation of prepositions by using noun classes to model their selectional preferences. We compare three variants of noun class information: (i) classes induced from the lexical resource GermaNet or obtained from clusterings based on either (ii) window information or (iii) syntactic features. Furthermore, we experiment with PP rule generalization. While we do not significantly improve over the baseline, our results demonstrate that (i) integrating selectional preferences as rigid class annotation in the parse tree is sub-optimal, and that (ii) clusterings based on window co-occurrence are more robust than syntax-based clusters or GermaNet classes for the task of modeling selectional preferences.

1 Introduction

The translation of prepositions is difficult in SMT: some prepositions convey a meaning (to sit UNDER/ON the table) while others are merely functional (to believe IN sth.). Both kinds of prepositions pose a significant challenge to the translation system as they largely depend on target-language-specific restrictions for which there is often not enough contextual information available. Translating prepositions is difficult as SMT systems must choose the correct preposition given the intended meaning of the preposition in the input sentence as well as the target-side context in which the preposition appears.

In English-to-German translation, there are cases in which the target-language does not require a preposition (e.g. to call FOR → fordern), or in which it is necessary to produce a target-language preposition even though there is no preposition in the input sentence (e.g. to enter → gelangen IN). The choice of prepositions is typically determined by a governor, such as verbs (to believe in sth.) or nouns (e.g. interest in sth.). In addition, the preposition depends on the semantic class of nouns that are governed. For example, to learn from can lead to two different translations in German: in the case of to learn from [a person], the correct translation is lernen VON, whereas to learn from [the past] should be translated as lernen AUS.

We present a novel method that uses noun class information to model selectional preferences of prepositions. By annotating noun class information into the parse trees used to train an English-German string-to-tree SMT-system, we aim at obtaining more precise translation rules. Instead of allowing any PP in a given rule, the noun class annotation restricts that rule to PPs of a specific semantic class. While this procedure adds semantically fine-grained information,
it also leads to a loss of rule generalization. We compensate for this loss by making generic, non-annotated rules accessible for the enriched system, and by generating new PP rules that cannot be derived from the parallel data. The selectional preferences are instantiated by three variants of noun class information:

- nominal concepts induced from the lexical semantic taxonomy GermaNet,
- k-Means cluster analyses relying on standard distributional window co-occurrence,
- k-Means cluster analyses relying on syntactic features from dependency-parsed data.

Using noun classes from a lexical resource such as GermaNet allows us to access a conceptually refined form of target-language information. In contrast, by using large target-language corpora as a basis for clustering, we generalize better over contexts (in a “raw” form vs. based on syntactic dependencies) and thus take into account additional target-language information based on very large corpora in a way that goes beyond the potential of an SMT-system that only has access to an n-gram language model.

Even though none of the enriched systems significantly outperforms a baseline without noun class information, our experiments provide insights into the integration of noun classes into a syntactic SMT system regarding (i) the method of annotation and (ii) the resources used. Integrating selectional preferences as rigid annotation in the parse tree is not optimal, as there is no generally applicable optimal level of semantic information. With regard to resources, we found that cluster analyses based on simple window information are better at capturing selectional preferences, with superior performance to both (a) the clusters relying on syntactic features and (b) the classes induced from the high-quality lexical resource GermaNet.

2 Related Work

Translating prepositions is an important problem in machine translation. So far, research has mostly been reported on rule-based systems. Gustavii (2005) uses bilingual features and selectional constraints to correct translations from a rule-based Swedish-English system; she reports a gain in accuracy for prepositions. Naskar and Bandyopadhyay (2006) outline a method to handle prepositions in an English-Bengali MT system: they use WordNet in combination with a bilingual example base for idiomatic PPs, but do not report any evaluation. Agirre et al. (2009) model Basque prepositions and grammatical case using syntactic-semantic features such as subcategorization triples for a rule-based system; they also report a gain in translation accuracy for prepositions. The approach of Shilon et al. (2012) is similar to the work of Agirre et al. (2009); however, Shilon’s system has a statistical component for ranking proposed translations, which leads to an improvement in BLEU for a small test set. Furthermore, Zollmann and Vogel (2011) use cluster information in syntactic SMT, although not specifically for translating prepositions.

Huang and Knight (2006) propose methods of relabeling syntax trees to improve statistical syntactic translation. Their annotation aims at making the used tag-set (based on the Penn Treebank) less general, assuming that it often fails to capture relevant grammatical distinctions and contexts that are crucial for translation. They distinguish between internal and external annotation. In the case of internal annotation, additional information about the node or its relatives that is otherwise not accessible to the respective node is annotated; this type of annotation consists of lexical and tag information. Their lexicalization strategies include annotating a preposition onto both its parent node (PREP) and its grandparent node (PP), leading to an improvement in BLEU. Other forms of lexicalization consist in annotating information about determiners, auxiliaries and conjunctions. The annotation of tags mainly aims at improving auxiliary and tense errors and is applied to VP-nodes. Furthermore, as external annotation, information about sister nodes and parent nodes, for example, is annotated in order to provide more information about
the context of the respective word or phrase. Huang and Knight (2006) report improvements for most of their annotation strategies.

The method presented in this paper is different from the previous approaches as it combines information about subcategorization and noun classes and it is applied using a purely statistical MT system. Furthermore, by annotating noun class information on NPs and PPs, we aim to introduce a semantic level in contrast to the mainly syntactically motivated annotation scheme of Huang and Knight (2006).

3 Obtaining Noun Class Information

Our system relies on noun class information in order to refine hierarchical translation rules such that they incorporate selectional preferences. In this section, we will describe three approaches to obtain noun class information by classifying noun types into semantic classes: (i) assigning nouns to GermaNet classes; (ii) clustering nouns on the basis of window information and (iii) clustering nouns on the basis of syntactic dependency information. Comparing these disjunctive methods should ensure a systematic assessment of integrating selectional preferences.

3.1 Pre-processing

In order to obtain a consistent noun class annotation, we applied two pre-processing steps to the target-language data prior to computing noun classes using the three variants.

In the first step, we attempt to resolve (possibly) inconsistent parsing decisions for word types tagged both as nouns and named entities. Only words recognized as nouns by the high- coverage morphological analyzer SMOR (Schmid et al., 2004) are considered as common nouns. The remaining instances are considered as named entities; they are classified into organization, location, person and a category for rest (Faruqui and Padó, 2010). Performing this pre-processing ensures that nouns are consistently labeled with the same noun class or named entity category\(^1\). A second benefit is that only nouns, for which we can expect to have either GermaNet coverage or a sound basis for feature extraction, are considered for clustering; “non-nouns” (such as typos or parse-errors which are often very low-frequency and thus likely to deteriorate clustering performance) are excluded from clustering.

The second pre-processing step consists in compound handling: as German noun compounding is very productive and can lead to sparsity and coverage problems, we applied compound splitting to all nouns using a linguistically-informed compound splitter (Fritzinger and Fraser, 2010), which disambiguates competing SMOR analyses relying on corpus statistics.

After pre-processing, noun class information is first computed for head nouns. Then, in a second step, compounds are added into classes based on their head noun. While this might introduce noise for a small number of non-compositional compounds, we assume that the gain in generalization is more important.

3.2 GermaNet

GermaNet (Hamp and Feldweg, 1997; Kunze, 2000) is a lexical resource for German similar to the English WordNet (Fellbaum, 1998). It is a lexical-semantic taxonomy that groups words of the same concept into synsets. For each head noun, we looked up the GermaNet class for a given hierarchical level, to determine the degree of generalization: GermaNet is graph-structured, and extracting the nouns at different levels results in more or less fine-grained sets of classes. We used noun classes from the levels 2, 3, 4, 5, counting from the top level\(^2\).

\(^1\)Note, however, that our type-based annotation method does not take into account polysemy of regular nouns or names that can also be nouns, e.g. Zimmermann (‘carpenter’), which is also a common family name.

\(^2\)Words belonging to several synsets (apple → plant/fruit) are added to the synset with the lowest GermaNet-internal ID. Nouns not covered by GermaNet (16.357 of 211.360 after compound processing) are assigned to a rest class.
Table 1: A selection of political nouns from an example cluster (window-features) and their GermaNet classes.

<table>
<thead>
<tr>
<th>Noun</th>
<th>GermaNet (level 5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minister</td>
<td>minister</td>
</tr>
<tr>
<td>Kanzler</td>
<td>chancellor</td>
</tr>
<tr>
<td>Mehrheit</td>
<td>majority</td>
</tr>
<tr>
<td>Opposition</td>
<td>opposition</td>
</tr>
<tr>
<td>Enthebung</td>
<td>dismissal</td>
</tr>
<tr>
<td></td>
<td>human being</td>
</tr>
<tr>
<td></td>
<td>human being</td>
</tr>
<tr>
<td></td>
<td>group</td>
</tr>
<tr>
<td></td>
<td>configuration</td>
</tr>
<tr>
<td></td>
<td>ending/stop</td>
</tr>
</tbody>
</table>

3.3 Clustering

For clustering, we used the standard k-Means implementation in R (R Core Team, 2013), with features extracted from the target-side part of the parallel data used to train the SMT system and a large web corpus (ca. 45 M sentences total, cf. section 5.1). Low-frequency nouns (f<5 in the combined corpora) were excluded from clustering, and added to the cluster with the nearest centroid in a post-clustering step. We applied two types of features:

- Content words from a window of 10 words to each side of the noun,
- Syntactically-motivated features referring to subcategorization criteria:
  1. prepositions governing the target nouns (“P”),
  2. verbs subcategorizing the target nouns (“VO”),
  3. verbs governing the target nouns in a prepositional phrase (“VPN”),
  4. nouns governing the target nouns in a prepositional phrase (“NPN”).

We observed that particularly the window-based approach induces “topic-like” clusters, see table 1, where a politics-related cluster contains persons (minister, chancellor) and other terms related to politics. In contrast, the classes assigned by GermNet resemble more a generalization over specific noun types as human beings (minister, chancellor) are grouped together and the remaining terms majority, opposition and dismissal are each in a separate group. Using syntactic features for clustering, in particular prepositions, aims at better capturing selectional preferences, and thus obtaining classes that provide salient information for the task of modeling the choice of prepositions in SMT; cf. for example Prescher et al. (2000), Erk et al. (2010), Joanis et al. (2008), Schulte im Walde (2006) and Schulte im Walde (2010) for more information.

A major problem consists in finding a number of clusters that provides both (i) a good representation of the nouns and (ii) the optimal level of abstraction for our SMT-system. In our experiments, we varied the cluster sizes and used sets of 10 – 300 clusters.

4 Using Noun Class Information in SMT

This section presents the basic enriched system and its variants extended with non-annotated baseline rules and new PP rules. In all experiments, a preposition is annotated on both its parent node (PREP) and its grandparent node (PP), as suggested by Huang and Knight (2006).

4.1 Annotating Rules with Noun Classes

Figure 1 illustrates how the target-language parse trees are annotated with noun class information by introducing indices for NP and PP nodes, nouns and prepositions. In this example:

3We work with a string-to-tree system: annotations on the English side of the rules are given only for better readability.
the noun class information serves to create two variants for the translation of learned \[\text{from NOUN}\] PP, namely

\[\text{VP} \rightarrow \text{PP-von-167} \ \text{gelernt}\]

and

\[\text{VP} \rightarrow \text{PP-aus-291} \ \text{gelernt},\]

indicating that nouns of the classes 167 (person) and 291 (abstract concept) represent appropriate fillers for the respective PPs subcategorized by the verb lernen (’to learn’), headed by the prepositions “von” and “aus”, respectively.

### 4.2 Adding Non-annotated NP+PP Rules

Noun class annotation on NP and PP nodes might lead to overly specific rules, resulting in a loss of rule generalization in comparison to the baseline. We thus added baseline rules (rules without cluster annotation) to the enriched rules (we will call this system “BL” in the experimental section). Rules derived from source-target pairs that occurred with \(f \leq 5\) are likely to be random and not useful for selection preferences, so we removed them, leaving only the non-annotated rules (system “BL+cutoff”). Alternatively, we only kept baseline rules with a higher translation probability than the respective annotated rules, thus favoring annotated rules. Rules such as \text{to buy nn1/nn2/nn3/...} are replaced with \text{to buy nn} (system “BL-subst”).

### 4.3 Generating New PP Rules

In addition to the problem that annotated rules can be too specific, not all potentially necessary rules can be obtained from the parallel data. New PP rules are generated by duplicating existing annotated PP rules in which prepositions are substituted. This creates new rules that are not accessible to the baseline and aims at providing the full possible set of rules containing functional prepositions, i.e. prepositions conveying no or only little meaning. Assuming that functional or subcategorized prepositions are the most difficult to translate, the prepositions for which to generate new rules rely on the set of subcategorized prepositions in a subcategorization lexicon (Eckle (1999)). This set comprises 17 prepositions: \(an, auf, aus, bei, durch, für, in, mit, nach, über, um, unter, von, vor, wegen, zu, zwischen\).
Table 2: Newly generated PP translation rules.

<table>
<thead>
<tr>
<th>prep-noun-verb tuple</th>
<th>freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>aus nn-166 lernen</td>
<td>to learn from nn-166</td>
</tr>
<tr>
<td>für nn-166 lernen</td>
<td>to learn for nn-166</td>
</tr>
<tr>
<td>in nn-166 lernen</td>
<td>to learn in nn-166</td>
</tr>
<tr>
<td>mit nn-166 lernen</td>
<td>to learn with nn-166</td>
</tr>
<tr>
<td>von nn-166 lernen</td>
<td>to learn from nn-166</td>
</tr>
<tr>
<td>über nn-166 lernen</td>
<td>to learn about nn-166</td>
</tr>
</tbody>
</table>

Table 3: Subcategorization tuples induced from large monolingual data.

Table 2 shows how the target-side of the original (annotated) rule is multiplied into six new rules containing the prepositions observed in combination with the verb *lernen* and nouns of class 166. The translation probabilities are derived from co-occurrence frequencies in the combined web and target-side part of the parallel corpus (cf. table 3), such as tuples of the form *n-prep-n, prep-n-verb*, etc. Only PP-nodes or PREP-nodes are modified, the rest of the rule (other nodes, terminal symbols and the source-side) remains the same. To keep the amount of generated rules manageable, we used a threshold of $f \geq 5$ to select the rules for which to generate new PP rules and only kept generated rules with a translation probability of $p \geq 0.001$ ("new rules"). Finally, we added both baseline and new rules ("BL+new").

5 Experiments and Results

We used a morphology-aware English-German translation system that first translates into a lemmatized representation, and then generates inflected forms based on morphological features predicted with a sequence model (e.g. Toutanova et al. (2008), Fraser et al. (2012)). This reduces morphological complexity of nominal phrases, and allows in particular to handle portmanteaus (combination of preposition and article: *zur=zu+der: to the*) which are split in pre-processing and merged in a post-processing step. Thus, during translation, prepositions occurring as portmanteaus are represented in the same way as non-portmanteau prepositions.

Table 4 illustrates the processing steps. The lemmatized representation (first column) contains feature markup on nouns for the features *number* and *gender*, which are considered part of the stem. The information about *gender* is obtained from a morphological resource and typically does not vary for a given noun, whereas *number* is indirectly determined by the source-side; as-
To predict morphological features, we trained a sequence model for the features number, gender, case and strong/weak inflection. Each model has access to stems, POS-tags and the feature to be modelled within a window of four positions to the right and the left of the current position. The stem-markup is part of the input to the feature prediction step and is basically propagated over the rest of the phrase, whereas the features case and strong/weak inflection are predicted solely based on context information, i.e. adjacent tags and stems (second column). Based on the predicted morphological features and the lemma, inflected forms can be generated using a morphological resource (third column). Finally, after generating inflected forms, split instances of portmanteau prepositions are merged relying on a simple set of rules as illustrated in the example (zu+der → zur) in the third column of table 4.

### 5.1 Data

We used 1.5 M sentences of parallel data (Europarl and news data from the 2009 WMT shared task), with the target-side part as language model data, to train a string-to-tree Moses system with GHKM extraction (Galley et al., 2004; Williams and Koehn, 2012). The tuning/test sets consist of 1025/1026 news sentences (from the 2009 WMT shared task). The German data was parsed with BitPar (Schmid, 2004). For generating inflected forms, we used the morphological tool SMOR (Schmid et al., 2004). For predicting the morphological features number, gender, case and strong/weak inflection, we trained one CRF for each of the four morphological features using the Wapiti toolkit (Lavergne et al., 2010).

The tuples for modelling translation probabilities for rule generation and the context vectors for clustering were obtained from a combination of the web corpus SdeWaC (44M sentences, Faab and Eckart (2013)) and the German part of the parallel data.

### 5.2 Results

Table 5 presents the results of the systems enriched with noun class information; none of the systems is significantly better than the baseline without semantic class information. Interestingly, the window-based cluster systems are better than the systems using GermaNet or syntactic fea-

---

In contrast to Romance languages, where the merging of portmanteaus (e.g. *a+le=au*) is mandatory, the merging of German portmanteaus is not always necessary. However, preliminary experiments indicated that merging whenever possible is a good strategy.
<table>
<thead>
<tr>
<th>System</th>
<th>BLEU</th>
<th>System</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>13.95</td>
<td>Window10</td>
<td>14.01</td>
</tr>
<tr>
<td>GermaNet-2 (25)</td>
<td>13.93</td>
<td>Window50</td>
<td>14.18</td>
</tr>
<tr>
<td>GermaNet-3 (79)</td>
<td>13.77</td>
<td>Window75</td>
<td>13.69</td>
</tr>
<tr>
<td>GermaNet-4 (175)</td>
<td>13.67</td>
<td>Window100</td>
<td>14.13</td>
</tr>
<tr>
<td>GermaNet-5 (392)</td>
<td>13.67</td>
<td>Window300</td>
<td>13.71</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Syntactic features</th>
<th>P</th>
<th>VO</th>
<th>VPN</th>
<th>NPN</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 classes</td>
<td>13.85</td>
<td>13.85</td>
<td>13.79</td>
<td>13.71</td>
</tr>
<tr>
<td>50 classes</td>
<td>13.84</td>
<td>14.06</td>
<td>14.06</td>
<td>13.91</td>
</tr>
</tbody>
</table>

Table 5: Results for different annotation settings: GermaNet and clusterings based on window information or syntactic features; the scores are averaged over two tuning runs. The numbers in brackets for GermaNet indicate the number of classes and the numbers 2,3,4,5 denote the respective level.

<table>
<thead>
<tr>
<th>System</th>
<th>BL</th>
<th>BL cutoff</th>
<th>BL subst</th>
<th>new rules</th>
<th>BL new</th>
</tr>
</thead>
<tbody>
<tr>
<td>Window75</td>
<td>14.16</td>
<td>13.96</td>
<td>14.07</td>
<td>13.66</td>
<td>14.01</td>
</tr>
<tr>
<td>Window100</td>
<td>14.01</td>
<td>13.94</td>
<td>13.96</td>
<td>14.14</td>
<td>14.02</td>
</tr>
</tbody>
</table>

Table 6: System variants with non-annotated rules and new PP rules.

While GermaNet is a high-quality resource, it tends to suffer from coverage problems and is too fine-grained (for example, the word chancellor is assigned to 2 classes at level 5: organism and living being, which is a distinction that is not needed in our application). On the other hand, the syntactic features are more sparse than window-based features. This is due to the simple fact that we can nearly always extract content words within a window for a given noun, but the extraction of syntactic features is more restrictive and thus, features can only be extracted in the respective syntactic constellation. The window clusters thus seem to provide the most robust representation of selectional preferences. The number of classes does not seem to have a strong overall influence, even though there is a tendency for less classes being favorable.

For three systems (Window50/75/100), we added non-annotated rules (“BL”, “BL-cutoff”, “BL-subst”), new PP rules (“new rules”) and a combination of new and non-annotated rules (“BL+new”), cf. table 6. While there is a moderate improvement for Window75, one of the worst systems in table 5, there is no further gain for the other two systems.

When analyzing the enriched systems’ output, we noticed that on average, more and shorter translation rules than in the baseline systems were used. For example, the enriched systems Window50/75 use on average 11.99/11.62 glue rules per sentence, whereas the baseline system only uses 7.10 glue rules on average. Similarly, the average rule length (here: the length of the target-side of a rule) decreases from 2.19 (baseline) to 1.91/1.92 for the window systems. The average sentence length is stable over these three systems, varying between 25.3 and 25.5 words. Assuming that the use of a low amount of glue rules and long translation rules is preferable, we consider this an indicator of a general problem with the enriched rules: longer and more specific rules in the enriched system do not match anymore and are thus replaced by a combination of shorter rules, resulting in a loss of the context provided by a single longer rule. This contradicts our initial objective of annotating noun class information to add new, generalized information about nouns in order to provide a better basis to model selectional preferences.
more than $100 billion will enter the monetary markets by means of public sales.

mehr als 100 Milliarden Dollar wird die Geldmärkte durch öffentlichen Verkauf gelangen.

more than 100 billion dollar will get money markets by means of public sale.

more than 100 billion dollar auf die Geldmärkte gelangen wird durch den öffentlichen Verkauf.

more than 100 billion dollar get on the money markets by means of the public sale.

the charge that she concentrated too much on foreign affairs , ...

der Vorwurf , dass sie auswärtige Angelegenheiten zu stark konzentriert ist , ...

der Vorwurf , dass sie zu sehr auf die auswärtigen Angelegenheiten konzentriert , ...

one of the local residents even classified the quarrels with eastern european immigrants as a fight for survival.

eine der Anwohner selbst ein Kampf für das Überleben der Streitigkeiten mit osteuropäischen Migranten eingestuft.

one of the residents even the fight for the survival of-the quarrels with eastern european immigrants classified.

one of the residents even classified the quarrels with eastern european immigrants like a fight for survival

Table 7: Examples for better translation of prepositions (BL=Baseline, W=Window50).

of prepositions. Thus, introducing noun classes as a new form of information by the means of parse-tree annotation comes at the cost of losing basic context information as rules spanning over larger chunks are often not available anymore.

5.3 Examples of Improved Translations

Table 7 gives three examples of improvements obtained with the enriched system: in the first sentence, the translation of enter → gelangen requires the preposition auf (to get on), which is correctly produced by the enriched system. Note that it is also possible to translate the phrase enter the money markets without using a preposition in German, for example with the verb erreichen+DIRECT OBJECT (to reach the money markets).

In the second sentence, the preposition for the translation of concentrate on is missing in the baseline, but is correctly produced by the enriched system. In the third sentence, the phrase Kampf für das Überleben (fight for the survival) is somewhat understandable, but the preposition ums (portmanteau: um+das) in the enriched system is a much better choice.

6 Evaluation and Discussion

In this section, we present a more in-depth evaluation in form of assessing the translation quality of prepositions. Furthermore, we illustrate typical problems encountered when translating prepositions, but also show examples for improved sentences. Finally, we discuss why the noun class annotation did not lead to more improvement.
Table 8: Correctly translated English prepositions.

<table>
<thead>
<tr>
<th></th>
<th>for</th>
<th>on</th>
<th>in</th>
<th>at</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>40</td>
<td>24</td>
<td>81</td>
<td>15</td>
</tr>
<tr>
<td>Window50</td>
<td>42</td>
<td>25</td>
<td>85</td>
<td>18</td>
</tr>
<tr>
<td>Total</td>
<td>59</td>
<td>48</td>
<td>110</td>
<td>30</td>
</tr>
</tbody>
</table>

6.1 Translation quality of prepositions

In addition to applying BLEU, we manually evaluated the translation quality of English prepositions, using a set of sentences (5–20 words long) containing the prepositions for, on, in, or at. This test set also includes sentences where a translation of the English preposition as a “null” preposition is necessary or possible, as illustrated in the following examples:

(1a) that lead to a knock-on fall in exports to western europe

(1b) das führt zu einem erheblichen Rückgang der Exporte nach Westeuropa

   that lead to a considerable fall the\_{GEN}(=of the) exports to western europe

(2a) ... has again commented on the problem of global warming

(2b) ... hat erneut ∅ das Problem der globalen Erwärnung kommentiert

In (1), the preposition in can be expressed in form of a genitive modification (der Exporte), but a translation as preposition is also possible. In (2), it is not possible to translate the preposition on when using the verb kommentieren which requires a direct object. However, with the verb sich äußern as translation of to comment, a preposition (zu) is required.

The fact that often several translation variants (e.g. depending on the choice of the verb) are possible makes it difficult to directly compare the systems’ output to a reference translation. We considered a preposition to be correctly translated if the produced PP or NP is an acceptable translation in the German sentence. Table 8 shows that the enriched system is slightly better than the baseline system, but overall there is only a small difference.

In general, we found it difficult to observe a systematic behaviour or “pattern” of (types of) prepositions or contexts that are handled better or worse in the enriched system in comparison to the baseline system. However, we noticed that there is a type of prepositions that seems to be especially hard to translate, namely prepositions with a predominantly literal meaning occurring in an infrequent subcategorized context. These are often mistranslated, in both our baseline and enriched systems, as illustrated in the following example:

(3a) for example, germany has been criticized for passivity

(3b) beispielsweise hat Deutschland *für Passivität kritisiert worden

   for example, Germany has *for passivity criticized been

(3c) wegen Passivität wurde zum Beispiel Deutschland kritisiert

The preposition for is often used literally and thus can be translated in a straightforward manner, e.g. with the preposition für. In this subcategorized context (criticized for) however, it expresses a cause, which makes für a totally inappropriate translation, cf. (3b). In contrast, wegen (because of) is the correct translation, as can be seen in the reference translation (3c). We noticed that similar constructions such as “detain FOR corruption” (WEGEN Korruption verhaften) or “look FOR sth.” (NACH etwas suchen) seem to be prone to the same error.
These examples provide insight into the complexity of the task of translating prepositions. Depending on the respective context of a PP, different factors such as the relation of being a merely functional preposition (i.e. subcategorized) vs. conveying a meaning, as well as the class of the involved noun seem to play roles of varying importance. However, with our rather inflexible annotation method we are not able to act in a context-dependent manner, but always provide the same type of information at the same level of granularity.

6.2 Conclusion

We started with the hypothesis that noun class information is useful to model selectional preferences in preposition translation rules. However, annotating semantic class information on NP/PP nodes of the parse trees in a string-to-tree system amounts to a hard constraint and our experiments indicate that this form of annotation leads to overly specific rules. We tried to compensate for this by making the non-annotated rules available and by adding new PP rules synthesized from monolingual data. However, previous work, such as e.g. the work of Marton and Resnik (2008), has shown that soft constraints often work better than hard constraints. It might therefore make sense to model selectional preferences through the use of feature functions which reward good choices, rather than markup in the string-to-tree grammar, but this would require extensive changes to the model and decoder.

Another problem with our approach is that there is no generally applicable optimal level of selectional preferences. This is in line with semantic research on selectional preferences as verb subcategorization features (Schulte im Walde, 2006; Joanis et al., 2008): across subcategorizing words, it is difficult to identify a generally acceptable semantic level of generalization in lexical resources. Because of this, the parse tree annotation is not flexible enough to take into account the varying needs of different contexts, as it always leads to rules of the same degree of specificity, and therefore cannot adapt to the respective contexts.

With regard to resources, we found that none of the variants we considered was able to obtain noun class information that is optimal: WordNets in general are known to be very fine-grained and contain many ambiguities, making it difficult to derive generally applicable noun groups (Navigli, 2006; Palmer et al., 2007). In contrast, window clusters might not contain the appropriate selectional preference information as they resemble topic clusters rather than a generalization over specific noun types. As opposed to the unstructured information used for the window clustering, the syntactic dependencies constitute the type of information that is needed to determine a valid preposition for a given context, i.e. the governing verb/noun or the noun in the PP. Thus, clusters learned from syntactic features were expected to better capture selectional preferences. However, these clusters failed to lead to improvements and had worse performance than the window clustering.

This work thoroughly explored different methods to obtain noun class information (exploiting distributional and resource-based information), but found that none of these variants is optimal. While each of these strategies has some advantages (e.g. either high-quality or high coverage), they also suffer from weaknesses (low coverage or too fine-grained) that could be (at least partially) addressed by combination with another method. For example, GermaNet could be used to provide a high-quality initial set of noun classes that is then expanded relying on distributional information providing a wide-range coverage. Combining the advantages of the resources we considered could lead to a more promising strategy to obtain classes providing salient information on selectional preferences and constitutes a challenging task for future work.
Acknowledgements

This work was funded by the DFG Research Projects Distributional Approaches to Semantic Relatedness and Models of Morphosyntax for Statistical Machine Translation – Phase 2 and the DFG Heisenberg Fellowship SCHU-2580/1-1. We would like to thank several anonymous reviewers for their helpful comments.

References


Predicting Human Translation Quality

Lucia Specia  
l.specia@sheffield.ac.uk
Kashif Shah  
kashif.shah@sheffield.ac.uk
Department of Computer Science, University of Sheffield  
Regent Court, 211 Portobello Street, S4 1DP, UK

Abstract

We present a first attempt at predicting the quality of translations produced by human, professional translators. We examine datasets annotated for quality at sentence- and word-level for four language pairs and provide experiments with prediction models for these datasets. We compare the performance of such models against that of models built from machine translations, highlighting a number of challenges in estimating quality and detecting errors in human translations.

1 Introduction

Metrics for translation quality estimation (QE) (Blatz et al., 2004; Specia et al., 2009) aim at providing an estimate on the quality of a translated text. Such metrics have no access to reference translations, as they are intended for translation systems in use. QE has shown promising results in several applications in the context of Machine Translation (MT), such as improving post-editing efficiency by filtering out low quality segments which would require more effort to correct than translating from scratch (Specia et al., 2009; Specia, 2011), selecting high quality segments to be published as they are, without post-editing (Soricut and Echihabi, 2010), ranking or selecting the best translation from multiple MT systems (Specia et al., 2010; Hildebrand and Vogel, 2013; Avramidis, 2013; Avramidis and Popović, 2013), or between translations from either an MT system or a translation memory (He et al., 2010), and highlighting sub-segments that need revision (Bach et al., 2011).

Generally speaking, QE models are built using supervised machine learning algorithms from examples of translations at a given granularity level (e.g. sentences). For training, these examples are annotated with quality labels and described by a number of features that can approximate quality (or errors). “Quality” is therefore defined according to the problem at hand and the labelled data, for example, post-editing time for a sentence or word-level errors. For an overview of various algorithms and features we refer the reader to the WMT12-14 shared tasks on QE (Callison-Burch et al., 2012; Bojar et al., 2013, 2014).

So far, QE has only been applied to machine translated texts. However, the above mentioned applications are also valid in the context of human translation. In particular, in scenarios where translations produced by humans may be of variable or questionable levels of reliability (e.g. crowdsourcing), it becomes important to estimate translation quality to, for example, select among multiple options of human translations (or even a mix of human and machine translations). In addition, even with professionally created translations, quality assurance is a common process and an estimation method could be useful, for example, to sample the lowest quality cases for checking/revision.

Even though it is known that human translations are generally different from machine translations, we put forward the hypothesis that it is possible and useful to have automated
metrics to estimate translation quality of both human and machine translations. In this paper we analyse existing human translations annotated for quality and errors and contrast them to machine translations. We use this data to experiment with an existing framework for quality estimation to predict quality in human translations. More specifically, we aim at answering the following questions:

1. Can we automatically distinguish machine from human translations?
2. Do professional human translators make mistakes?
3. Are human translation errors the same as machine translation errors?
4. Can quality estimation approaches capture issues in human translations?

We discuss each of these questions in Sections 3, 4, 5, and 6, respectively. Before that, we introduce the datasets and settings used in our experiments in Section 2.

2 Datasets and experimental settings

2.1 Datasets

Our datasets are those used for the WMT14 shared task on quality estimation\(^1\) and were produced in the context of the QTLaunchPad project.\(^2\) They contain news texts in four language pairs (Table 1): English→Spanish (en-es), Spanish→English (es-en), English→German (en-de), and German→English (de-en). Each language pair dataset contains a different number of source sentences and their human translations, as well as 2-3 versions of machine translations: by a statistical (SMT) system, a rule-based (RBMT) system and, for en-es/de only, a hybrid system. Source sentences were extracted from tests sets of WMT13 and WMT12, and the translations were produced by top MT systems of each type (SMT, RBMT and hybrid – hereafter MT-1, MT-2, MT-3) which participated in the translation shared task in 2103, plus the professional translation provided by WMT as reference (HT). In addition, for the word-level analysis, for all language pairs except English→Spanish, which already had enough sentences, we included some customer data (mostly technical documentation) provided and annotated by language service providers as part of the QTLaunchPad project.

This data is very different from existing corpora of human translations annotated for quality. Existing resources contain translations from students, while ours only contain translations produced by professional translators, and annotated by (other) professional translators. In addition, our data contains translations from multiple state of the art MT systems, also annotated by professional translators. For comparison purposes, in the remaining of the paper we report statistics for the human versus all MT data together.

Sentence-level data At sentence-level, the details about the datasets are given in Table 1. All translations for each source sentence were annotated by a single professional translator (and that one translator annotated all sentences for a given language pair) using the following three options representing the translator’s perception on the effort that would be needed to post-edit such a sentence:

- **1** = Perfect translation, no post-editing needed at all.
- **2** = Near-miss translation: translation contains a maximum of 2-3 errors, and possibly additional errors that can be easily fixed (capitalisation, punctuation, etc.).
- **3** = Very low quality translation, cannot be easily fixed.

\(^1\)http://www.statmt.org/wmt14/quality-estimation-task.html

\(^2\)http://www.qt21.eu/launchpad/
### Word-level data
For word-level annotation, a subset of sentences of type “2” (near-miss) from MT systems and from human translators (Table 2) were annotated with core issue types (errors) of the Multidimensional Quality Metric (MQM), as shown in Figure 1. In addition to the 16 fine-grained labels, two levels of labels were automatically generated by climbing up the MQM hierarchy: Accuracy versus Fluency, and OK (no issue) versus BAD (any issue). Each translation was annotated by 1-5 professional translators. For translations annotated by more than one translator, only one annotation was randomly selected and used in our analysis. For a discussion on annotator agreement within these datasets, see (Lommel et al., 2014).

![Figure 1: MQM core issue types used for the word-level annotation task.](image)

### Table 1: Number of source and target sentences labelled for post-editing effort at sentence-level.

<table>
<thead>
<tr>
<th># Source</th>
<th># HT+MTs</th>
<th># Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,104 English</td>
<td>4</td>
<td>4,416 Spanish</td>
</tr>
<tr>
<td>500 English</td>
<td>4</td>
<td>2,000 German</td>
</tr>
<tr>
<td>500 German</td>
<td>3</td>
<td>1,500 English</td>
</tr>
<tr>
<td>500 Spanish</td>
<td>3</td>
<td>1,500 English</td>
</tr>
</tbody>
</table>

### Table 2: Number of sentences labelled at word-level, from news and technical domains.

<table>
<thead>
<tr>
<th>Source→target</th>
<th># WMT (news)</th>
<th># Technical</th>
</tr>
</thead>
<tbody>
<tr>
<td>English→Spanish</td>
<td>2,339</td>
<td>-</td>
</tr>
<tr>
<td>English→German</td>
<td>467</td>
<td>398</td>
</tr>
<tr>
<td>German→English</td>
<td>250</td>
<td>200</td>
</tr>
<tr>
<td>Spanish→English</td>
<td>440</td>
<td>610</td>
</tr>
</tbody>
</table>

### 2.2 Settings
Prediction models are only built for sentence-level, given the small number of human translations labelled at word-level (at most 294, for en-es). Our word-level analysis focuses on error distributions. For the building and evaluation of sentence-level prediction models (as described in Sections 3 and 6), we use the following settings.

**Dataset splits** We use the standard training and test splits as distributed by WMT14: each MT system or HT dataset is split into 70% for training and 30% for test.

[3](http://www.qt21.eu/launchpad/content/background-and-principles)
Learning algorithms We use the Support Vector Machines (SVM) implementation within the QuEst toolkit for quality estimation\(^4\) (Specia et al., 2013; Shah et al., 2013) to perform classification (SVC) (Section 3) and regression (SVR) (Section 6) with Radial Basis Function as kernel and parameters optimised using grid search.

Evaluation metrics To evaluate our models, we use standard metrics for regression (MAE: mean absolute error) and classification (precision, recall and F1). In all tables, bold-faced figures are significantly better (paired t-test with \(p \leq 0.05\)) wrt the baseline for the given language pair. As baseline for the regression models, we consider the Mean of the training data, i.e., simply outputting the average value of the training set to all test instances. Similarly, as baseline for the classification models, we consider assigning the most frequent class (MC) in the training set to all test instances.

Features We use the QuEst toolkit to extract two feature sets for each dataset:

- Baseline features (BL): 17 features used as baseline in the WMT shared tasks on QE. Examples of baseline features for sentence-level include the following:
  - no. of tokens in the source & target texts
  - average source token length
  - average no. of occurrences of target words in target text
  - no. of punctuation marks in source & target texts
  - language model probability of source & target texts using LMs built from large source/target language corpora of human texts
  - avg. no. of translations per source word built using lexical tables from the IBM 1 model thresholded such that \(P(t|s) > 0.2\)
  - % of 1-grams, 2-grams & 3-grams in frequency quartiles 1 & 4 (lower/higher frequency) in a large corpus of the source language
  - % of 1-grams in source text seen in a large corpus of the source language

- All features (AF): 80 common MT system-independent features (superset of BL).

The resources used to extract all features (language models, etc.) are available as part of the WMT14 shared task on QE.

3 Can we distinguish machine from human translations?

In this experiment we train an SVM classifier to distinguish human translations from machine translations at sentence-level. We put together all MT and human translations for each language pair, label all human translations as 1, and all system translations as 0. We then train a binary classifier to distinguish them. Results are given in Table 3, where MC stands for “majority class” (always picking MT). They show a large variation across language pairs, although MC is outperformed in all cases in terms of F1. The lower performance for en-es and en-de may be because here translations from three MT systems are put together (only 25% of the examples are HT), while for the remaining datasets, only two MT systems are available, and therefore the data distribution is less skewed (33% of the examples are HT). Nevertheless, figures for en-es are substantially better than those for en-de, possibly because of the larger size of the en-es dataset.

\(^4\)http://www.quest.dcs.shef.ac.uk/
With similar classifiers (albeit different datasets), Gamon et al. (2005) reported as trivial the problem of distinguishing human translations from machine translations. However, our results seem to indicate that this is now a harder problem than some years ago, possibly pointing in the direction that MT systems produce more translations that are better in quality, and therefore closer to human translation nowadays. Moreover, human translations also contain errors, which gives us a further motivation for modelling the prediction of quality in human translations (see Figure 2).

<table>
<thead>
<tr>
<th></th>
<th>Model</th>
<th>#feats</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>en-de</td>
<td>MC</td>
<td>-</td>
<td>0.3041</td>
<td>0.1316</td>
<td>0.1566</td>
</tr>
<tr>
<td></td>
<td>BL</td>
<td>17</td>
<td>0.3272</td>
<td>0.1200</td>
<td>0.1756</td>
</tr>
<tr>
<td></td>
<td>AF</td>
<td>80</td>
<td>0.3281</td>
<td>0.1193</td>
<td>0.1801</td>
</tr>
<tr>
<td>de-en</td>
<td>MC</td>
<td>-</td>
<td>0.5041</td>
<td>0.2416</td>
<td>0.2961</td>
</tr>
<tr>
<td></td>
<td>BL</td>
<td>17</td>
<td>0.5420</td>
<td>0.2321</td>
<td>0.3262</td>
</tr>
<tr>
<td></td>
<td>AF</td>
<td>80</td>
<td>0.5468</td>
<td>0.2333</td>
<td>0.3271</td>
</tr>
<tr>
<td>en-es</td>
<td>MC</td>
<td>-</td>
<td>0.6541</td>
<td>0.1521</td>
<td>0.2312</td>
</tr>
<tr>
<td></td>
<td>BL</td>
<td>17</td>
<td>0.7012</td>
<td>0.1524</td>
<td>0.2561</td>
</tr>
<tr>
<td></td>
<td>AF</td>
<td>80</td>
<td>0.7188</td>
<td>0.1533</td>
<td>0.2527</td>
</tr>
<tr>
<td>es-en</td>
<td>MC</td>
<td>-</td>
<td>0.7311</td>
<td>0.3513</td>
<td>0.4625</td>
</tr>
<tr>
<td></td>
<td>BL</td>
<td>17</td>
<td>0.7665</td>
<td>0.3651</td>
<td>0.4942</td>
</tr>
<tr>
<td></td>
<td>AF</td>
<td>80</td>
<td>0.7639</td>
<td>0.3667</td>
<td>0.4954</td>
</tr>
</tbody>
</table>

Table 3: Performance of classifier to distinguish between human translations and machine translations (all MT systems together). “MC” corresponds to always picking machine translation (most frequent) as label.

4 Do professional human translators make mistakes?

In order to answer this question, we look at the distribution of the 1-3 scores at sentence-level (Figure 2) and the distribution of OK versus BAD word-level labels (Figure 3). Both sets of distributions show that, for all language pairs, human translations (HT), albeit professionally produced, contain errors. In the sentence-level figures, the first set of bars for all language pairs show that in the best case only about 80% of the human translations are labelled “1” (perfect). While – not surprisingly – very low quality translations (label “3”) are virtually non-existent (maximum 1.2%), many cases of near-misses are found for all language pairs. For English→Spanish, 27% of the translations are considered near-misses, whereas for other languages pairs this rate is between 15 and 20%. The bars for MT systems essentially show the inverse behaviour: very few perfect translations (less than 10% for all language pairs except Spanish→English), predominantly near-miss translations for English↔Spanish, and a mostly even distribution between very low quality and near-miss translations for German↔English.

It is worth noticing that the translators annotating datasets for errors received explicit guidelines to consider only true errors for the annotation. They were instructed not to label any segment/word as incorrect or near-miss because of preferential changes, i.e., because they would simply have preferred a different translation. They were also instructed to consider a segment/word correct when they were not sure about such a segment/word because of lack of context, style guidelines, etc. Some examples of near-miss human translations (with issues highlighted and identified) are shown in Table 4.

Looking at the distribution of OK and BAD word-level annotations (Figure 3), we see that even though both HT and MT segments had already been pre-labelled as near-misses (i.e., as
Figure 2: Percentage of 1-3 scores given as labels at sentence-level data for human (HT) and each machine (MT-i) translation system.
<table>
<thead>
<tr>
<th>Lang.</th>
<th>Source</th>
<th>Target</th>
<th>Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>de-en</td>
<td>Deutsche Welle: Anfang der Woche hatte Deutschland zunächst signalisiert, dass es gegen den Antrag auf einen Beobachterstatus der Palästinenser bei den Vereinten Nationen stimmen würde.</td>
<td>Deutsche Welle: At the beginning of the week, Germany had initially signalled that it would vote against the Palestinians’ application for observer status within the United Nations.</td>
<td>agreement</td>
</tr>
<tr>
<td>en-de</td>
<td>So I had plenty of time to think about the subject of boredom.</td>
<td>So hatte ich viel Zeit, um an das Thema der Langeweile zu denken.</td>
<td>grammar</td>
</tr>
<tr>
<td>en-es</td>
<td>People assume we are like the Bullingdon Club without meeting us.</td>
<td>La gente supone que parezcamos al Club Bullingdon sin vernos</td>
<td>mistranslation, function words, mistranslation</td>
</tr>
<tr>
<td>es-en</td>
<td>La princesa D’Arenberg guarda sus vestidos de fiesta del modisto con “los máximos cuidados... porque un vestido no es solamente un vestido, también es el conjunto de recuerdos que conlleva“.</td>
<td>Princess D’Arenberg looks after her couturier gowns with &quot;the utmost care... because a dress not just a dress, it’s also the many memories that go with it.</td>
<td>terminology, omission</td>
</tr>
</tbody>
</table>

Table 4: Examples of near-miss human translations. Issues are highlighted and listed in order.

containing 1-3 errors that are easy to fix), as expected, MT segments contain more errors for all language pairs. Only up to 10% of the words in HT segments contain errors. For MT, this percentage reaches 40% for English→Spanish.

5 Are human translation errors the same as machine translation errors?

To answer this question we look at the distribution of specific issue types coming from the word-level annotation. In Figure 4 we show the distribution of errors in HT and MT grouped by fluency and accuracy types (here we ignore the “OK” category for clarity purposes). Once again, these statistics only consider segments that had already been pre-labelled as near-misses. For all language pairs except English→German, MT segments tend to contain considerably more words labelled as having fluency issues than as containing accuracy issues. In human translations, however, fluency issues are more frequent in language pairs involving Spanish, whereas accuracy issues are more frequent in language pairs involving German, although English→German shows a close distribution between accuracy and fluency issues. This seems to indicate that the types of errors in translations may be more dependent on the language pair than on the type of translation (MT or HT).

A more detailed view on the types of errors by HT and MT is given in Figure 5. Here we look at percentages of specific issues (again ignoring the “OK” category) in human and machine (a mixture of all MT systems) near-miss translations. Given the limited size of the datasets, some issues are not observed for certain language pairs. Overall, the distributions of specific issue types are very distinct in HT and MT segments, as well as across language pairs. Mis-translation is by far the most frequent error type in human translations for German↔English. For Spanish↔English, fluency errors are the most frequent. We note that the latter are not a combination of all errors under “fluency” in Figure 1. Instead, they are a more general category that annotators were asked to use when they could not flag the specific fluency issue with the word.
Figure 3: Percentage of words labelled as OK versus BAD in human (HT) and machine (MT) near-miss translations (MT contains a mixture of all MT systems). The table shows the number of words tagged for issues, including the “OK” tag, which in fact means that no issue was found for the word.

Figure 4: Percentage of words labelled as containing fluency versus accuracy issues in human (HT) and machine (MT) near-miss translations (MT contains a mixture of all MT systems).
Figure 5: Percentage of words labelled with each type of MQM issue in human (HT) and machine (MT) near-miss translations.
Can quality estimation approaches capture issues in human translations?

In what follows we show the performance of regression models trained on HT and MT data independently (Table 5), for the sentence-level annotated data. The performance obtained for models trained on MT data is comparable to the state of the art, based on the results of the latest WMT14 shared task (Bojar et al., 2014). In absolute terms, the figures show that models trained on HT datasets are better (lower MAE) than models trained on any MT dataset, for all language pairs. That could be seen as indicative that tools used for MT quality estimation are also applicable for HT quality estimation. However, although all HT and MT models were trained on datasets of the same size, the distribution of scores in each of these datasets is very different (see Figure 2). Human translations are “perfect” in approximately 80% of the cases for all languages. Therefore, it becomes much harder to outperform the “Mean” baseline in HT models. This is reflected in the consistently lower MAE scores obtained by the Mean baseline on the HT data. Therefore, a better way of comparing the performance of models for HT against models for MT is to measure the improvement on the MAE scores between the Mean baseline and the best (AF) prediction model. The absolute and relative improvements for each language pair are shown in Figure 6. In terms of absolute improvement, the figures for MT are always more substantial than those for HT. This is also the case in relative terms, except for German→English, where the HT model achieves relatively better improvement over the Mean baseline than the MT models, although the difference is minor (18% improvement versus 15% improvement).

Our results seems to indicate that it is generally harder to predict human translation quality. In addition to the highly skewed data distribution, one reason for that could be that errors in human translations may be more subtle than in machine translations, requiring more sophisticated features than the ones used in current quality estimation approaches. In fact, another interesting finding from Table 5 is that there is zero or little gain for moving from the BL to the AF feature sets for HT, whereas the gain is evident for models built from MT data. This seems to indicate again that the features we resort to are not appropriate or sufficient to capture the quality of human translations.

To further inspect this problem, we take the MQM core issue types (see Figure 1) as guidance on the types of quality issues features should attempt to capture. We note that many issue types are not covered at all or only approximated by features in current quality estimation approaches. In what follows we provide a discussion for each issue type:  

5 A detailed description of the issue types can be found on [http://www.qt21.eu/launchpad/content/list-mqm-issue-types](http://www.qt21.eu/launchpad/content/list-mqm-issue-types)
### Table 5: Error (MAE) scores for prediction models built for each language pair and translation system.

<table>
<thead>
<tr>
<th></th>
<th>en-de</th>
<th>en-es</th>
<th>de-en</th>
<th>es-en</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Model #feats</td>
<td>MAE</td>
<td>Model #feats</td>
<td>MAE</td>
</tr>
<tr>
<td></td>
<td>HT</td>
<td></td>
<td>HT</td>
<td></td>
</tr>
<tr>
<td>BL</td>
<td>Mean - 0.3552</td>
<td>0.3350</td>
<td>BL  Mean - 0.3883</td>
<td>0.3633</td>
</tr>
<tr>
<td></td>
<td>BL 17</td>
<td>0.3325</td>
<td>BL 17</td>
<td>0.3519</td>
</tr>
<tr>
<td>AF</td>
<td>80</td>
<td>0.3519</td>
<td>AF 80</td>
<td>0.3519</td>
</tr>
<tr>
<td>MT-1</td>
<td>Mean - 0.4577,0.435</td>
<td>0.4422</td>
<td>MT-1 Mean - 0.4228,0.3821</td>
<td>0.3714</td>
</tr>
<tr>
<td></td>
<td>BL 17</td>
<td>0.435</td>
<td>BL 17</td>
<td>0.3714</td>
</tr>
<tr>
<td></td>
<td>AF 80</td>
<td>0.435</td>
<td>AF 80</td>
<td>0.3714</td>
</tr>
<tr>
<td>MT-2</td>
<td>Mean - 0.5782,0.5412</td>
<td>0.4912</td>
<td>MT-2 Mean - 0.5782,0.4720</td>
<td>0.4002</td>
</tr>
<tr>
<td></td>
<td>BL 17</td>
<td>0.4912</td>
<td>BL 17</td>
<td>0.4002</td>
</tr>
<tr>
<td></td>
<td>AF 80</td>
<td>0.4912</td>
<td>AF 80</td>
<td>0.3902</td>
</tr>
<tr>
<td>MT-3</td>
<td>Mean - 0.6000,0.5782</td>
<td>0.4818</td>
<td>MT-3 Mean - 0.6000,0.4720</td>
<td>0.4002</td>
</tr>
<tr>
<td></td>
<td>BL 17</td>
<td>0.4818</td>
<td>BL 17</td>
<td>0.3902</td>
</tr>
<tr>
<td></td>
<td>AF 80</td>
<td>0.4818</td>
<td>AF 80</td>
<td>0.3902</td>
</tr>
</tbody>
</table>

**Accuracy**

- **Terminology**: Normative terminology infringed. This issue is not directly covered by current approaches to quality estimation. However, as a proxy to it, both monolingual (target) and bilingual terminology lists could be used for simple checks, such as whether all content words (or nouns) in the translation belong to the terminology list.

- **Mistranslation**: Incorrect word translation chosen (overly literal, false friend, should not have been translated, entity, date/time/number, unit conversion). This issue cannot be easily automated, apart from some mechanical checks on date/time/number format.

- **Omission**: Translation for source word is missing. Certain existing features approximate this issue type, e.g., source versus target segment word counts, counts of words with certain POS tags in both source and target segments, and language models of the target language, which can detect unusual constructions due to – among other things - omissions.

- **Addition**: Word that is not in the source segment is added to the translation. Existing features approximate this issue as in the case of “omission”.

- **Untranslated**: A source word is left untranslated in the translation. This issue is currently covered by out-of-vocabulary features based on language model of the target language.
Fluency

- **Register/style**: Incorrect use of words due to variants/slang, company style or style guide. This issue is not directly covered by existing approaches, but it is approximated by the target language model features, as long as this model is trained on documents with the correct register/style.

- **Spelling**: Incorrect word spelling due to capitalisation or diacritics. This issue is also approximated by language model features, which are trained on truecased models. Spell checkers could also be used.

- **Typography**: Incorrect use of punctuation, unpaired quote marks or brackets. These issues are captured by a number of features, such as those checking for missing closing brackets or quotation symbols in the target segment, and those contrasting the percentage of different punctuation symbols in the source and target languages.

- **Grammar**: The several grammar-related issues (morphology, part of speech, agreement, word order, function words, tense/mood/aspect) are captured partly by target language model features, and partly by advanced syntactic features based on probabilistic context free grammars, dependency structures and categorical combinatory grammar (Felice and Specia, 2012; Almaghout and Specia, 2013).

- **Unintelligible**: Parts of the translation are not understandable enough to be analysed. This issue is only approximated by language model features of the target language.

7 Conclusions

This paper has presented an analysis and experiments on quality prediction of professionally produced translations. The data analysis has shown that although intuitively we know that human translations differ significantly from machine translations, distinguishing them using automated methods is not a trivial task. In particular, it seems to be a harder problem nowadays then it was ten years ago. This is most likely due to overall improvements in the quality of machine translation systems over the time. In addition, the human translations analysed, albeit professionally created, contain errors in up to almost 30% of the cases. We have shown that the types of errors in human translations tend to be different from those in machine translations, but that larger differences are observed across language pairs.

Finally, we have shown that human translation quality seems harder to estimate than machine translation quality. We believe this is mostly due to two reasons: skewed label distribution (most human translations are labelled as perfect), and the limitations of existing features, which do not capture more subtle or complex issues present in human translations. Our on-going work is aimed at addressing these two challenges: we are collecting a larger dataset including more lower quality human translations (produced by less experienced translators) and designing more linguistically motivated features.

Acknowledgements

This work was supported by funding from the from European Union’s Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 296347 (QTLaunchPad).

References


Data Selection for Compact Adapted SMT Models

Shachar Mirkin
Xerox Research Centre Europe, Meylan, France
shachar.mirkin@xrce.xerox.com

Laurent Besacier
LIG, University Of Grenoble Alps, Grenoble, France
laurent.besacier@imag.fr

Abstract
Data selection is a common technique for adapting statistical translation models for a specific domain, which has been shown to both improve translation quality and to reduce model size. Selection relies on some in-domain data, of the same domain of the texts expected to be translated. Selecting the sentence-pairs that are most similar to the in-domain data from a pool of parallel texts has been shown to be effective; yet, this approach holds the risk of resulting in a limited coverage, when necessary n-grams that do appear in the pool are less similar to in-domain data that is available in advance. Some methods select additional data based on the actual text that needs to be translated. While useful, this is not always a practical scenario. In this work we describe an extensive exploration of data selection techniques over Arabic to French datasets, and propose methods to address both similarity and coverage considerations while maintaining a limited model size.

1 Introduction
Data selection (DS) is a key method for domain adaptation. It is commonly used in statistical machine translation (SMT) for selecting from a pool of parallel sentences a subset that is more similar to the target domain, and using it to train the translation model, the language model, or both. Data selection can also be used when more compact models are needed due to memory limitations, for instance. A wide variety of selection methods have been used over the years, where the main principle is to measure the similarity of sentences from the pool to some in-domain data, either the development or the (source side of the) test set. Such similarity is often based on information theory metrics, like perplexity, applied to either side of the training data (source or target) or – as often turned out to be more effective – to both. One problem these methods suffer from is that they “overfit” the in-domain data, since sentences with unseen n-grams are less likely to be selected. Some methods address this coverage issue by expanding the selected training data to include more instances of previously unseen or infrequent n-grams. Techniques based on information retrieval (IR) have also been widely used for data selection, especially for choosing training or tuning data based on the test set, either from parallel corpora or by mining comparable corpora to find relevant parallel phrases or sentences. This produces highly adapted models for the test set, and generally reduces the number of out-of-vocabulary (OOV) words. The limitation of such methods is that they typically rely on the availability of the text for translation before the final model is produced. While sometimes possible, this is not always the case.

This work proposes an in-depth investigation of several data selection techniques for SMT adaptation. We experiment with less conventional and challenging domains for Arabic to French translation (translation of Web blogs and of dialectal conversation transcripts), and with very
little available in-domain data. Our work provides insights into the usefulness of the different selection methods to each dataset.

In addition, we propose two data selection methods that aim to ease the tension between the similarity and coverage objectives, while the need for compact models that require rather aggressive data filtering, is taken into account. In the first proposed method, an IR model is added to a perplexity-based one, resulting in a model that is ready to use with or without relying on the test set. The second, denoted AVSF, is a domain-adaptation-fitted version of VSF (Lewis and Eetemadi, 2013), an algorithm which was designed to reduce model size without jeopardizing \( n \)-gram coverage. Both proposed methods are able to show competitive results in comparison to prior DS techniques. Over one dataset, their performance is better than other assessed methods; over the other – where data selection with size constraints proved more difficult – they outperform methods with comparable training data size, demonstrating a good tradeoff between translation performance and model size.

The rest of the paper is organized as follows. Section 2 presents common approaches for data-selection in SMT. Section 3 details the setting of our experiments. Experiments based on information theory metrics are presented in Section 4. Our proposed methods are described in Sections 5 (IR-based adaptation) and 6 (adapted VSF). Section 7 shows a summary of the results and outlines our main conclusions.

## 2 Data selection for domain adaptation

Data selection is a way to adapt the about-to-be-trained model by using only the part of the training data that is more similar to the target domain. DS is a general approach, that has been applied to other tasks other than SMT, including Chinese word segmentation and Part-of-Speech tagging (Song et al., 2012). In SMT, DS is common practice for domain adaptation, where a subset of the bilingual parallel corpus is used for training some or all the models comprising typical phrase-based SMT models (translation, reordering and language models). Apart from better adaptation, data selection has the advantage of making the training set, and in turn - the generated models, smaller. This is a factor we consider in this work, avoiding methods or parameters with which the training data is not significantly smaller than the entire set.

In this section we review the most common techniques used for selecting training data in SMT. The assumption is that we have a small in-domain corpus, denoted \( I \), and a large “general” bilingual corpus, \( G \), a pool from which we wish to select bi-sentences that will help better translate texts of the same domain as \( I \).

### 2.1 Information theory metrics

One prominent line of research is using information theory metrics to assess each one of the sentences in the pool, \( G \), and choose the ones that are most similar to the provided in-domain data, \( I \).

**Perplexity (PP) and cross-entropy (CE)** Perplexity (PP) is perhaps the best known metric for DS for SMT. The idea is to compute the perplexity of a language model (LM) built on \( I \), measure the perplexity of this LM over each sentence in \( G \), and select the \( m \) sentences with the lowest PP scores. Gao et al. (2002) and Moore and Lewis (2010) employed this metric for language model adaptation, and in (Foster et al., 2010; Yasuda et al., 2008) it was used for translation model adaptation. The same technique is sometimes (e.g. Moore and Lewis (2010)) referred to as cross entropy (CE). Since, by definition, the cross-entropy is simply the exponent in the perplexity score, the base of the exponent is 2 and all scores are positive, a smaller PP means a smaller CE and vice versa. In other words, selecting \( m \) sentences with the lowest PP scores or with the lowest CE scores is equivalent.
Cross-entropy difference (CED) While PP (or its equivalent CE) were used extensively before, Moore and Lewis (2010) proposed using the difference between the cross entropies scores with respect to the in-domain corpus and with respect to the general corpus. The idea is to prefer sentences that are typical to the in-domain (i.e. low CE) and untypical to the general domain (i.e. high CE). As above, the sentences with the lowest cross entropy difference are selected. Axelrod et al. (2011) have extended over the CED method by computing it bilingually, taking the average of the CED over the source and over the target. They showed that bilingual CED outperforms the monolingual version, as well as other related metrics. In their experiments, best results were achieved when the in-domain and the reduced-general corpora were placed in two different translation tables. This method is still considered the state of the art in this line of research.

In our experiments we assess all the above metrics, both monolingually and bilingually, over two different datasets. We validate that bilingual data selection works best for translation models, but show that target-based selection sometimes outperforms it for language model training. Further, quite intuitively, we learn that when the general corpus is rather similar to the in-domain one, perplexity outperforms cross entropy difference. In such cases, data selection serves mainly for the purpose of reducing model size rather than performance improvement. See Section 4 for further details about these results.

2.2 Retrieval-based selection
A different approach for data selection is based on Information Retrieval (IR). Typically, the data is selected based on the source side of the test set. This makes this approach appropriate only for certain scenarios, when the text to translate is known in advance, and when the translation can be delayed until a model is constructed for it. Eck et al. (2004) used this approach to adapt the LM of an SMT model; Hildebrand et al. (2005) adapted both the translation model (TM) and the LM by using the Lemur IR system\(^\text{1}\) for searching the training set for sentences that are similar to each of the test set sentences, and training the model over the retrieved dataset. Perplexity was used in their work to determine the size of the selection. They evaluated the effect of removing from the retrieved set duplicate sentences, caused by sentence being retrieved by multiple queries, and found it was not leading to significant performance changes. Lu et al. (2007) also used Lemur to select sentences from the bilingual corpus based on the test set. Giving more weight to duplicate sentences resulted with somewhat better results in their experiments. In our experiments, we remove duplicate sentences in order to keep the model smaller. Their results, with the top-1000 sentences for each query, improved somewhat over the baseline that used the entire bilingual corpus, while reducing the phrase-table size by almost 30%. IR was also used for related domain adaptation tasks. For instance, Chen et al. (2012) used the test set to generate an adapted tuning set, and in (Afli et al., 2012, 2013) retrieval is used to expand the training data from comparable, rather than parallel, corpora.

While methods based on information-theory metrics aim to find sentences that are similar to the development set (with no real guarantee that it will be similar to the test set), IR-based approaches can achieve tighter adaptation by specifically considering the test set. Yet, as mentioned, it is not always plausible to assume that the text to translate is available in advance. When adaptation for the test set is not possible, we risk encountering at translation time unseen or infrequent \(n\)-grams that do occur in the parallel corpus but were not selected.

2.3 Corpus coverage
Gascó et al. (2012) addressed the issue of lacking coverage by expanding the training data with more sentences that contain its infrequent source \(n\)-grams, and showed how to reduce the selec-

\(^{1}\)http://www.lemurproject.org/
tion to the test set \( n \)-grams. Lewis and Etemadi (2013) suggested a related approach, termed Vocabulary Saturation Filter (VSF), that filters the training data to contain only a certain, limited number of each source and target \( n \)-gram. Keeping track of the \( n \)-gram occurrences, they pass over the training data and add each sentence pair to the selected set only if the pair contains \( n \)-grams that occur less than the predefined threshold. This method ensures that every possible \( n \)-gram in the pool will be represented in the selected training data. It does not, however, make use of any in-domain data as its motivation is to reduce the model size, rather than domain adaptation. In Section 6 we show a simple way to adapt this method for this task.

3 Experimental Setting

Our experiments were conducted within the setting of the TRAD project, on the Arabic-French language pair. Below we describe the datasets, corpora and the setting of our experiments.

Evaluation datasets Two different datasets were evaluated, and for each a development set was provided, consisting of approximately 10,000 source tokens (of untokenized text). This is the only in-domain data we had at our disposal. The datasets are described below, and their size details are given in Table 1, in terms of source and target tokens, as well as in terms of the number of (bi-)sentences. These datasets are different by nature; this is reflected, for example, by the length of the sentences, evident from the table.

- **BLOGS**: Web blogs.
- **EGYP**: Transcription of conversations in Egyptian dialectal Arabic.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Sentences</th>
<th>Tokens-Ar</th>
<th>Tokens-Fr</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOGS</td>
<td>398</td>
<td>10K</td>
<td>15K</td>
</tr>
<tr>
<td>EGYP</td>
<td>941</td>
<td>10K</td>
<td>13K</td>
</tr>
</tbody>
</table>

Table 1: Development sets. This is the only in-domain data we have at our disposal.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Sentences</th>
<th>Tokens-Ar</th>
<th># of refs</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOGS</td>
<td>409</td>
<td>10K</td>
<td>4</td>
</tr>
<tr>
<td>EGYP</td>
<td>828</td>
<td>10K</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 2: Test sets.

Evaluation metric We evaluate our translations with BLEU (Papineni et al., 2002) using an official script of the evaluation campaign that removes punctuations and lowercases the detokenized output.

Corpora The list of bilingual corpora we used is given below. Table 3 shows the size of each corpora. As our pool of bi-sentences we use a concatenation of these corpora according to the listed order. This corpus, of 14.5M bi-sentences, was also used to produce baseline models in our experiments.

- **NEWS**: News commentary.
- **WIT3** (Cettolo et al., 2012): Transcribed and translated TED talks.\(^4\)

\(^2\)http://www.trad-campaign.org/
\(^3\)Within the TRAD project, this dataset is referred to as ‘H5’. This is the only dataset that is not publicly available.
\(^4\)Downloaded from https://wit3.fbk.eu/mt.php?release=2012-02
• **TRANSCRIPTS** (H4): Radio and television transcripts; standard Arabic.
• **MULTI-UN**: United Nations official documents.\(^5\)
• **OPENSUBTITLES** Tiedemann (2012): Movie subtitles.\(^6\)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Sentences</th>
<th>Tokens-Ar</th>
<th>Tokens-Fr</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEWS</td>
<td>91K</td>
<td>2.2M</td>
<td>2.4M</td>
</tr>
<tr>
<td>Wtr3</td>
<td>87K</td>
<td>1.9M</td>
<td>2.4M</td>
</tr>
<tr>
<td>TRANSCRIPTS</td>
<td>21K</td>
<td>561K</td>
<td>778K</td>
</tr>
<tr>
<td>MULTI-UN</td>
<td>9.9M</td>
<td>222.4M</td>
<td>285.5M</td>
</tr>
<tr>
<td>OPENSUBTITLES</td>
<td>4.4M</td>
<td>27.7M</td>
<td>32.3M</td>
</tr>
</tbody>
</table>

Table 3: Bilingual corpora for training.

**SMT system and preprocessing** We used Moses (Koehn et al., 2007) as our phrase-based SMT system. **IRSTLM** (Federico et al., 2008) was used to train 5-gram language models over the target side of the (selected) bilingual corpora. Arabic tokenization was done similarly to MADA-TOKAN (Habash et al., 2009), but due to project constraints we used a re-implementation of this tokenizer. More precisely, 300M Arabic words of the **MULTI-UN** corpus, segmented using MADA, were used to train a tokenizer using OpenNLP.\(^7\) To improve tokenization of punctuations, we applied the Moses tokenizer after the Arabic segmentation was applied. The translation models are trained on lowercased tokenized text and we apply a standard detokenizer prior to evaluation.

**IR system** For indexing and retrieval, we used Lucene,\(^8\) with its default settings. We indexed all unigrams and bigrams of the preprocessed Arabic side of the bilingual concatenated corpus, and used the inverted index for retrieval for both datasets. The preprocessing of the indexed corpus is identical to that applied to any other source data, such as the development or test sets.

4 **Perplexity-based adaptation**

First, we assess domain adaptation techniques based on information theory metrics. The concatenated corpora is our pool, \(G\), and each small development set, \(D\), is used both for tuning and for representing the in-domain data \((I)\). Ideally, these tasks would use different sets, but we had no additional in-domain data available, and the data we did have was too small to split.

4.1 **Perplexity (PP)**

We train a LM using the development set, \(D\), and compute its perplexity for each sentence in \(G\). We refrain from building an incremental language model over \(G\) and measuring its perplexity over \(D\), but rather use the more intuitive parameter \(m\) to determine in advance how many bi-sentences we wish to use for generating the model.

We experimented with selection using perplexity over the source (denoted \(pp\text{-}src\)), over the target (\(pp\text{-}tgt\)), and over both (\(pp\text{-}bi\)), where the sum of perplexities of a bi-sentence is used. Once scored, the \(m\) bi-sentences with the lowest perplexity scores are selected. Table 4 shows an example of these experiments, when using the BLOGS dataset and selecting from the **MULTI-UN** corpus to generate the SMT model. As seen in the table, DS based on both source and target

\(^{5}\)http://opus.lingfil.uu.se/MultiUN.php

\(^{6}\)http://www.opensubtitles.org; the corpus was downloaded from http://opus.lingfil.uu.se/OpenSubtitles2012.php

\(^{7}\)https://opennlp.apache.org/

\(^{8}\)http://lucene.apache.org/
proves most useful, and selection based on the source alone yields particularly inferior results. This outcome was consistent over additional values of $m$, over different pools of bilingual data and over the EGYP dataset as well.

---

**Table 4:** Perplexity-based DS results, when selecting based on the source (src), the target (tgt), or both (bi). $m$ denotes the number of selected sentence-pairs. Results are computed on the BLOGS development set.

<table>
<thead>
<tr>
<th>BLOGS</th>
<th>$m$</th>
<th>Selection</th>
<th>BLEU$_{dev}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>100K</td>
<td>pp-src</td>
<td>10.56</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pp-tgt</td>
<td>17.18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pp-bi</td>
<td><strong>17.85</strong></td>
</tr>
<tr>
<td></td>
<td>300K</td>
<td>pp-src</td>
<td>16.82</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pp-tgt</td>
<td>18.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pp-bi</td>
<td><strong>21.15</strong></td>
</tr>
</tbody>
</table>

---

4.2 Cross-entropy difference (CED)

We assessed the methods suggested by Moore and Lewis (2010) and Axelrod et al. (2011) for monolingual or bilingual CED. As above, we used the development set, $D$, as our in-domain data; for the general data, we used a random sample of $G$ of the same size as $D$. Following the PP results, we assessed selection using CED based on target alone and based on both source and target, but not over the source alone. We further assessed the option to select data for the TM bilingually but for the LM monolingually, using only the target side of the corpus. A subset of our results over the EGYP dataset, using perplexity and CED, is shown in Table 5.

---

**Table 5:** Perplexity and CED results for the EGYP dataset. Results are computed over the development set. The shaded row shows the baseline, where all bilingual data is concatenated and no selection is applied.

<table>
<thead>
<tr>
<th>EGYP</th>
<th>$m$</th>
<th>TM</th>
<th>LM</th>
<th>BLEU$_{dev}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1M</td>
<td>pp-bi</td>
<td>pp-bi</td>
<td>10.60</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ced-bi</td>
<td>ced-bi</td>
<td>11.18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ced-bi</td>
<td>ced-bi</td>
<td>11.36</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ced-bi</td>
<td>ced-bi</td>
<td><strong>11.75</strong></td>
</tr>
<tr>
<td></td>
<td>2M</td>
<td>pp-bi</td>
<td>pp-bi</td>
<td>11.18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ced-bi</td>
<td>ced-bi</td>
<td>11.59</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ced-bi</td>
<td>ced-bi</td>
<td>12.06</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ced-bi</td>
<td>ced-bi</td>
<td><strong>12.25</strong></td>
</tr>
<tr>
<td></td>
<td>3M</td>
<td>pp-bi</td>
<td>pp-bi</td>
<td>10.85</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ced-bi</td>
<td>ced-bi</td>
<td>11.91</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ced-bi</td>
<td>ced-bi</td>
<td>11.99</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ced-bi</td>
<td>ced-bi</td>
<td><strong>12.03</strong></td>
</tr>
<tr>
<td></td>
<td>14.5M</td>
<td>-</td>
<td>-</td>
<td>10.63</td>
</tr>
</tbody>
</table>

---

From Table 5 we learn that for the EGYP dataset: (i) adapting the LM based on the target only is better than bilingual adaptation, and (ii) CED outperforms PP over any selection size. In contrast, as shown in Table 6, for BLOGS, PP is the preferred choice. Another differ-
ence between the datasets is that BLOGS requires a larger amount of selected data to reach the performance of the simple baseline. This property of the datasets surfaced all along our experiments, and indicate that the BLOGS dataset is closer to the general corpus than EGYP. Different selection strategies had to be followed.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
\textbf{BLOGS} & \textbf{Selection} & \textbf{BLEU$_{dev}$} \\
\hline
100K & ced-bi & 15.95 \\
      & pp-bi  & 17.85 \\
\hline
1M   & ced-bi & 23.17 \\
      & pp-bi  & 24.92 \\
\hline
2M   & ced-bi & 24.39 \\
      & pp-bi  & 25.39 \\
\hline
3M   & pp-bi  & 25.80 \\
14.5 & -      & 26.04 \\
\hline
\end{tabular}
\caption{Perplexity and CED results for the BLOGS dataset, over the development set. The shaded row shows the baseline, where no selection is applied.}
\end{table}

For subsequent experiments, we chose the 2M selection size which obtained good results in our experiments while remaining reasonable in terms of model size.\footnote{We further expand over this model in subsequent experiments, and we must therefore bound it, even for BLOGS.} With respect to the selection metric, we use the best-performing one for each dataset: ced (ced-bi for TM and ced-tgt for LM) for EGYP, and pp-bi for BLOGS.

\section{IR over an adapted model}

Data-selection methods based on information retrieval were described in Section 2. The principle is to generate a tightly adapted model for the (source side of the) test set by obtaining parallel corpora that covers its $n$-grams. We implement this approach, following prior work, with some modifications. Yet, realizing that in real-world scenarios, using the test set is not always feasible, our goal is to provide a model that can also support real-time response. In this section we describe a method that achieves this goal and the experiments using it over the two datasets. The idea is to enable working in both immediate and delayed modes. To that end, we create a model that is IR-ready, but still IR-independent. That is, a model that would be relatively easy and fast to update when we receive a text to translate and can afford a short delay, but that can also perform well when an immediate translation is required.

To support IR-based adaptation we create an inverted index of the entire preprocessed source side of the bilingual corpus. All $n$-grams are indexed, up to a maximal predefined length (2, in our experiments). Then, we train an adapted model based on CED or PP, using some in-domain data as described in Section 4. Since this model will not be used as our final one, and we are only after its tables (phrase table, language model and reordering table), it does not need to be tuned. Next, we let another part of the in-domain data play the role of the test set. We extract $n$-grams from its source side, optionally ordering them (see below). We keep track of the number of occurrences of each $n$-gram in the retrieved set that is being collected, $\mathcal{R}$. Searching up to $k$ instances of each $n$-gram, we add to $\mathcal{R}$ the source sentences retrieved by searching exact matches of the $n$-gram, and their corresponding target sentences, and update the counts of each $n$-gram that appears in them. When the next $n$-gram is up for search, we deduct its current count in $\mathcal{R}$ from the maximum number of requested hits. The motivation is to obtain enough instances of each $n$-gram while keeping the retrieved dataset small for space and
speed considerations. Note that this does not constrain the absolute number of occurrences of each $n$-gram, and in general, more frequent $n$-grams will tend be more frequent in $R$. With the motivation of keeping the model compact, and since the results of retrieval may overlap when separate queries lead to identical retrieved sentences, we perform a bilingual de-duplication. That is, we remove all identical bi-sentences from the selected parallel corpus, allowing duplicates to remain in either source or target, but not both. The de-duplicated set, $R'$, is then used to construct additional translation and language models. The two types of adaptation are then combined in a single log linear setting, where each makes up one TM or LM. That is, we add an additional TM or LM model based on the IR data rather than train models with the entire selected data. Such separation was shown to be helpful by, e.g., Axelrod et al. (2011), and enables quickly generating the models (Mirkin and Cancedda, 2013). The updated configuration, with 2 TMs and 2 LMs, is then tuned using the development set $D$, producing a ready-to-use model. If test-set adaptation is possible, we apply the same IR selection over the text for translation, and use the TM and the LM generated with it to substitute those created with the previously-available in-domain data. If we can afford, time-wise, to re-tune the model, that would be the preferred choice. Yet, tuning is a lengthy process, and if the models are of similar properties, tuning may be skipped, as shown in (Mirkin and Cancedda, 2013).

The above steps are summarized in Algorithm 1. As mentioned, only a small in-domain dataset was provided for us in these experiments. Due to this constraint, the same dataset was used for multiple roles: as a development (tuning) set, and as the seed dataset, guiding the PP and IR adaptations. If more in-domain data is available, these tasks should be performed using different sets, in order to avoid overfitting. In Algorithm 1, we refer to the in-domain data as $D$, regardless of its role, as was actually done in our experiments.

**Algorithm 1**: IR over an adapted model

**Input**: Bilingual pool, $G$; in-domain data, $D$; optionally: text to translate, $T_{src}$

**Output**: An adapted model

- Index the source side of $G$
- Generate a PP-adapted model, using $D$ // No tuning required
- Train IR-based models for $D$:
  - $R_D = \{\}$
  - Extract $n$-grams from $D_{src}$; order them
  - For each $n$-gram, $w$:
    - Search for its instances
    - Update counts for all $n$-grams in the retrieved set, $R(w)$
    - $R_D = R_D \cup R(w)$
  - Bilingually de-duplicate $R_D$, producing $R'_D$
  - Train TM and LM from $R'_D$
- Add the new models as TM & LM in a log-linear configuration with the PP-adapted ones
- Tune the combined model

If given $T_{src}$:

- Train IR models from $R_T$ or $R_T \cup R_D$ // as done for $D$
- Replace the TM and LM in the tuned model

### 5.1 IR experiments

To run this method, like most methods using domain adaptation with IR, one must determine the values of a set of parameters. Below we list these parameters and describe the outcomes of exploring their usage over the development sets. Here, only the retrieved set is used to construct the SMT model, without the PP-adapted one, in order to let the different parameter values be better reflected in the results.
- \( k \), the number of requested search hits: A higher value results in more training data, at the cost of increasing model size. We experimented with \( k \) values of 100, 500 and 1000, where \( k = 1000 \) yielded the best results.

- \( n \), the maximal number of tokens in the queried \( n \)-gram: A search for a unigram matches all longer \( n \)-grams that contain it. However, since we limit the number of hits per query, it is not always the case in practice. Longer \( n \)-grams potentially constitute a better match to the query, and may therefore be valuable. Still, our experiments showed that using \( n = 1 \) is usually sufficient, possibly since the \( k \) value we used was able to obtain enough matches.

- The IR similarity metric: we used the default metric used by Lucene: a \( tf-idf \) weighted cosine similarity between the query and the document (a sentence, in our case).

- The order of searched \( n \)-grams: Since the retrieved set is sensitive to the order of the search, we assessed several ordering techniques:
  
  (i) “As-is”: the original order of \( n \)-grams in the searched set.

  (ii) Decreasing frequency: searching for the more frequent \( n \)-grams first.

  (iii) Ratio of frequencies in the search set (e.g. mathcalD, representing in-domain data) and the entire pool. The idea is to give priority to words that are more prominent in the in-domain set in comparison to their “regular” prominence, much like CED. Since the number of occurrences in the entire corpus may be very high, we take its squared root, and add 1 to it, to avoid division by 0: \( r = \frac{freq_D(w)}{\sqrt{freq_G(w)+1}} \), where \( w \) is the \( n \)-gram under consideration.

  In this set of experiments, ordering based on the last option showed but a slight improvement over the default, as-is, order. Hence, we did not further reorder the \( n \)-grams prior to the search.

Another outcome of exploring the IR parameters over the development sets was that the baseline, using the entire pool, beats the IR-based selection for BLOGS (26.04 vs. 24.80 with \( k = 1000 \)), while for EGYP, a very small retrieved set obtains as good results as the entire pool (10.65 vs. a baseline of 10.63).

So far we have selected the non-IR adaptation method and the IR parameters. We now turn to assess the proposed combined model. We compare using different sets in the retrieval table – the development set, the test set, or both concatenated. In the latter case, we add the already-obtained \( R_D \) to \( R_T \) since it provides more statistics for generating the TM and LM, without requiring additional on-the-fly retrieval. Processing the larger dataset indeed takes longer, but is still relatively limited in comparison to the entire size of the data used in the model. Each experiment using the test set used the tuning of the development set configuration with the same parameters, only replacing the phrase table and LMs.

The results are shown in Tables 7 and 8. First, the tables show that for both datasets, even when using only the development set (the \( R_D \) parts of the tables), the proposed IR model improves over PP- or CED-adapted models of a similar size. In the case of EGYP, it improves over any other model we have managed to generate.\(^{10}\) Second, unsurprisingly, using the test set improves translation performance. The improvement is more significant for BLOGS than EGYP. Checking the reduction of OOVs when using the test sets, we learned that the test set significantly reduced the number of output sentences with OOVs for BLOGS (by 36%, for

\(^{10}\)We note that all results for this dataset of other participants in the evaluation campaign were significantly lower than the ones presented here.
Table 7: EGYP results over the test set using our proposed IR method over a ced-adapted model of 2M bi-sentences. This is the model found to perform best over the development set, where LM is adapted based on the target only (see Section 4). Its result is shown in the shaded line, as well as ced with 3M bi-sentences and a baseline using all of $G$.

<table>
<thead>
<tr>
<th>IR &amp; $k$</th>
<th>Sentences</th>
<th>BLEU$_{test}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ced</td>
<td>-</td>
<td>2M</td>
</tr>
<tr>
<td>ced</td>
<td>-</td>
<td>3M</td>
</tr>
<tr>
<td>no selection</td>
<td>-</td>
<td>14.5M</td>
</tr>
<tr>
<td>$\mathcal{R}_D$</td>
<td>500</td>
<td>2.27M</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>2.49M</td>
</tr>
<tr>
<td>$\mathcal{R}_T$</td>
<td>500</td>
<td>2.26M</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>2.47M</td>
</tr>
<tr>
<td>$\mathcal{R}_T \cup \mathcal{R}_D$</td>
<td>500</td>
<td>2.43M</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>2.63M</td>
</tr>
</tbody>
</table>

Table 8: BLOGS results over the test set for applying IR over a pp-bi adapted model of 2M bi-sentences. For comparison, other shaded lines show pp-bi with 3M sentences and the baseline using all of $G$.

<table>
<thead>
<tr>
<th>IR &amp; $k$</th>
<th>Sentences</th>
<th>BLEU$_{test}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>pp-bi</td>
<td>-</td>
<td>2M</td>
</tr>
<tr>
<td>pp-bi</td>
<td>-</td>
<td>3M</td>
</tr>
<tr>
<td>no selection</td>
<td>-</td>
<td>14.5M</td>
</tr>
<tr>
<td>$\mathcal{R}_D$</td>
<td>500</td>
<td>2.59M</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>3.06M</td>
</tr>
<tr>
<td>$\mathcal{R}_T$</td>
<td>500</td>
<td>2.66M</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>3.16M</td>
</tr>
<tr>
<td>$\mathcal{R}_T \cup \mathcal{R}_D$</td>
<td>500</td>
<td>3.11M</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>3.93M</td>
</tr>
</tbody>
</table>

$k = 1000$); for EGYP the reduction was more modest (17%). As seen in our additional results, this dataset is very different from the available bilingual corpus and much of its vocabulary does not occur in the pool at all. Adding the development set to the test set was not found helpful for EGYP, but was so for BLOGS, where, generally speaking, improvement was observed with any addition of data.

6 Adapted Vocabulary Saturation Filter (AVSF)

The proposed IR method overcomes to some extent the “overfitting” problem of PP-based techniques. Still – when test-set adaptation is not possible – it considers the development set as its core source for adaptation, and is therefore prone to have limited coverage of the actually necessary $n$-grams.

VSF, an algorithm suggested by Lewis and Eetemadi (2013), aims to reduce the training data by including each $n$-grams only a certain number of times. Thus, any $n$-gram, up to the determined length, that appears in the training data, also appears in its compact version. We find this algorithm suitable to compensate for drawbacks of adaptation with PP and of IR without using the test set. VSF was not designed for domain adaptation and indeed, does not make use
of any domain-specific data, such as the development set. We propose an adapted version of VSF, denoted AVSF, to be used for domain adaptation.

The order of the data provided to VSF has a direct effect on the selected training data. Lewis and Eetemadi (2013) discussed this issue, and ordered the training data by alignment score. They were not addressing domain adaptation in that work, and the adaptation for our task is simple: first we sort the training data according to a perplexity-based metric and then apply VSF over the reordered corpus. The idea is that sentences more relevant for the domain will be selected first and will be less likely to be skipped due to \( n \)-gram saturation.

Table 9 shows experiments we conducted using this algorithm, in comparison to its non-adapted version. VSF has 2 parameters: \( n \), the maximal length of the \( n \)-grams we are trying to cover, and \( t \), the minimum required frequency of each \( n \)-gram. Experiments over the development sets, with smaller corpora, showed that while increasing \( n \) is useful, \( t \) does not make much difference; we therefore set \( t \) to 1 in all our experiments. Applying VSF to our (arbitrarily ordered) bilingual corpus, with \( n = 2 \) and \( t = 1 \) results with a large amount of selected data, 6.9M bi-sentences. We consider this size as contradicting one of our goals in this work – to keep the model small – and we therefore do not train a model using this selection. Yet, for AVSF, where the order is based on domain adaptation techniques, we can use \( n = 2 \) by applying the algorithm only over the top-\( m \) sentences of the ordered training data. This results in a smaller selected set, while guaranteeing a coverage of the more relevant part of the corpus \( n \)-grams.

**Table 9: AVSF results for EGYPT, when VSF is applied over the top \( m \) bi-sentences of the CED ordered training set. \( m = 14.5M \) refers to the entire bilingual corpus. In all experiments, \( t = 1 \).**

<table>
<thead>
<tr>
<th>( n )</th>
<th>Sent. order</th>
<th>Sentences</th>
<th>BLEU_{test}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>14.5M</td>
<td>11.20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VSF</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>841K</td>
<td>8.76</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>6.9M</td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AVSF</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ced-bi (14.5M)</td>
<td>925K</td>
<td>10.29</td>
</tr>
<tr>
<td>2</td>
<td>ced-bi (3M)</td>
<td>348K</td>
<td>10.78</td>
</tr>
</tbody>
</table>

| 1      | pp-bi (3M)   | 1.63M    | 13.44       |
| 2      | pp-bi (4M)   | 2.25M    | 13.81       |

| 2      | pp-bi (3M)   | 1.77M    | 30.61       |
| 2      | pp-bi (4M)   | 2.34M    | 31.86       |

The results show that increasing the size of the training data, within the size limitations we imposed, is helpful. Selection based on top-\( m \) is also beneficial, and provides more flexibility with respect to the VSF parameters. AVSF therefore proves very useful, and is also efficient in terms of run time, since once the corpus has been ordered according to PP scores, as done in
Table 11: Summary of prominent results of each assessed method. The \( \parallel \) symbol denotes a separation of translation or language models. The first model listed is the one that determines the reordering table. For IR we also show the results when the test set was available for use.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>Parameters</th>
<th>Sentences</th>
<th>BLEU</th>
<th>BLEU with ( \bar{T}_{src} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>EGYP</td>
<td>Baseline</td>
<td>-</td>
<td>14.5M</td>
<td>11.20</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>PP</td>
<td>( ced-bi ) (LM), ( ced-tgt ) (LM), 2M</td>
<td>2M</td>
<td>13.18</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>IR</td>
<td>( ced ) (2M) ( \parallel ) ( IR_{2b}, k = 1000, n = 1 )</td>
<td>2.49M</td>
<td>13.50</td>
<td>13.97</td>
</tr>
<tr>
<td></td>
<td>AVSF</td>
<td>( n = 2, t = 1, ced ) (4M)</td>
<td>2.25M</td>
<td>13.81</td>
<td>-</td>
</tr>
<tr>
<td>BLOGS</td>
<td>Baseline</td>
<td>-</td>
<td>14.5M</td>
<td>35.49</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>PP</td>
<td>( pp-bi ), 3M</td>
<td>3M</td>
<td>29.11</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>IR</td>
<td>( pp-bi ) ( \parallel ) ( IR_{2b}, k = 1000, n = 1 )</td>
<td>3.06M</td>
<td>31.18</td>
<td>33.04</td>
</tr>
<tr>
<td></td>
<td>AVSF</td>
<td>( n = 2, t = 1, pp-bi ) (4M)</td>
<td>2.34M</td>
<td>31.86</td>
<td>-</td>
</tr>
</tbody>
</table>

7 Summary of results and conclusions

In Table 11 we summarize the prominent results obtained for each assessed method. The results show that AVSF and the proposed IR configuration outperform the state of the art selection with CED, even with smaller model sizes. AVSF seems to be doing better in that respect, while the IR model has the advantage of being able to tightly adapt to the test set, when some delay is permitted.

In conclusion, in this work we have investigated multiple data selection methods for domain adaptation in SMT. The generated model size played an important role in our research as we tried to achieve the best possible results with models that are trained on no more than 30% of the initial bilingual corpus. Considering the different characteristics of selection techniques, we proposed two separate methods that combine existing methods in order to benefit from the advantages of each one of them. Our extension to well-known IR-based adaptation proved competitive and enables supporting two modes of operation: instant and delayed translations. Our proposed adaptation of VSF to the task at hand was demonstrated to be useful in obtaining good performance through small models. An immediate extension of these results would be to apply the IR method over the AVSF adapted model.

Assessing two distinct datasets, we learned that DS methods are not always consistent in their success over different datasets. This is naturally reflected in the parameter values, but also in the method that needs to be used. In that respect, our experiments consistently showed that when the dataset is more “similar” to the pool (as reflected, e.g. in the BLEU scores), using more data is useful and PP is preferable over CED. Our experiments revealed some hints of how to anticipate that, but further research is required to be able to predict the most effective method for the domain and the range of parameters to assess, in order to reduce the search space. Achieving that can potentially cut the adaptation effort considerably.

Acknowledgements

This work was conducted within the framework of the TRAD project, funded by the French Government, Direction générale de l’armement (DGA).
References


Pivot-based Triangulation for Low-Resource Languages

Rohit Dholakia
rdholaki@cs.sfu.ca
Anoop Sarkar
anoop@cs.sfu.ca
School of Computing Science,
Simon Fraser University,
Burnaby, V5A 1S6, Canada

Abstract

This paper conducts a comprehensive study on the use of triangulation for four very low-resource languages: Mawukakan and Maninkakan, Haitian Kreyol and Malagasy. To the best of our knowledge, ours is the first effective translation system for the first two of these languages. We improve translation quality by adding data using pivot languages and experimentally compare previously proposed triangulation design options. Furthermore, since the low-resource language pair and pivot language pair data typically come from very different domains, we use insights from domain adaptation to tune the weighted mixture of direct and pivot based phrase pairs to improve translation quality.

1 Introduction

Triangulation for phrase-based statistical machine translation (SMT) [Utiyama and Isahara, 2007, Cohn and Lapata, 2007, Wu and Wang, 2007] refers to the use of a pivot language when translating from a source language to a target language. Previous research into triangulation for machine translation either used Europarl [Cohn and Lapata, 2007, Utiyama and Isahara, 2007, Huck and Ney, 2012] or languages with large corpora in same domain [Gispert and Mario, 2006] or assumed presence of languages that are very closely related [Nakov and Ng, 2012, Wang et al., 2012]. However, low resource languages are quite different when it comes to the kind and size of parallel data that is available. This paper considers machine translation into English from four diverse low-resource languages: Mawukakan and Maninkakan, which are West African languages; Haitian Kreyol, in the domain of short messages sent in the aftermath of the Haiti earthquake in 2010; and Malagasy, an Austronesian language from Madagascar. This is the first comprehensive study of triangulation for these four languages, and to our best knowledge, Mawukakan and Maninkakan have not been studied before in the SMT literature.
Faced with a low-resource language pair, several questions arise when trying to use the approach of triangulation:

- [Utiyama and Isahara, 2007] use a different way of computing lexical scores from [Cohn and Lapata, 2007]. Which one is better suited for triangulation in a resource-poor scenario?
- In [Utiyama and Isahara, 2007, Cohn and Lapata, 2007, Wang et al., 2012, Wu and Wang, 2007] many different feature functions are provided for the log-linear model over triangulated phrase pairs. We conduct extensive experiments to show which features should be used for real world low-resource languages based on the data settings for each language pair.
- In [Cohn and Lapata, 2007] a mixture of the direct system and the triangulated system is shown to work better. However, they used uniform weights. In [Wang et al., 2012] a few different weights were selected heuristically while in [Wu and Wang, 2007] 0.9 is assumed for the baseline. We provide an algorithm that combines grid search for learning the mixture weights and minimum error rate training of the direct and triangulated log-linear models.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Direct</th>
<th>Src-Pivot</th>
<th>Pivot-Tgt</th>
<th>Domains</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Utiyama and Isahara, 2007]</td>
<td>560K</td>
<td>560K</td>
<td>560K</td>
<td>Multi-Parallel</td>
</tr>
<tr>
<td>[Cohn and Lapata, 2007]</td>
<td>700K</td>
<td>700K</td>
<td>700K</td>
<td>Multi-Parallel</td>
</tr>
<tr>
<td>[Cohn and Lapata, 2007]</td>
<td>10K</td>
<td>10K</td>
<td>10K</td>
<td>Multi-Parallel</td>
</tr>
<tr>
<td>Mawukakan</td>
<td>3K</td>
<td>3K</td>
<td>2M</td>
<td>Different</td>
</tr>
<tr>
<td>Maninkakan</td>
<td>4K</td>
<td>4K</td>
<td>2M</td>
<td>Different</td>
</tr>
<tr>
<td>Haitian Kreyol</td>
<td>120K</td>
<td>30K</td>
<td>2M</td>
<td>Different</td>
</tr>
<tr>
<td>Malagasy</td>
<td>88K</td>
<td>30K</td>
<td>2M</td>
<td>Different</td>
</tr>
</tbody>
</table>

Table 1: Comparison of our data settings (last four rows) with previous work. Haitian Kreyol data are short messages sent after earthquake. Malagasy data is automatically aligned news articles in Malagasy). For these two languages we use the Bible as our source-pivot bitext as they have no parallel data source with French, our pivot language. Mawukakan and Mawukakan have a very small source-pivot and source-target bi-texts, but the source-pivot corpus has common sentences with the source-target corpus. We use French as the pivot language to keep the same experimental setting for all our source languages.

To answer some of the above questions, we study the effectiveness of pivot-based triangulation for languages with insufficient resources, Mawukakan, Maninkakan, Malagasy and Haitian Kreyol. Table 1 compares our data settings with previous research into triangulation. Note that we are using all the available data for each language pair. In most cases there was only one possible choice for each source-pivot or pivot-target parallel corpus. Mawukakan and Maninkakan are two languages from the Mandekan family, spoken by almost 3.5 million people in West Africa. The Mandekan languages are a part of the Niger-Congo language family. Maninkakan and Mawukakan have little writing tradition, are written using multiple alphabets\(^1\) and have very little

\(^1\)The data we have used has Latin script, obtained via LDC. See Table 5.
resources for machine translation. Malagasy is the national language of Madagascar, spoken by 18 million people worldwide. Haitian Kreyol is the national language of the Republic of Haiti and data used is from the Sixth Workshop on Machine Translation, 2011 [Callison-Burch et al., 2011]. It comprises short messages sent to the number 4636 after the devastating earthquake in January, 2010. Although nine systems participated in the workshop on Haitian Kreyol, the approach of triangulation was not used.

In the aftermath of the earthquake in Haiti in January, 2010, Mission 4636 set up a service where anyone in Haiti could send a message for free to a phone number 46362. A group of volunteers translated the messages into English and helped the relief organizations provide swift help to the affected masses. Microsoft Research released a translation system to the public, for Haitian Creole, 5 days after the devastating earthquake [Lewis et al., 2011]. The fast turnaround time3 and the usefulness of machine translation in the time of crisis inspired the featured task in the 6th Workshop on Statistical Machine Translation.

Malagasy is an Austronesian language and the national language of Madagascar, spoken by 18 million around the world. Although it shares several words with Ma’anyan, it has influences from Arabic, French, Swahili and Bantu. Characters can have diacritics but not always. Numbers are written right-to-left like Arabic, while some words are in common with French. It follows the Latin alphabet but with 21 characters. Finally, the dataset we have is real-world news articles translated by volunteers across the world4 and aligned using a sentence aligner, thus, introducing some inconsistencies.

---

2http://www.mission4636.org
3To know the exact timeline, refer to http://languagelog.ldc.upenn.edu/nll/?p=2068
4http://www.ark.cs.cmu.edu/global-voices/
2 Related Work

Consider a source language $s$, pivot language $p$ and target language $t$. When using the cascading approach, we build two systems, between $s$ and $p$ and between $p$ and $t$. In this paper, we do not discuss the approach of cascading, which would translate sentences in $s$ to $p$ and use the n-best list to translate the sentence into $t$. It was shown previously [Utiyama and Isahara, 2007, Gispert and Mario, 2006] that cascading was not the best approach.

The second approach is the pivot-based approach where a triangulated phrase table is generated between the source and target, by using the common pivot phrases between the source pivot and pivot target tables [Utiyama and Isahara, 2007, Cohn and Lapata, 2007, Wu and Wang, 2007]. [Utiyama and Isahara, 2007] observed that the triangulated table was able to achieve comparable BLEU scores to the direct system for French, German and Spanish. This could be owing to the fact that the data comprised multi-parallel 560K sentences. [Cohn and Lapata, 2007] observe that multiple pivot languages lead to more fluent translations compared to one pivot language. Multiple pivot language lead to multiple alternative translations, thus, increasing phrase coverage and rewarding the more appropriate translations and reducing out-of-vocabulary words further. They also propose a systematic way of combining the triangulated translation model with the direct model using linear interpolation and log-linear interpolation, although they assume the same weight for all the models. To “simulate” a low-resource scenario, the top 10K multi-parallel sentences are considered for source pivot, pivot target and source target systems. [Sennrich, 2012] compared various methods of using linear interpolation in a domain adaptation setting. [Wu and Wang, 2007] also approach triangulation in a similar way to [Cohn and Lapata, 2007] but use different methods to compute lexical weights. [Nakov and Ng, 2012] propose a language-independent approach to improving translation for low-resource languages, but the approach assumes the presence of a resource-rich language that bears similarity to the low-resource language, the similarities helping in creating a large triangulated phrase table. In [Wang et al., 2012], the resource-rich language is adapted to be more like the resource-poor one. Notice that this also assumes both are very similar. Results are reported using both Malay-Indonesian and Bulgarian-Macedonian, the third language being English in both cases. [Gispert and Mario, 2006] translate Catalan to Spanish via English by using news corpora on both source pivot and pivot target side. [Huck and Ney, 2012] report on BLEU score improvements by using $10^9$ parallel sentence between German and French.

[Kholy et al., 2013] observe that using categories for source target pairs when combining the direct and triangulated models helped in improving the BLEU score. In other words, a source target pair can be in both the direct and triangulated phrase tables, or only one of them could be in both. They enumerate the different possibilities and use them as separate decoding paths. [Zhu et al., 2013] try to address the problem of missing translations in triangulation (as pivot phrases are not always in
both tables) by using a random walk approach. The initial triangulated phrase table is extended by treating the table as a graph and using a random walk to obtain more translations. [Crego et al., 2010] focus on improving one system (German-English) by using a dynamically build model from auxiliary sources. In other words, they translate the source sentence using various models and then use a framework to combine the different outputs. [Bertoldi et al., 2008] suggested using alternative decoding paths when having different translation models. In our experiments, we found that alternative decoding paths did not work so well. This could be partly be because there are not that many alternatives when having two translation models of very different sizes and from different domains. When we do have alternative paths, they may not always be useful.

A common thread that binds the previous work using the approach of triangulation is the usage of resource-rich languages. The fundamental reason behind the effectiveness of triangulation is the reduction in the number of OOVs when using the pivot language(s). All the Europarl languages are based on parliamentary proceedings and have minimal noise. Hence, the improvements using triangulation over the direct systems cannot be generalized for systems for low-resource languages.

3 Design choices for Triangulation

Given a source language $s$, pivot language $p$ and target language $t$, pivot-based triangulation uses common pivot phrases between the source-pivot phrase table $p_{sp}$ and pivot-target $p_{pt}$ to generate a new phrase table between source and target. As the triangulated table is generated using common phrases, the feature values cannot be computed using the alignments and co-occurrence counts. We discuss two ways of computing phrase scores in section 3.1 and two ways of computing lexical scores 3.2. Following [Cohn and Lapata, 2007] we build a mixture model of the direct source-target system and the triangulated source-pivot-target system. In Section 3.4, we propose a new iterative method to find the mixture weights.

3.1 Phrase pair scores

3.1.1 Product Approach

[Utiyama and Isahara, 2007] computes feature values for the triangulation phrases by multiplying values from source-pivot and pivot-target phrase tables:

$$p_{ts}(t \mid s) = \sum_p p_{pt}(t \mid p)p_{sp}(p \mid s)$$

(1)

$$p_{ts}(s \mid t) = \sum_p p_{sp}(s \mid p)p_{pt}(p \mid t)$$

(2)
We are marginalizing over the pivot phrases \( p \), essentially making an independence assumption of the following form, as in [Cohn and Lapata, 2007]:

\[
p_w(t \mid s) = \sum_p p_w(t, p \mid s) = \sum_p p_w(t \mid p, s) p_w(p \mid s) \approx \sum_p p_w(t \mid p) p_w(p \mid s)
\]

### 3.1.2 Joint probability for triangulation scores

[Cohn and Lapata, 2007] propose using the joint probability \( p_{tr}(s, t) \) to calculate the triangulated phrase scores \( p_{tr}(t \mid s) \) and \( p_{tr}(s \mid t) \). Since we do not have observed counts in the triangulated phrase table, counting the pairs after triangulation will not be a true reflection of the joint probability.

The joint probability of a phrase pair looks as shown in Equation 3, which decomposes to Equation 4. Making an independence assumption, shown in Equation 5, we compute the joint probability of a triangulated phrase pair as shown in Equation 6.

\[
P(s, t) = \sum_p P(s, p, t) \quad (3)
\]

\[
P(s, p, t) = \sum_p P(p, t) P(s \mid p, t) \quad (4)
\]

\[
P(s \mid p, t) \approx P(s \mid p) \quad (5)
\]

\[
p_{tr}(s, t) = \sum_p p_{pt}(t)p_{pt}(p \mid t)p_{sp}(s \mid p) \quad (6)
\]

The counts for the direct system are used to compute the joint and the conditional distributions in this equation.

### 3.2 Lexical Scores

#### 3.2.1 Product Approach

Similar to phrase scores, we compute the triangulated lexical scores using the product of the lexical scores of the source-pivot and pivot-target tables.

\[
p_{lex_{tr}}(t \mid s) = \sum_p p_{lex_{pt}}(t \mid p)p_{lex_{sp}}(p \mid s) \quad (7)
\]

\[
p_{lex_{tr}}(s \mid t) = \sum_p p_{lex_{sp}}(s \mid p)p_{lex_{pt}}(p \mid t) \quad (8)
\]
3.2.2 IBM Model 1 Alignments

[Cohn and Lapata, 2007] propose an alternative way to compute the lexical score by using unsupervised alignments between source and target phrases in the triangulated phrase table. They use the IBM Model 1 [Brown et al., 1993] (Model 1, henceforth) score between the phrase pairs in the triangulated table. Treating the triangulated phrase table as a parallel corpus, we learn the Model 1 alignment scores in both directions using 5 iterations of the EM algorithm [Dempster et al., 1977]. Given a foreign sentence \( f = f_1, \ldots, f_m \), English sentence \( e = e_1, \ldots, e_l \), the IBM Model 1 score between the sentences is calculated as follows:

\[
p(f, a \mid e) = \frac{e}{(l + 1)^m} \prod_{j=1}^{m} t(f_j \mid e_{a(j)})
\]

(9)

Model 1 learns the likelihood of the alignment of the individual words, while also considering the fact that a triangulated table will have less number of source phrases translating into good and some noisy translations. Noisy translations will automatically get a lower Model 1 score, something less likely to happen when using the simpler approach of multiplying the lexical scores. This effect of noisy translations ending up as a viable translation during decoding is also because of the limited source-pivot training corpora available. Several translations have been only seen once and the phrase lengths are not very long either (85% of Mawu or Manin phrase table has \( \leq 3 \) words).

3.3 Connectivity Features:

The phrase pairs in the triangulated phrase table are contingent upon the common pivot phrases. As a result, we can have phrase pairs that map “!?” to a target phrase “and making the soup thick !?” in Haitian Kreyol to English triangulated phrase table. Due to the fan-out nature of triangulation, spurious phrase pairs like above get high enough feature values to end up as a translation during decoding. To reward phrase pairs that have more alignment links between and to penalize pairs that don’t, we add two connectivity features to the phrase table, as proposed in [Kholy and Habash, 2013] for Persian to Arabic translation using English as the pivot language. For a source phrase \( p_s \), target phrase \( p_t \), and with the number of alignment links between them \( N \), the strength feature is:

\[
\begin{align*}
\text{source strength} &= \frac{N}{S} \\
\text{target strength} &= \frac{N}{T}
\end{align*}
\]

where \( S \) is the length of the source phrase \( p_s \) and \( T \) is the length of the target phrase \( p_t \). To compute the connectivity strength feature, the alignments in the source-pivot phrase pair are intersected with the pivot-target phrase pair.
3.4 Translation Model Combination

[Cohn and Lapata, 2007] propose a mixture of the direct source-target system model $p_d$ with the triangulated source-pivot-target model $p_{tr}$:

$$p_{interp}(s \mid t) = \lambda_d p_d(s \mid t) + (1 - \lambda_d) p_{tr}(s \mid t) \quad (10)$$

In their data setting, setting $\lambda_d$ to 0.5 was a reasonable choice. [Nakov and Ng, 2012] try different heuristically selected values and re-learn the log-linear weights. However, both of these choices are unreasonable in our low-resource data setting because our datasets come from different domains (thus, using uniform weights would be unreasonable) and using weights that were set heuristically would not be a systematic search over the parameter space.

**Grid search for interpolation:** For Haitian Kreyol, we are trying to improve translations for real-world short messages using common phrases between Bible and parliamentary proceedings. For Malagasy, we are trying to do the same for news articles. To get the best of both worlds, we would want a $\lambda_d$ in equation (10) which maximizes our BLEU score, where $p_d$ represents the direct translation model while $p_t$ represents the triangulated translation model.

Algorithm 1: Grid Search for Interpolation

**Input:** triangulated phrase table $p_{tr}$,
direct phrase table $p_d$,
$\lambda_d$, $\lambda_{tr} = 1 - \lambda_d$, $\text{prev}\_\text{bleu} = 0$,
minimum = $e^{-2}$

**Output:** best $\lambda_d$

1: **while** $\delta_{\text{bleu}} > \text{minimum}$ **do**
2: interpolate $p_d$, $p_{tr}$ to give $p_{interp}$
3: Run MERT using $p_{interp}$ as translation model
4: find $\text{bleu}\_\text{heldout}$
5: $\delta_{\text{bleu}} = \text{bleu}\_\text{heldout} - \text{prev}\_\text{bleu}$
6: $\text{prev}\_\text{bleu} = \text{bleu}\_\text{heldout}$
7: Based on $\delta_{\text{bleu}}$, find new $\lambda_d$
8: **end while**

Algorithm 1 is an iterative method that learns the best $\lambda_d$ using a publicly available toolkit, CONDOR [Berghen and Bersini, 2005], a direct optimizer based on Powell’s algorithm, that does not require explicit gradient information for the objective function. The approach can be easily extended to multiple triangulated models or different co-efficients for each feature. Line 2 interpolates the two translation models using equation (10). We re-tune the log-linear weights using MERT for the interpolated feature values (on the same tuning data as the baseline) and use the tuned model to find BLEU score on the same heldout set. Based on the difference between the BLEU score obtained and the previous BLEU (Line 7), CONDOR searches for the new co-efficient in
Table 3: Different languages have different interpolation co-efficients that lead to the best system. Although we always start with 0.85, we iterate systematically over different values to obtain the best co-efficient.

<table>
<thead>
<tr>
<th>Language</th>
<th>Best $\lambda_d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mawukakan</td>
<td>0.84</td>
</tr>
<tr>
<td>Maninkakan</td>
<td>0.75</td>
</tr>
<tr>
<td>Haitian Kreyol</td>
<td>0.95</td>
</tr>
<tr>
<td>Malagasy</td>
<td>0.82</td>
</tr>
</tbody>
</table>

Table 4: All results for all languages. Product approach refers to using product of both phrase and lexical scores. Strength refers to adding connectivity features on top of Product. IBM Model 1 substitutes IBM Model 1 scores in place of product lexical score while Joint substitutes joint phrase scores in place of product phrase scores. Both Model 1 and joint approaches do not use the strength feature.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Mawu</th>
<th>Manin</th>
<th>Haitian-Creole</th>
<th>Malagasy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>7.08</td>
<td>9.41</td>
<td>33.6</td>
<td>18.8</td>
</tr>
<tr>
<td>Uniform</td>
<td>7.3</td>
<td>10.50</td>
<td>33.44</td>
<td>18.51</td>
</tr>
<tr>
<td>Product</td>
<td>7.39</td>
<td>10.91</td>
<td>33.84</td>
<td>19.17</td>
</tr>
<tr>
<td>Product + Strength</td>
<td>7.03</td>
<td>10.80</td>
<td>33.92</td>
<td>19.03</td>
</tr>
<tr>
<td>IBM Model 1</td>
<td><strong>7.64</strong></td>
<td>10.69</td>
<td><strong>34.00</strong></td>
<td><strong>19.20</strong></td>
</tr>
<tr>
<td>Joint</td>
<td>7.42</td>
<td><strong>11.06</strong></td>
<td>33.85</td>
<td>19.10</td>
</tr>
</tbody>
</table>

4 Experiments

4.1 Datasets

Table 5 contains the details about the data sets for the four language pairs. Data for Mawukakan$^5$ and Maninkakan$^6$ has been released by LDC. For Malagasy, the training

---

5http://catalog.ldc.upenn.edu/LDC2005L01
6http://catalog.ldc.upenn.edu/LDC2013L01
and development sets have been used as-is\(^7\). As there is no separate heldout data, the top 500 sentences of the test set is used as heldout. All experiments in Haitian Kreyol use the same training, development, heldout and test sets as the WMT 2011 shared task. The training corpus for Haitian Kreyol comprises only 16% in-domain data, while the development, heldout and test sets comprise only real-world short messages. All but training data for Haitian Kreyol have raw and clean versions. The clean version has the same short messages as raw, but have been manually cleaned of misspellings and other errors e.g. caf* in raw has been changed to café in clean. The pivot language used in all our experiments is French because we can only use French for Mawukakan and Maninkakan. We experimented with additional pivot languages for Haitian Kreyol but they did not help. The pivot-target data set is the 1.9M sentence pair French-English EuroParl (v7) corpus.

### 4.2 Setup

All the experiments have been run using the Moses toolkit [Koehn et al., 2007], following the standard pipeline. After tokenizing and lowercasing and removing any empty lines, the alignments in both directions are generated using GIZA++ [Och and Ney, 2003], followed by the –grow-diag-final-and heuristic to extract phrases. Weights for the features are learnt based on Algorithm 1 outlined in section 3.4 (Both tuning and heldout sets used are same for all the results in Table 4.) All BLEU scores reported are case-insensitive. SRILM [Stolcke, 2002] was used to generate the language models. For Haitian Kreyol, an interpolated 5-gram language model, using the English side of WMT data and the English side of Europarl, is used. For the other three languages, the language model used is 5-gram Gigaword. We use KenLM [Heafield, 2011] for LM scoring during decoding.

### 4.3 Results

The BLEU scores for all languages are in Table 4. Baseline in Table 4 refers to translation model generated by just using the source-target parallel data for each of the language pairs. We use another baseline, Uniform that refers to using a triangulated translation model combined with Baseline, but by using uniform weights (0.5 each.) All the BLEU scores, including baseline scores, are reported on the held-out data (devtest-

---

\(^7\)http://www.ark.cs.cmu.edu/global-voices/
Table 6: Examples of improvements in translations. These examples show how the pivot language can provide new useful candidate translations missing from the direct system.

<table>
<thead>
<tr>
<th>Language</th>
<th>Category</th>
<th>Example translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Haitian Kreyol</td>
<td>Before</td>
<td><em>do we still have earth-shock for haiti?</em></td>
</tr>
<tr>
<td></td>
<td>After</td>
<td><em>are there always earthquake in haiti?</em></td>
</tr>
<tr>
<td></td>
<td>Reference</td>
<td><em>are there any more earthquakes in haiti?</em></td>
</tr>
<tr>
<td>Maninkakan</td>
<td>Before</td>
<td><em>we will go there tɛnɛnlön</em></td>
</tr>
<tr>
<td></td>
<td>After</td>
<td><em>we will go there on monday.</em></td>
</tr>
<tr>
<td></td>
<td>Reference</td>
<td><em>we will go there on monday.</em></td>
</tr>
</tbody>
</table>

Table 7: Significance tests for our results. All use the same tuning and heldout set. (We used multeval [Clark et al., 2011] for the significance tests)

<table>
<thead>
<tr>
<th></th>
<th>Mawu</th>
<th>Manin</th>
<th>Kreyol</th>
<th>Malagasy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our best vs Baseline</td>
<td>&gt;0.05</td>
<td>&lt;0.01</td>
<td>&gt;0.05</td>
<td>&gt;0.05</td>
</tr>
<tr>
<td>Our best vs Uniform</td>
<td>&gt;0.05</td>
<td>&gt;0.05</td>
<td>&lt;0.03</td>
<td>&lt;0.03</td>
</tr>
</tbody>
</table>

clean for Haitian Kreyol.) For Haitian Kreyol, our baseline BLEU score is +0.6 BLEU points higher than the best system from the 2011 Workshop on Haitian Kreyol. Despite using a disjoint and out-of-domain Bible as source-pivot, both Haitian Kreyol and Malagasy lead to a better BLEU score compared to both our baselines. For Mawukakan, we observed that triangulation was not causing a significantly better BLEU score while Maninkakan showed a BLEU score increase of 1.5 points. The significance results are shown in Table 7. Except in the case of Haitian Kreyol where it improves by a small margin, adding the two connectivity features reduces the BLEU score. This is likely due to source-pivot having a small intersection with cleaner Europarl alignments. In Mawukakan and Maninkakan, 60% and 66% phrase pairs have a source connectivity strength of more than 0.5 while 67% and 69% have more than 0.5 in the backward direction. Unsupervised alignments on the triangulated phrase table (using IBM Model 1) helps in the case of Malagasy and Haitian Kreyol. Adding the Joint and decomposed conditionals as features does well Maninkakan, leading to the best system for it, while IBM Model 1 lexical scores combined with the product of phrase scores works best for Haitian Kreyol and Malagasy. On the WMT 2011 Haitian Kreyol devtest-clean data, our system gets 34% BLEU score and a prominent web-based free translation system gets 16.72%. Example translations are shown in Table 6.

5 Conclusion

In this paper, we compared previously proposed and novel models, features and design choices in triangulation for low-resource languages. We show that in a noisy domain adaptation setting which we faced in Haitian Kreyol and Malagasy due to the Bible as a source-pivot corpus, the use of unsupervised alignments to compute the phrase
table feature scores led to a significantly higher BLEU score. We showed that the joint probability method [Cohn and Lapata, 2007] is better for languages with short, smaller sized phrase tables which is the case in Maninkakan. For interpolating the direct source-target system with the source-pivot-target system, we introduced an algorithm to automatically learn the mixture weights. Our algorithm provides better results across different low-resource language pairs.

References


An Arabizi-English Social Media Statistical Machine Translation System

Jonathan May∗
USC Information Sciences Institute, Marina del Rey, CA 90292
jonmay@isi.edu

Yassine Benjira
ybenjira@ sdl.com

Abdessamad Echihabi
aechihabi@sdl.com
SDL Language Weaver, Los Angeles, CA 90045

Abstract

We present a machine translation engine that can translate romanized Arabic, often known as Arabizi, into English. With such a system we can, for the first time, translate the massive amounts of Arabizi that are generated every day in the social media sphere but until now have been uninterpretable by automated means. We accomplish our task by leveraging a machine translation system trained on non-Arabizi social media data and a weighted finite-state transducer-based Arabizi-to-Arabic conversion module, equipped with an Arabic character-based n-gram language model. The resulting system allows high capacity on-the-fly translation from Arabizi to English. We demonstrate via several experiments that our performance is quite close to the theoretical maximum attained by perfect deromanization of Arabizi input. This constitutes the first presentation of a high capacity end-to-end social media Arabizi-to-English translation system.

1 Introduction

Arabic-English machine translation systems generally expect Arabic input to be rendered as Arabic characters. However, a substantial amount of Arabic in the wild is rendered in Latin characters, using an informal mapping known as Romanized Arabic, Arabish, or Arabizi. Arabizi mainly differs from strict transliteration or romanization schemes such as that of Buckwalter or ALA-LC in that it is not standardized. Usage is inconsistent and varies between different dialect groups and even individuals. Despite these drawbacks, Arabizi is widely used in social media contexts such as Twitter. As can be seen in Figure 1, it is not uncommon for users to use a mix of Arabic script, Arabizi, and even foreign languages such as English in their daily stream of communication.

Arabizi can be viewed as a romanization of Arabic consisting of both transliteration and transcription mappings. Transliteration is the act of converting between orthographies in a way that preserves the character sequence of the original orthography. An example of transliteration in Arabizi is the mapping of the character ٣ to ‘3’ due to the similarity of the glyphs. Transcription (specifically, phonetic transcription) between orthographies is the act of converting in a way that preserves the spoken form of the original orthography as interpreted by a reader of the new orthography’s presumed underlying language. An example of transcription in Arabizi is the mapping of the character چ to any of ‘g’, ‘j’, or “dj.” This reflects the fact that in various

∗ This work was done while the first author was employed by SDL Language Weaver

1http://www.loc.gov/catdir/cpso/cpsd/romanization/arabic.pdf
dialects may be pronounced as [g] (as in god), [ʒ] (as in vision), or [dʒ] (as in juice), and that the digraph “dj” is used in French for [dʒ].

For a machine translation system to properly handle all textual language that can be called “Arabic,” it is essential to handle Arabizi as well as Arabic script. However, currently available machine translation systems either do not handle Arabizi, or at least do not handle it in any but the most limited of ways. In order to use any of the widely available open-source engines such as Moses (Koehn et al., 2007), cdec (Dyer et al., 2010), or Joshua (Post et al., 2013), one would need to train on a substantial corpus of parallel Arabizi-English, which is not known to exist. Microsoft’s Bing Translator does not appear to handle Arabizi at all. Google Translate only attempts to handle Arabizi when characters are manually typed, letter by letter, into a translation box (i.e. not pasted), and thus cannot be used to translate Arabizi web pages or documents, or even more than a few paragraphs at once.2

Because much communication is done in Arabizi, particularly in social media contexts, there is a great need to translate such communication, both for those wanting to take part in the conversations, and those wanting to monitor them. However, the straightforward approach to building an Arabizi-English machine translation system is not possible due to the lack of Arabizi-English parallel data.

In this paper we address the challenge of building such an end-to-end system, focusing on coverage of informal Egyptian communication. We find that we are able to obtain satisfactory performance by enhancing a conventionally built Arabic-to-English system with an initial Arabizi-to-Arabic deromanization module. We experiment with manually built, automatically built, and hybrid approaches. We evaluate our approaches qualitatively and quantitatively, with intrinsic and extrinsic methodologies. To our knowledge, this is the first end-to-end Arabizi-English social media translation system built.

2There are other online tools for rendering real-time typed Arabizi into Arabic script for use in search engines, such as Yamli (www.yamli.com).
2 Building an Arabizi-to-Arabic Converter

The design of our phrase-based machine translation system is modular and uses weighted finite-state transducers (wFSTs) (Mohri, 1997) to propagate information from module to module. It can thus accept a weighted lattice of possible inputs and can generate a weighted lattice of possible outputs. Our Arabizi-to-Arabic converter is one module in a pipeline that tokenizes, analyzes, translates, and re-composes data in the process of generating a translation. A schematic overview of the modules in our translation system is shown in Figure 2. An advantage of this framework is that it allows us the opportunity to propagate ambiguity through the processing pipeline so that difficult decisions may be deferred to modules with better discriminative abilities. As an example, consider the sequence “men” which could represent either the English word “men” or an Arabizi rendering of من (from). Without contextual translation of surrounding words, it is difficult to know whether the author intended to code switch to English or not. In the context of translations of surrounding words, this may be clearer, but it is inconvenient to build deromanization directly into an already complicated machine translation decoder. We find an effective solution is to persist both alternatives in the translation pipeline and ultimately let the translation module decide which input path to take. Thus the phrase “the monuments من film 7elw awii” (the monuments من very nice film) may be handled alongside the sentence “Howa nas kteer من el skool ray7een?” (Are there many people من school going?). In this work we do not consider attempts to translate code switches into languages other than the source – thus, switches into French or English, for example, would be passed to the output untranslated.

We design our converter module as a character-based wFST reweighted with a 5-gram character-based n-gram language model of Arabic. The language model is straightforwardly learned from 5.4m words of Arabic. We use a character-based language model instead of a word-based language model in order to avoid “over-correcting” out-of-vocabulary words, which are typically Arabic names. A portion of the character-based wFST is shown in Figure 3. Next we describe the strategies considered in its construction.
Figure 3: Portion of a wfST used to perform deromanization. This wfST represents the conditional probability of Arabic character sequences given Arabizi character sequences. In the portion shown we see that “5a” can be transformed to خ with probability 0.67 and to خ with probability 0.33, while ‘t’ can be transformed to ح with probability 0.84 and to ح with probability 0.16. The self-loop labeled ‘ρ’ follows the convention of Allauzen et al. (2007) and represents all character sequences not otherwise indicated. The complete wfST has 962 states and 1550 arcs.

<table>
<thead>
<tr>
<th></th>
<th>Test 1</th>
<th>Test 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segments</td>
<td>7,794</td>
<td>27,901</td>
</tr>
<tr>
<td>English word tokens</td>
<td>51,163</td>
<td>168,677</td>
</tr>
<tr>
<td>‘Arabizi’ word tokens</td>
<td>35,208</td>
<td>118,857</td>
</tr>
<tr>
<td>Percent deromanizable</td>
<td>78.2</td>
<td>97.7</td>
</tr>
</tbody>
</table>

Table 1: Statistics of the test corpora of parallel data used for intrinsic and extrinsic evaluation. The source side of the parallel data is presumed to be Arabizi, but the percentage of deromanizable tokens (those that contain Latin characters) indicates a more heterogeneous mix comprising emoticons, Arabic characters, and other symbols.
Figure 4: Portion of (left) manually constructed and (right) automatically induced Arabizi-to-Arabic conditional probability table. The automatically induced table includes wider coverage not in the manual table (e.g. “th” → “ت”) and multi-character sequences unlikely to be thought of by an annotator (e.g. “3an” → “عن”).

2.1 Expert construction

As a first attempt at building an Arabizi-to-Arabic wFST, we asked a native Arabic speaker familiar with finite-state machines to generate probabilistic character sequence pairs for encoding as wFST transitions. This effort yielded a set of 83 such pairs, some of which are shown in the left side of the table in Figure 4. While these entries largely match conventional tables of Arabizi-to-Arabic mapping, it is clear that even a human expert might easily construct a less-than-optimal table. For instance, while it is straightforward for a human to choose to deterministically map the sequence “sh” to the Arabic shin (€)، this would be a bad idea. Such a choice only covers cases where “sh” is intended to convey the voiceless postalveolar fricative [ʃ] (as in shower). The same character sequence can also be used to convey an alveolar fricative followed by a glottal fricative, [ʃh] (as in mishap) though, as in English, this sequence is relatively uncommon in Arabic. It is hard in general for humans to estimate character sequence frequencies; our human expert gave equal weight to the voiceless and voiced demromanizations of “th,” respectively, [TH] (as in bath) and [DH] (as in father). In fact, [TH] is more likely in Arabic. It is also difficult and tedious to consider correspondences between sequences of more than two characters, but such context is sometimes necessary. The Arabizi character ‘a’ has many potential corresponding Arabic characters, and sometimes should not correspond to any character at all. But this is highly context-dependent; in the sequence “3an”, for example, the ‘a’ represents the “short” Arabic vowel “fatha,” which is not typically rendered in everyday Arabic script. Creating the correspondences that properly differentiate between long and short vowels in all proper contexts with all appropriate probabilities seems like a task that is too difficult for a human to encode.

2.2 Machine Translation-based construction

For the next attempt to build a wFST we sought inspiration in statistical machine translation system construction, which begins with the unsupervised alignment of words in hand-aligned sentences. We collected a corpus of 863 Arabizi/Arabic word pairs. We treated the word pairs

---

http://en.wikipedia.org/wiki/Arabico_chat_alphabet

After much thought, we came up with “تسهيل” or “tashil” (facilitate).
as sentence pairs and the characters as words, and estimated Arabizi-to-English character alignments using a standard GIZA implementation (Och and Ney, 2003) with reorderings inhibited. We then extracted character sequence pairs up to four characters in length per side that were also consistent with the character alignments, in accordance with standard practice for building phrase translation correspondence tables (Koehn et al., 2003). This resulted in a set of 3138 unique sequence pairs. We estimated conditional probabilities of Arabic given Arabizi by simple maximum likelihood. A portion of the learned table is shown on the right side of Figure 4. We can see that, in comparison to the manually constructed table on the left side of the figure, the automatically constructed table captures more—perhaps unintuitive—correspondences, and sequence pairs which provide longer context. Figure 5 compares the distribution of the lengths of the sequences learned via manual and automatic means. Note that while this automatic method learns long-context sequences, the manual annotator indicated cases of character deletion (generally of vowels) that are not learnable using this approach. However, the effects of deletion are covered via the automatic method’s learning of long-context sequences where the Arabic sequence is shorter than the Arabizi sequence (see the examples for “3an” in Figure 4). Another potentially negative consequence of the automatic approach is that many useless, noisy pairs are introduced, and this can degrade quality and impact performance.

2.3 Semi-automatic construction

We sought to marry the small description length and human intelligence behind the manual approach with the empirically validated probabilities and wide coverage of the automatic approach. Consequently, after inspecting the automatically built wFST, we constructed a reduced version that only contained sequence pairs from the original if the Arabizi side had fewer than three characters (see Figure 5). We then added the vowel-dropping sequence pairs from the manual wFST. This forms a hybrid of the two aforementioned constructions we call the “semi-automatic” method. While this manual intervention was feasible given the relatively small size of the automatically generated table and the availability of a native Arabic speaker, a more prin-

---

The table below shows the distribution of Arabizi-to-Arabic character sequence lengths in automatic and manually generated approaches to wFST building. Entries in *boldface* indicate the subsets of the automatic or manual construction that were included in the semi-automatic construction.

<table>
<thead>
<tr>
<th>Arabizi length</th>
<th>Arabic length</th>
<th>automatic count</th>
<th>manual count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>55</td>
<td>51</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>178</td>
<td>25</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>341</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>112</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>736</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>415</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>369</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>698</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>216</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 5: Distribution of Arabizi-to-Arabic character sequence lengths in automatic and manually generated approaches to wFST building. Entries in **boldface** indicate the subsets of the automatic or manual construction that were included in the semi-automatic construction.
Table 2: Deromanization performance (note: not machine translation performance) of manually and automatically constructed modules, measured as word-based BLEU against a reference deromanization.

<table>
<thead>
<tr>
<th>deromanization approach</th>
<th>BLEU Test 1</th>
<th>BLEU Test 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>18.2</td>
<td>0.3</td>
</tr>
<tr>
<td>manual</td>
<td>20.1</td>
<td>1.7</td>
</tr>
<tr>
<td>manual + lm</td>
<td>21.5</td>
<td>2.9</td>
</tr>
<tr>
<td>automatic + lm</td>
<td>25.6</td>
<td>7.7</td>
</tr>
<tr>
<td>semi-automatic + lm</td>
<td>25.8</td>
<td>8.0</td>
</tr>
</tbody>
</table>

2.4 Intrinsic Evaluation

Even though our wFST-based machine translation system architecture is designed such that we can persist multiple deromanization (and non-deromanization) possibilities, it is helpful to examine the Viterbi deromanization choices of our methods, both qualitatively and quantitatively.

For quantitative evaluation, both intrinsic and extrinsic, we use two test corpora of sentence-aligned Arabizi-English social media data made available to us as part of DARPA-BOLT. Statistics of the corpora are shown in Table 1. The data also includes reference deromanizations of the Arabizi. We evaluate our deromanization approaches using the familiar BLEU metric against these reference deromanizations. The results are shown in Table 2. We see that the inclusion of a language model is helpful, and that the models influenced by corpus-based automatic learning (i.e. “automatic” and “semi-automatic”) outperform the manual model. We note, however, that the semi-automatic model, which is strongly influenced by the manual model, outperforms the automatic model slightly, and with far fewer transducer arcs.

One might expect 0 BLEU for the baseline case, where we use no deromanization method at all. This is not so due to the nature of social media data. As indicated in Table 1, many non-Arabizi tokens, such as emoticons, URLs, Arabic words, and English code switches, occur throughout the data, often mixed into predominantly Arabizi segments. The Test 1 corpus contains a significantly larger percentage of such tokens than the Test 2 corpus.

One might also expect higher overall BLEU scores at the bottom of Table 2, given the general track record of transliteration performance (Darwish, 2013; Al-Onaizan and Knight, 2002). We note that dialectical Arabic is in general not a written language, and as such there are many different spellings for words, even when rendered in Arabic script. Thus the task is closer to machine translation than classic transliteration (in that “correctness” is a squishy notion). Additionally, we did not specifically optimize our deromanizer for this intrinsic experiment, where we must decide whether or not to deromanize a possibly non-Arabizi word. Choosing incorrectly penalizes us here but should not impact extrinsic MT performance (evaluated in Section 4), due to our pipeline architecture’s ability to present both deromanized and non-deromanized options to downstream modules (see discussion in Section 2).

For some qualitative analysis, we consider an example comparison between our various deromanizer approaches in Figure 6. We observe the following:

- The Arabizi sentence starts with the chat acronym “isa,” which is expandable to إِنَّ فَيَاءَ اللَّهِ “in sha allah” (God willing). The manual wFST outputs “sa” while the automatic wFST outputs “issa.” Both are expected to be wrong, since acronyms are not handled in the current approach.
### 3 System Description

As illustrated in Figure 2, our deromanization module is one component in a pipeline of processing that forms a machine translation system. Aside from the deromanization module, which we vary in the following experiments, our system is constant and built as follows: The preprocessing additionally consists of a regular expression-based tokenization and normalization module to separate punctuation, and a word morphological segmentation module based on the type-based unsupervised approach of Lee et al. (2011). The machine translation module is phrase based, in the style of Koehn et al. (2003), and is trained on informal Arabic-English parallel and monolingual data made available through DARPA BOLT. The post-processing consists of deterministic detokenization based on the output word sequence. The capitalizer is part of our pipeline, as noted in Figure 2, but since we do not evaluate cased translations it was turned off for these experiments.

### 4 Extrinsic Experiments

In Table 3 we show the results of evaluating our informal Arabic-English MT system on the two aforementioned test sets while equipped with various configurations of the deromanization module. We also evaluate, as an upper bound, performance using a system with no deromanization module, but with a reference deromanization as input. We report detokenized, case-free
Table 3: Comparison of end-to-end MT performance using a deromanization module and Arabic-English system to translate Arabizi-English. The automatically learned wFST approach outperforms the manual wFST and makes good progress toward the reference deromanization upper bound. Scores reported are detokenized, lowercased BLEU.

<table>
<thead>
<tr>
<th>deromanization approach</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>7.7</td>
</tr>
<tr>
<td>manual</td>
<td>9.6</td>
</tr>
<tr>
<td>manual + lm</td>
<td>12.0</td>
</tr>
<tr>
<td>automatic + lm</td>
<td>15.1</td>
</tr>
<tr>
<td>semi-automatic + lm</td>
<td>15.3</td>
</tr>
<tr>
<td>reference deromanization</td>
<td>18.4</td>
</tr>
</tbody>
</table>

BLEU. The scores in Table 3 track those in Table 2, indicating a strong correlation between deromanizer performance and translation performance.

Turning to the qualitative results in Figure 6, we note the following:

- Although the non-deromanized system mostly passes the input through unchanged, we produce the words “god” and “fear.” The latter is likely an error due to a spurious low-count alignment of “ele” to “fear” in training data, but the former is due to a correspondence with “isa,” which, as previously noted, is shorthand for “god willing.” This is indicative of small amounts of Arabizi appearing in our training data.

- In the manual-based cases, the incorrect deromanization of “isa” leads to an unknown Arabic word being selected and then transliterated back into English, producing “asa” or “sa.” In the automatic-based cases the same could have happened, but the decoder instead chose to use the non-deromanized alternative and produced “god” as in the baseline case. Naturally, the reference deromanization, which correctly expands the acronym, leads to the best translation of this token.

- Since our MT engine normalizes away ligatures, the substantial differences between our deromanization approaches and the reference deromanization due to ligature placement results in little tangible effect on translation performance. This accounts for the correct translations of “at the end of the week.”

- The deromanizers’ inabilities to properly include the additional lam in “ele” accounts for the erroneous translation of “ele” as “to.”

5 Related Work

After this work had been substantially completed, we became aware of a similar effort by Al-Badrashiny et al. (2014). That effort, which resulted in the “3arrib” standalone deromanizer for Egyptian Arabic, also uses a wFST-based approach but verifies suitability using a hand-crafted Arabic morphological analyzer. Additionally, an effort was made in 3arrib to handle 32 special cases such as the expansion of “isa.” We compare their work to ours in Table 4. It should be noted that the 3arrib system was used in the preparation of the Test 1 and Test 2 data. That is, the initially collected Arabizi data was run through 3arrib, then post-edited by annotators. The intrinsic empirical results in particular should thus be taken with a grain of salt.
<table>
<thead>
<tr>
<th>deromanization approach</th>
<th>deromanization</th>
<th>translation</th>
<th>Test 1</th>
<th>Test 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>none</td>
<td></td>
<td></td>
<td>18.2</td>
<td>0.3</td>
</tr>
<tr>
<td>semi-automatic + lm</td>
<td></td>
<td></td>
<td>25.8</td>
<td>8.0</td>
</tr>
<tr>
<td>3arrib (Al-Badrashiny et al., 2014)</td>
<td></td>
<td></td>
<td>56.0</td>
<td>51.0</td>
</tr>
<tr>
<td>reference deromanization</td>
<td></td>
<td></td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>18.4</td>
<td>14.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>17.9</td>
<td></td>
</tr>
</tbody>
</table>

Table 4: A comparison of qualitative and quantitative, extrinsic and intrinsic results using the deromanization method described in the current work and that of Al-Badrashiny et al. (2014). While our work does not employ deep expert knowledge such as a hand-built morphological analyzer or special case handling such as acronym expansion, we are nonetheless able to build a system with comparable extrinsic performance to the boutique system of Al-Badrashiny et al. (2014). Note too that the intrinsic scores of Al-Badrashiny et al. (2014) reflect the fact that the gold data for this task was constructed by post-editing 3arrib output.
Darwish (2013) addresses many of the problems tackled in this work, though not in the context of machine translation. Like our baseline experiments, that work uses a hand-constructed transliteration table to map between Latin and Arabic sequences. Darwish (2013) places particular emphasis on detecting the difference between Arabizi and non-Arabizi words, and not attempting to deromanize the latter. He trains a conditional random field (CRF) to identify Arabizi words and reports accuracy of 98.5%. Since we have the luxury of downstream modules that can take ambiguous input (see Section 2), we simply allow each word to be transliterated or not and allowed nonsense deromanizations of non-Arabic words to be ignored by the translation engine in lieu of handling the original word. While the CRF approach is an appealing one that we will consider, we note that by not making a firm decision we allow words that are ambiguously Arabizi or English to be discriminated by a system that contains the rich context necessary for translation.

Chalabi and Gerges (2012) present an approach to Arabizi transliteration and mention the applicability of this functionality to improving machine translation but do not specify the approach taken in great detail.

Irvine et al. (2012) perform deromanization of Urdu as part of an overall normalization task for cleaning Urdu text messages. Their approach to building a subsequence correspondence table, which is described in Irvine et al. (2010), is similar to ours, though their training data does not include Arabizi.

Al-Onaizan and Knight (2002) use a cascade of wFSTs to attack the converse problem, that is, romanizing names from Arabic script into English.

This work can be considered a special case of handling user-generated content, as opposed to more formal content such as that from news or government sources. Others who have focused on handling user-generated content for machine translation include Jiang et al. (2012); Pennell and Liu (2011) and Carter et al. (2011). We took a comparatively simple approach to special cases such as URLs, emoticons, and hash tags, by using regular expressions to avoid translating untranslatable entities or splitting up special formatting.

6 Conclusion

Translation systems that can cope with the realities of informal communication need to be built with an understanding of the cultural forces that shape the way communication happens. In this work, we explored the consequences of societies wishing to communicate with a language that is not normally written in Latin characters (or, indeed, written at all) but being constrained to the Latin character set for historical, technological, or perhaps arbitrary reasons. These limitations prove no real barrier to infinitely creative humans but can confound computer systems built with regular assumptions in mind. We have shown that adapting our systems to match real-world behavior is not difficult, but requires an awareness of the forces at play.

Acknowledgement

We thank Steve DeNeefe, Dragos Munteanu, Thomas Wieland, Eli Pak, Amos Kariuki, and Markus Dreyer for their contributions to this work. We thank Aliya Deri, Marjan Ghazvininejad, Nima Pourdamghani, Hui Zhang, Kevin Knight, and Daniel Marcu for providing useful feedback. This work was supported by TSWG contract N41756-13-C-3050 and by DARPA contract HR0011-12-C0014.

References


Automatic Dialect Classification for Statistical Machine Translation

Saab Mansour
Aachen University, Aachen, Germany
mansour@cs.rwth-aachen.de

Yaser Al-Onaizan
onaizan@us.ibm.com
Graeme Blackwood
blackwood@us.ibm.com
Christoph Tillmann
cstill@us.ibm.com
IBM T.J. Watson Research Center, Yorktown Heights, NY, USA

Abstract

The training data for statistical machine translation are gathered from various sources representing a mixture of domains. In this work, we argue that when translating dialects representing varieties of the same language, a manually assigned data source is not a reliable indicator of the dialect. We resort to automatic dialect classification to refine the training corpora according to the different dialects and build improved dialect specific systems. A fairly standard classifier for Arabic developed within this work achieves state-of-the-art performance, with classification precision above 90%, making it usefully accurate for our application. The classification of the data is then used to distinguish between the different dialects, split the data accordingly, and utilize the new splits for several adaptation techniques. Performing translation experiments on a large scale dialectal Arabic to English translation task, our results show that the classifier generates better contrast between the dialects and achieves superior translation quality than using the original manual corpora splits.

1 Introduction

Training data for statistical machine translation (SMT) are extracted from various sources representing different domains (e.g., newswire, webforums, ...). The source of the data (encapsulated by meta-information) can be utilized to perform domain adaptation using different techniques. For example, mixture modeling of grammars trained on different sources of data (Foster and Kuhn, 2007), or provenance features using different sources of data (Chiang et al., 2011).

The meta-information based corpora split may contain further domain granularities. In this work, we tackle the case where the corpora contain a mixture of dialects. Dialects refer to varieties of a language, differing by vocabulary, morphology, grammar, etc. In this scenario, the meta-information split is rendered unreliable, and better splitting is required to achieve improvements using standard adaptation methods.
We start with developing an automatic dialect classifier for the purpose of refining the corpora splits. The classifier is applied on an Arabic dialect identification task, where we distinguish between the Egyptian Arabic (ARZ) and Modern Standard Arabic (MSA) dialects. MSA is the standard written form of Arabic while ARZ and other dialectal forms are mainly used for speech. Due to the prevalence of MSA in written form, most of the corpora collected for training SMT systems contain a majority of MSA data. Nevertheless, dialectal data has a strong presence in on-line content such as weblogs, forums and user commentary. Applying an automatic dialect classifier on corpora designated as dialectal shows that a large portion of the data is actually MSA, making dialectal identification essential for a successful utilization of the data.

Next, we extensively experiment with applying the classifier output for domain-adaptation, and compare using the classifier output to using meta-information based data splits. Various adaptation methods are investigated, including: domain-specific SMT tuning, mixture modeling, and the so called provenance features. Applying the developed methods on a competitive dialectal Arabic to English translation task, where the Arabic data contains a mixture of dialects, our results show that using the classifier output improves over the meta-information based splits. We also show that some adaptation methods can hurt the performance, and a combination of techniques is required to guarantee improvements. Finally, we perform simple system selection of the dialect-specific SMT systems and show that we can achieve gains for all dialects.

The paper is structured as follows. We review related work in Section 2. The automatic dialect classifier is introduced in Section 3 and the adaptation methods in Section 4. The experimental setup is described in Section 5. Classification and translation results along with an analysis are discussed in Section 6 and Section 7 correspondingly. Lastly, we conclude with few suggestions for future work in Section 8.

2 Related Work

Various adaptation techniques have been suggested in the past for SMT. The techniques use either meta-information to define the different corpora, e.g., (Foster and Kuhn, 2007; Chiang et al., 2011) or automatic clustering methods, e.g., (Eidelman et al., 2012; Sennrich, 2012a), and focus on training data splitting. We differ from previous work by using automatic dialect classification to refine the splitting of the training data. Furthermore, we use the classifier to split the tuning and test sets, build dialect specific systems and combine them using system selection based on the dialect classification.

Interest in techniques for handling varieties of a language has been growing in the last few years. In 2014, two workshops will be held dealing with resources, techniques and tools specialized for language varieties, LT4CloseLang\(^1\) at EMNLP and VarDial\(^2\) at COLING. The discriminating similar languages (DSL) shared task (Tan et al., 2014) offers an opportunity for consistent comparison of different classification methods. The DSL evaluation is done mainly on European languages. In this work, we focus on dialectal varieties of the Arabic language. Nevertheless, the methods developed are generic and can be applied to other languages. Zbib et al. (2012) discuss machine translation of Arabic dialects. Using human annotated dialectal data, they achieve improvements over a general SMT system. We differ from their work by using automatic dialect classification for SMT. Previous work on dialect classification discussed

\(^{1}\)http://www.c-phil.uni-hamburg.de/view/Main/LTforCloseLang2014
\(^{2}\)http://corporavm.uni-koeln.de/vardial/
the definition of the problem, and built automatic classifiers (Zaidan and Callison-Burch, 2011; Elfardy and Diab, 2013; Zaidan and Callison-Burch, 2014). Ideas for applying the classifier for SMT were discussed but not implemented. In this work, we implement a competitive dialect classifier which is then successfully applied for SMT and shows strong improvements over a competitive baseline. To the best of our knowledge, this work presents the first successful application of automatic dialect classification for SMT.

3 Dialect Classification

The task of dialect classification attempts to identify the dialect of a given sentence. In this work, we use a supervised sentence-level dialect classifier to designate sentences as MSA or ARZ. Sentences with ARZ content or structure are identified as ARZ, otherwise they are marked as MSA (Zaidan and Callison-Burch, 2014). The classifier is trained on the Arabic Online Commentary Set (AOC) (Zaidan and Callison-Burch, 2011). The data consists of commentary by online readers of Arabic newspapers with a high degree of dialectical content, together with human-annotated labels indicating the dialect of each sentence. The data had been obtained by a crowd-sourcing effort. In the current paper, we focus on the MSA-ARZ\(^3\) split of the data. The split contains 25K sentences and 650K words, where around half of the sentences are annotated as MSA and half as ARZ.

To implement the automatic sentence classifier, we use a linear SVM framework, i.e., the open source LIBLINEAR toolkit (Hsieh et al., 2008; Fan et al., 2008). The trainer can easily handle a large number of instances and features. As the objective function, we use \(L_1\) regularized \(L_2\)-loss support vector classification\(^4\). We set the penalty term \(C = 0.5\). To classify a sentence \(t^n_1 = t_1...t_n\), we compute a linear score \(s(t^n_1)\) as follows:

\[
s(t^n_1) = \sum_{s=1}^{d} w_s \cdot \sum_{i=1}^{n} \phi_s(c_i, t_i)
\]

where \(\phi_s(c_i, t_i)\) is a binary feature function which takes into account the context \(c_i\) of token \(t_i\). The weight vector \(w \in \mathbb{R}^d\) is a high-dimensional vector obtained during training. In our experiments, we classify a tokenized sentence as being Egyptian Dialectal (ARZ) if \(s(t^n_1) > 0\).

The feature functions we use include token (word-level) unigram and bigram, Part-of-Speech unigram and dictionary based features. The features are combined according to Eq. 1. We described the used classifier and features in more detail in (Tillmann et al., 2014).

4 Adaptation Methods

In this section, we introduce different approaches to domain adaptation that will be utilized to generate dialect-specific SMT systems.

\(^3\)Note that we denote Egyptian Arabic with ARZ instead of the EGY label used by (Zaidan and Callison-Burch, 2011). ARZ is the standard ISO language code for Egyptian Arabic.

\(^4\)In the LIBLINEAR toolkit settings, we use solver type 5 with default termination criterion.
4.1 SMT tuning

We carry out a weight vector $\lambda^M_1 = \lambda_1...\lambda_M$ tuning of a standard log-linear SMT model:

$$q(e^I_1, f^J_1) = \sum_{m=1}^{M} \lambda_m h_m(e^I_1, f^J_1),$$  \hspace{1cm} (2)

where $(e^I_1, f^J_1)$ are target and source sentences of length $(I, J)$ correspondingly, and $h^M_1$ are feature functions.

We use pairwise ranking optimization (PRO) (Hopkins and May, 2011) to tune the scaling factors. Instead of optimizing the log-linear model probability, PRO directly optimizes the final translation quality. To perform adaptation using PRO tuning, the development set can be varied to represent different domains. We experiment with using different development sets obtained from different domains, as well as using the dialect classifier to obtain dialect specific development sets. The scaling factors obtained for a specific tuning set will represent an adapted system for the domain of the tuning set.

4.2 Mixture tuning

Mixture modeling is a technique for combining several models using weights assigned to the different components. Domain adaptation could be achieved using mixture modeling when the weights are related to the proximity of the components to the domain being translated. As we generate several translation models differing by the training corpora domain, interpolating these models could yield further improvements. In this work, we focus on mixture modeling using linear interpolation.

Linear interpolation is a commonly used framework for combining different SMT models (Foster and Kuhn, 2007). Given $n$ phrase models $p^n_i = p_1...p_n$, and $\lambda^n_i$ interpolation weights, linear interpolation is defined as follows:

$$p(\tilde{f}|\tilde{e}; \lambda) = \sum_i \lambda_i \cdot p_i(\tilde{f}|\tilde{e})$$  \hspace{1cm} (3)

In this work, the interpolation weights are optimized over a development set which represents a specific domain. We use the phrase model perplexity as an objective function:

$$\hat{\lambda} = \arg\min_{\lambda} \left\{ -\frac{1}{N} \sum_{(\tilde{f}, \tilde{e})} \log p(\tilde{f}|\tilde{e}; \lambda) \right\}$$  \hspace{1cm} (4)

$(\tilde{f}, \tilde{e})$ are phrase pairs extracted from the development set using standard phrase extraction methods (symmetrized word alignment and heuristic phrase extraction). We use the L-BFGS optimization technique as done by (Sennrich, 2012b). Note that we apply linear interpolation to all extracted rules (including phrase, hierarchical, and tree-to-string rules).

4.3 Provenance features

Chiang et al. (2011) suggest provenance features for improving SMT performance. Instead of
training one model on the whole data, they suggest to condition the models on the provenance, i.e., the meta-information (genre, collection) of the corpus the data is coming from.

In this work, we use IBM Model 1 lexical smoothing as provenance features. Splitting the training data into $n$ sub-corpora $z_1^{n}$, we introduce $2 \cdot n$ provenance features (for standard and inverse Model 1 directions) into the log-linear framework of SMT. The log-linear weights of the provenance features are optimized as part of the PRO tuning of the whole set of SMT features. Adaptation is then achieved by tuning the weights of the features to improve performance on a target dialect tuning set.

## 5 Experimental Setup

### 5.1 Training corpora

We evaluate our dialect adaptation methods empirically in the context of the BOLT Phase 2 Dialectal-Arabic-to-English task\(^5\). The dialect chosen for Phase 2 is Egyptian Arabic (ARZ). The BOLT program goes beyond previous projects, shifting the focus from translating structured standardized text, such as Modern Standard Arabic (MSA) newswire, to a user generated noisy text such as Arabic dialect forums or sms. Translating Arabic dialects is a challenging task due to the scarcity of training data and the lack of common orthography causing a larger vocabulary size and higher ambiguity. Due to the scarcity of the ARZ training data, MSA resources are being utilized for the project. In such a scenario, an important research question arises on how to use the MSA data in the most beneficial way to translate the given dialect.

The training data for the BOLT Phase 2 program is summarized in Table 1. The table includes information about domain, dialect and size (automatic classification results are discussed in Section 6.1). Preprocessing includes Arabic tokenization and segmentation based on

<table>
<thead>
<tr>
<th>Corpus</th>
<th>#Sent</th>
<th>#Ar tok</th>
<th>Meta-info dialect</th>
<th>Automatic dialect classification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ARZ(%)</td>
</tr>
<tr>
<td><strong>Train</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Forums</td>
<td>299K</td>
<td>4.1M</td>
<td>ARZ</td>
<td>183K (61%)</td>
</tr>
<tr>
<td>Broadcast</td>
<td>169K</td>
<td>3.9M</td>
<td>MSA</td>
<td>18K (11%)</td>
</tr>
<tr>
<td>Newswire</td>
<td>885K</td>
<td>24.9M</td>
<td>MSA</td>
<td>29K (3%)</td>
</tr>
<tr>
<td>Other</td>
<td>726K</td>
<td>5M</td>
<td>MIX</td>
<td>184K (25%)</td>
</tr>
<tr>
<td>All</td>
<td>2.1M</td>
<td>37.9M</td>
<td>MIX</td>
<td>415K (20%)</td>
</tr>
<tr>
<td><strong>Tune</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEV12</td>
<td>1 209</td>
<td>17 702</td>
<td>ARZ</td>
<td>507 (42%)</td>
</tr>
<tr>
<td>P1R6</td>
<td>2 715</td>
<td>52 206</td>
<td>ARZ</td>
<td>1 481 (55%)</td>
</tr>
<tr>
<td>DEV10-wb</td>
<td>968</td>
<td>42 092</td>
<td>MSA</td>
<td>49 (5%)</td>
</tr>
<tr>
<td><strong>Dev</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEV12</td>
<td>1 510</td>
<td>27 134</td>
<td>ARZ</td>
<td>584 (39%)</td>
</tr>
<tr>
<td>P1R6</td>
<td>1 137</td>
<td>17 991</td>
<td>ARZ</td>
<td>739 (65%)</td>
</tr>
<tr>
<td>DEV10-wb</td>
<td>1 059</td>
<td>42 563</td>
<td>MSA</td>
<td>46 (4%)</td>
</tr>
</tbody>
</table>

Table 1: BOLT parallel training data by genre, meta-information dialect and automatic dialect classification results. MIX may contain additional Arabic dialects. The number of sentences (#Sent) and Arabic tokens (#Ar Tok) are given. We report the percentage of sentences classified as ARZ or MSA for each of the corpora listed.

English preprocessing includes lowercasing and punctuation tokenization.

The Forums data was collected from Egyptian webforums, therefore it is written mainly in the ARZ dialect. Broadcast data was collected and transcribed manually from various Arabic TV sources. In the broadcast domain speakers usually use MSA but sometimes also switch (for a short phrase) to dialectal speech. The newswire data is mostly written in MSA. We tune and test the SMT systems using 3 sets: DEV12 is extracted from LDC2012E30-BOLT Phase 1 DevTest Source and Translation V4, and has 1 reference, P1R6 from LDC2012E124-BOLT Phase 1 Translation Training Data R6 (1 reference), and DEV10-wb from LDC2010E30 GALE Phase 5 DevTest NW & WB Translations V3.0 (4 references). Note that the sets include two parts, a tune part which is used mainly for PRO tuning and a held-out development part which is used for testing and will be displayed in the results. Most of the BOLT training data is available through the linguistic data consortium (LDC) and is regularly part of the NIST open MT evaluation. For language model training purposes, we use an additional 8 billion words (4B words from the LDC gigaword corpus and 4B words collected from web resources).

5.2 Translation system

We use an in-house implementation of a chart-based decoder (Zhao and Al-Onaizan, 2008). The decoder utilizes phrase, hierarchical, and tree-to-string rules to perform derivations. For the tree-to-string grammar, the source side of the parallel training data is parsed and word-alignment is performed. Tree-to-string rules together with their probabilities are then automatically learned from the data (Liu et al., 2006). Reordering patterns can be learned from linguistic labels assigned to chunks by combining parsing and alignment information. For Example, the rule \([X,VP][X,VB][X,NP] \rightarrow [X,NP][X,VB]\) rewrites a VP with two constituents VB and NP into an NP VB order in the target. The tree-to-string grammar bounds the search space to the available reordering patterns. However, if the correct word order cannot be generated by the tree-to-string grammar, the system resorts to hierarchical or phrase based rules to extend the coverage.

The hypothesis score is defined by the standard log-linear model combination, which includes in this case count-based features for phrase, glue, hierarchical and tree-to-string rules. Additional standard models such as length penalty and lexical smoothing are also incorporated into the decoder. All MT experiments are optimized with PRO to minimize the combined error measure of BLEU (Papineni et al., 2002) and TER (Snover et al., 2006), \((\text{TER-BLEU})/2\).

6 Classification Results

In this section, we present classification accuracies as well as classification results on the SMT training data. We train a dialect classifier as suggested in Section 3. The classifier performance is presented in Table 2. The table includes two sets of experiments, a 10-fold cross validation using the MSA-ARZ portion of the Arabic Online Commentary (AOC) data, and the performance on DEV12 (tune part in Table 1) when training the classifier on the whole AOC data. The performance is measured in terms of accuracy (fraction of sentences correctly tagged) and dialect precision and recall, e.g., for ARZ:

\footnote{For a list of the NIST MT12 corpora, see http://www.nist.gov/itl/iad/mig/upload/OpenMT12_LDCAgreement.pdf}
The classifier achieves high precision results on the ARZ portion of the sets. This is important for the adaptation experiments as we are mostly interested in adapting the systems towards the ARZ dialect. As most of the training data is MSA, having a correctly classified portion of ARZ data will help gain more improvements when adapting towards the ARZ domain.

In comparison to state-of-the-art, Elfardy and Diab (2013) report 85.3% accuracy for their best setup on a similar 10-fold cross-validation experiment. Zaidan and Callison-Burch (2014) report 87.9% accuracy on a similar setup. Our results show an improvement of 1.2% absolute over the best reported results on this task.

6.1 Classifier analysis

In this section, we run the classifier over the BOLT data and measure its dialectal degree, and whether the dialectal degree corresponds to the labeled provenance of the data. Classification statistics are presented in Table 1, where we report the number and percentage of sentences classified as ARZ or MSA. The ARZ forum data contains a majority of ARZ sentences, but quite a few sentences are MSA such as greetings and quotations from Islamic resources (Quran, Hadith ...). The broadcast conversation data is mainly MSA, but sometimes the speaker switches to dialectal usage for a short phrase and then switches back to MSA. Lastly, the newswire data has a vast majority of MSA sentences. We conclude that the data contains a mixture of dialects, and a more refined splitting using the dialect classification information could help improve adaptation methods.

Classifications examples from the BOLT data are given in Table 3. In the first document fragment, the user starts with MSA sentences, then switches to ARZ marked by the ARZ indicator and using the prefix before a verb which is not allowed in MSA. The user then switches back to MSA. The classifier is able to classify these sentences correctly. The second text fragment shows some sentences from the newswire corpus that are mis-classified. The first sentence contains the word which corresponds to the letter 'd' in the abbreviation 'tdk'. The word is contained in one of our ARZ dictionaries such that the corresponding binary based feature fires and triggers a mis-classification. In this context, the word is part of an abbreviation which is split in the Arabic text. In the other examples, only a few of the binary features fire and features that correspond to Arabic prefixes tend to support a classification as ARZ.

<table>
<thead>
<tr>
<th>Set</th>
<th>Acc</th>
<th>P</th>
<th>R</th>
<th>F</th>
<th>Acc</th>
<th>P</th>
<th>R</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>cross-validation</td>
<td>89.1</td>
<td>91.1</td>
<td>85.7</td>
<td>88.3</td>
<td>87.5</td>
<td>92.2</td>
<td>89.8</td>
<td></td>
</tr>
<tr>
<td>DEV12</td>
<td>87.8</td>
<td>92.8</td>
<td>83.0</td>
<td>87.6</td>
<td>83.2</td>
<td>93.0</td>
<td>88.0</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Arabic dialect classification results: predicting MSA vs. ARZ. Accuracy (Acc), precision (P), recall (R) and F-measure (F) are given in percentages [%].

\[
\text{precision} = \frac{\# \text{ARZ correctly tagged}}{\# \text{ARZ tagged}}, \quad \text{recall} = \frac{\# \text{ARZ correctly tagged}}{\# \text{ARZ reference}}
\]
Table 3: Automatic classification examples. The classes ARZ and MSA, Arabic source and English target sentences are given. Dialectal words are in **bold**.

<table>
<thead>
<tr>
<th>Classification result</th>
<th>Arabic</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td>correct</td>
<td>مSA</td>
<td>أنا قرأت اللفظة والردوة .</td>
</tr>
<tr>
<td></td>
<td>MSA</td>
<td>الموضوع فكرة حلوة</td>
</tr>
<tr>
<td></td>
<td>ARZ</td>
<td>وأننا مع الأخ الذي يقول</td>
</tr>
<tr>
<td></td>
<td>MSA</td>
<td>الدين مهم في كل حاجة</td>
</tr>
<tr>
<td>incorrect</td>
<td>ARZ</td>
<td>وقد قادتتي دي كه</td>
</tr>
<tr>
<td></td>
<td>ARZ</td>
<td>وينحو خبراء النقل باللغة</td>
</tr>
<tr>
<td></td>
<td>ARZ</td>
<td>لا أستطيع تذكر ما قال إلي</td>
</tr>
</tbody>
</table>

Table 4: PRO tuning adaptation: The baseline is tuned using P1R6+DEV10, the TUNE.MSA and TUNE.ARZ sets are based on the classifier output over the concatenation of all tuning sets.

<table>
<thead>
<tr>
<th>Tuning set</th>
<th>DEV12 (T-B)/2</th>
<th>P1R6 (T-B)/2</th>
<th>DEV10 (T-B)/2</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>15.35</td>
<td>15.02</td>
<td>0.73</td>
</tr>
<tr>
<td>TUNE.MSA</td>
<td>15.52</td>
<td>15.15</td>
<td>0.73</td>
</tr>
<tr>
<td>TUNE.ARZ</td>
<td>15.58</td>
<td>15.05</td>
<td>1.15</td>
</tr>
</tbody>
</table>

7 Translation Results

The baseline SMT system used in this work is based upon a tree-to-string decoder as described in Section 5.2. To create the rule tables, we use the concatenation of three word alignments, namely, HMM, IBM model 4 and maximum entropy aligner to maximize performance (Tu et al., 2012). The PRO tuning is done using the concatenation (P1R6+DEV10-wb).tune, as it performed best among all possible combinations.

Next, we experiment with the various adaptation methods suggested in Section 4. We focus on the comparison between using splits based on the meta-information and splits based on the automatic classifier output (Table 1).

7.1 SMT tuning

Tuning an SMT system using a domain-specific tuning set can adapt the scaling factors towards the target domain. For example, a bigger word-penalty scaling factor will encourage shorter sentences. Using meta-information based tuning sets, we found that the best combination is to use P1R6+DEV10-wb for tuning. To experiment with tuning sets based on dialect classification, we concatenate all tuning sets into TUNE=DEV12+P1R6+DEV10-wb. We then split the concatenated tuning set into an ARZ and an MSA part based on the classifier output and denote these splits TUNE.ARZ and TUNE.MSA respectively.

A comparison between the baseline system and the tuning using the classifier-based splits...
over the dev sets is given in Table 4. From the results, we note that tuning towards MSA does not hurt the results on DEV10-wb-dev, with a slight degradation on the ARZ dev sets. Tuning towards ARZ degrades the results on the ARZ dev sets and a bigger degradation is observed on the DEV10 set. Examining the scaling factors for the ARZ tuning set, almost no change is observed (in comparison to the baseline). We hypothesize that without domain-specific models, tuning towards a target domain is not effective. In the following experiments, we introduce more adaptation into the system and re-apply the PRO adaptation.

### 7.2 Mixture tuning

In this section, we experiment with optimizing the linear mixture weights towards a specific dialect as presented in Section 4.2. The optimization is done using phrase perplexity as the objective function. As components for the mixture, we use the meta-information split from Table 1. As tuning sets, we evaluate the performance of the meta-information based sets versus using the classifier. Note that we do not split the training corpora further according to dialects here, but concentrate on the tuning of the mixture weights instead.

The resulting optimal weights from the L-BFGS optimization are presented in Table 5. Note that we use All data (a concatenation of all corpora) as an additional corpus to ensure optimal translation results. The first block of weights is based on meta-information tuning sets. We experiment with the baseline SMT system tuning set as-well-as a supposedly ARZ tuning set (DEV12+P1R6). P1R6+DEV10-wb contains mostly MSA sentences, therefore the Newswire corpus is assigned the highest weight. When using the DEV12+P1R6 tuning set, the weight shifts to the Forums and All data, as they are the corpora most similar to the tuning set.

For the classifier split tuning sets, when tuning on TUNE.ARZ, weight is shifted to the Forums based model. Tuning on the MSA part TUNE.MSA, the weight shifts back to Broadcast, Newswire and All data, which contain a majority of MSA sentences. To summarize, we note that mixture tuning is producing expected results, and using the classifier splits assigns higher weights to the corresponding dialect. Therefore, the classifier based mixture tuning is more reliable and generates better contrast between the corpora. Next, we use the weights based on the classifier splits to create interpolated rule tables and build SMT systems using those tables.

The SMT results of different mixture modeling experiments are summarized in Table 6. Performing linear interpolation of the rule tables using uniform weights (linear.uniform) already achieves gains over the baseline. Note that PRO retuning using the baseline tuning set (P1R6+DEV10) is performed, unless stated otherwise. Linear interpolation with weights optimized on the MSA set (the weights associated with TUNE.MSA in Table 5), linear.MSA, achieves further gains on the MSA DEV10-wb-dev set, with a loss on the ARZ sets as expected.

#### Table 5: Optimal mixture weights for different tuning sets. The tuning sets are constructed using meta-information or the classifier output.

<table>
<thead>
<tr>
<th>Tune set</th>
<th>Forum</th>
<th>Broadcast</th>
<th>Newswire</th>
<th>Other</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>Meta-info.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P1R6+DEV10</td>
<td>0.23</td>
<td>0.26</td>
<td>0.34</td>
<td>0.06</td>
<td>0.12</td>
</tr>
<tr>
<td>DEV12+P1R6</td>
<td>0.40</td>
<td>0.15</td>
<td>0.05</td>
<td>0.03</td>
<td>0.37</td>
</tr>
<tr>
<td>Classifier</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TUNE.ARZ</td>
<td>0.62</td>
<td>0.11</td>
<td>0.03</td>
<td>0.06</td>
<td>0.17</td>
</tr>
<tr>
<td>TUNE.MSA</td>
<td>0.09</td>
<td>0.23</td>
<td>0.19</td>
<td>0.03</td>
<td>0.46</td>
</tr>
</tbody>
</table>
Table 6: Mixture tuning adaptation: linear interpolation using uniform weights (linear.uniform), ARZ and MSA optimized weights are compared. +pro.Dialect indicates PRO tuning adaptation.

Performing PRO tuning using TUNE.MSA over the linear.MSA system achieves further gains on the MSA set. The total gain over the baseline is -0.6% (T-B)/2 for DEV10-wb-dev. The resulting system is adapted for MSA sentences and performs well under this condition.

To build a system targeting the ARZ dialect, we repeat the same procedure as for MSA. We start with linear interpolation using ARZ optimized mixture weights (linear.ARZ), which achieves -0.3% and -0.4% improvements over DEV12-dev and P1R6-dev correspondingly. Loss is observed on the MSA set as expected. Adding the PRO tuning using the ARZ classified sentences hurts the results in this case, with a 0.2% degradation on DEV12-dev.

Finally we experiment with selecting hypotheses from the output of the MSA optimized system (linear.MSA+pro.MSA) and the ARZ optimized one (linear.ARZ+pro.ARZ). The idea is to use the MSA optimized system for MSA classified sentences and the ARZ optimized system for ARZ sentences. Using this selection, we might see improvements on the whole dev sets, as it might be the case that the ARZ system improved on the ARZ sentences and got much worse on the MSA sentences, masking the gains on the whole dev set. The system selection retains the gains on the MSA data, but not on the ARZ sets. We conclude that the ARZ system in this case did not improve on the ARZ part of the data. Next, we add domain specific models to the SMT system, giving more flexibility for PRO to overweight dialect specific features and target the ARZ dialect.

7.3 Provenance features

To compare meta-information based and classifier based corpora splits for provenance features, we devise two provenance setups: 1) m1Manual, manual splitting with 4 corpora, Forums, Broadcast, Newswire and Other to train Model 1 models in standard and inverse directions (8 additional features in the decoder), 2) m1Class, classifier based splitting, Forums, Broadcast and Other corpora are split into MSA and ARZ parts using the classification, the Newswire corpus is kept intact as it is mostly MSA (14 additional features in the decoder).

The results using the provenance features on top of the dialectal optimized systems are given in Table 7. From the results, we note that adding provenance features achieves further improvements, and using m1Class has a slight edge over the m1Manual provenance features. In this case, the system selection (from (lin+pro).MSA+m1Class and (lin+pro).ARZ+m1Class)
To analyze the results further, we split the DEV12 and P1R6 dev sets into the corresponding dialectal parts, and measure the effect of adding provenance features over these parts. In such a case, we expect that ARZ optimized systems will improve over the ARZ part, while MSA optimized systems will improve over the MSA part. The results are summarized in Table 8. Note that in this table we are using subsets of the dev sets. Concentrating on the MSA part of the dev sets, we note that adding the provenance features is improving mainly on P1R6, with a slight gain for classifier based provenance (m1Class) over meta-information based (m1Manual). As a contrast, the ARZ optimized system is performing poorly on the MSA parts of the dev sets. The picture is similar for the ARZ part of the dev sets, this time the main improvement is on the DEV12 set, with a bigger gain for m1Class over m1Manual, 0.3% (T-B)/2.

7.4 Translation examples

In this section, we perform manual translation error analysis. Translation examples are given in Table 9. The examples show that the system selection of dialectal optimized systems (sel.) improves over the baseline (base). The first two examples are ARZ sentences while the last is an MSA one. These examples were selected to demonstrate the difficulty of dialectal language translation and to show how a dialect classifier can remedy the problems encountered.

In the first sentence, the word العربية means ‘Arab’ but only in ARZ it could also mean ‘car’. The sentence is classified correctly, and the ARZ optimized system is able to generate the correct lexical meaning of the word. Similarly, in the second example, the word يَبْقِى means ‘has become’ in MSA and ‘is’ in ARZ. The ARZ system generates a better translation. The third sentence is an MSA sentence, where the baseline has a reordering error of ‘controls’ being generated before ‘professional’, and the word تراقي (consider) is dropped. The MSA optimized system generates a better reordering as-well-as a better lexical choice. We conclude
that ignoring the effects of dialectal data in MT makes the task even more ambiguous, and
dialectal identification is crucial to lessen the ambiguity and improve the lexical choice.

8 Conclusions and Future Work

In this work, we implement and successfully apply an automatic dialect classifier for SMT.
The classifier is applied on the BOLT task, where we compare meta-information based data
splits versus using the classifier output. The various splits are utilized for three adaptation
methods: PRO tuning adaptation, mixture adaptation and provenance features. For mixture
adaptation, our results show that the classifier based splits generate better contrast between
the different training corpora weights, where more emphasis is placed on the ARZ forums data
when using the ARZ tuning set based on the classifier output compared to the ARZ tuning set
based on meta-information. For PRO tuning adaptation, we conclude that using the classifier
splits without additional dialect specific models is not helpful and can degrade the performance.
When adding the provenance features, a system selection of ARZ and MSA optimized systems
improves over the baseline by 0.5% on the ARZ dev set.

In future work, it would be interesting to measure the effect of the classifier quality for
the adapted SMT systems. For mixture modeling, we started experimenting with training data
splitting by the classifier to create dialect specific rule tables and perform rule table interpola-
tion. A problem occurs when optimizing the mixture weights, where some of the ARZ splits
were assigned lower weights than the MSA counterparts when optimizing towards ARZ. We
hypothesize that this result is obtained due to many unknown phrase pairs in the ARZ tables
which are rather small in size. Smoothing for unknown phrase pairs should be applied when
more splits are used and sparseness becomes a problem. Many other techniques for adaptation
using dialect classification could be experimented with in future work. For example, phrase
level classification, or using the classifier scores as a feature in the SMT decoder.
Acknowledgement

The current work has been funded through the Broad Operational Language Translation (BOLT) program under the project number DARPA HR0011-12-C-0015.

References


A Tunable Language Model for Statistical Machine Translation

Junfei Guo\textsuperscript{1,2} \hspace{1cm} guojf@ims.uni-stuttgart.de
Juan Liu\textsuperscript{1} \hspace{1cm} liujuan@whu.edu.cn
Qi Han\textsuperscript{3} \hspace{1cm} qi.han@vis.uni-stuttgart.de
Andreas Maletti\textsuperscript{2,4} \hspace{1cm} maletti@ims.uni-stuttgart.de

\textsuperscript{1} School of Computer, Wuhan University, China
\textsuperscript{2} Institute for Natural Language Processing, University of Stuttgart, Germany
\textsuperscript{3} Institute for Visualization and Interactive Systems, University of Stuttgart, Germany
\textsuperscript{4} Institute of Computer Science, University of Leipzig, Germany

Abstract

A novel variation of modified Kneser-Ney model using monomial discounting is presented and integrated into the Moses statistical machine translation toolkit. The language model is trained on a large training set as usual, but its new discount parameters are tuned to the small development set. An in-domain and cross-domain evaluation of the language model is performed based on perplexity, in which sizable improvements are obtained. Additionally, the performance of the language model is also evaluated in several major machine translation tasks including Chinese-to-English. In those tests, the test data is from a (slightly) different domain than the training data. The experimental results indicate that the new model significantly outperforms a baseline model using SRILM in those domain adaptation scenarios. The new language model is thus ideally suited for domain adaptation without sacrificing performance on in-domain experiments.

1 Introduction

Language modeling (Manning and Schütze, 2001) is a central, important, and well-studied topic in natural language processing because the obtained language models (LM) are used in many diverse language technology tasks such as machine translation (Koehn, 2010b), speech recognition, and information retrieval (Manning et al., 2008). Most applied language models are based on \textit{n}-grams, which are sequences of \textit{n} consecutive words. Abstractly speaking, an \textit{n}-gram language model represents a probability distribution over sequences of \textit{n} words. These distributions are typically obtained with the help of maximum likelihood estimation (MLE) from large monolingual corpora. However, they are smoothed to move probability mass to \textit{n}-grams that are infrequent or unseen in the training data. The most popular smoothing method in statistical machine translation is the modified Kneser-Ney model by Chen and Goodman (1996), which is implemented in language model toolkits such as SRILM by Stolcke (2002) and KenLM by Heafield (2011).

To accommodate rare \textit{n}-grams, the relative frequencies of \textit{n}-grams in the training data are slightly discounted. Here we replace the discounting used in the modified Kneser-Ney model by a monomial discounting. This modification allows a simple adjustment (i.e., tuning)
of the obtained language models (via their discount parameters) to different domains via a standard tuning step (similar in principle to the parameter optimization used in statistical machine translation). Our new model is trained on a large monolingual corpus as usual, but its discount parameters are tuned to a small development set, which usually coincides with the tuning set used to tune the parameters of the machine translation system. We demonstrate that very little development data is sufficient to achieve good performance.

In general, an accurate estimation of cross-domain \( n \)-grams is difficult to achieve with only knowledge about in-domain \( n \)-grams because even huge in-domain training data is typically insufficient to combat cross-domain data sparseness. The standard solution interpolates the large trained LM with an additional (usually small) LM for the target domain. Our model can utilize both types of data in a single model because the tuning step of our monomial discounting model offers a natural way to adapt it to a new domain. The basic \( n \)-gram probabilities are trained using the large amount of background training data, but the new discount parameters are adjusted using data from the target domain. The tuning is driven by perplexity (Jelinek et al., 1977) as a standard measure of language model performance. We optimize the discount parameters such that the perplexity is optimal on the development data using a simple grid search (for our two discount parameters). In experiments we observe that the LM perplexity does not deteriorate compared to the baseline, which is a modified KNESER-NEY model as implemented in SRILM. This applies to both the in-domain as well as the cross-domain setups. More precisely, we observe solid improvements in the cross-domain setups and comparable (i.e., the same) performance in in-domain setups. In addition, our model can still be interpolated with a domain-specific LM to improve it even further.

Perplexity is a rather synthetic measure and does not necessarily correlate well with the performance of downstream tasks, such as statistical machine translation, that utilize language models. Consequently, we also confirm the benefits of our new language model with the help of an evaluation of statistical machine translation performance on medium-scale translation tasks (incl. Chinese-to-English and English-to-German). In these experiments we compare our new domain adaptation LM using monomial discounting to the well-known modified KNESER-NEY model that is implemented in both SRILM and KenLM. As translation models we use the popular phrase-based model (Zens et al., 2002) and the hierarchical phrase-based model (Chiang, 2005), which are both implemented in MOSES (Koehn et al., 2007). The obtained experimental results indicate that systems using our language model significantly outperform the baselines that use the SRILM language model. The improvements are particularly pronounced in domain adaptation scenarios. Only in the biological domain for English-to-German translation we observe no improvements at all, but in this case the overall translation quality (and the trained translation model) is potentially too poor to yield reasonable translations, so that the impact of the language model might be minimal.

In summary, we present a small monomial discounting LM, which can easily be tuned to new domains and is thus ideally suited for domain adaptation. This is achieved by optimizing the LM discount parameters on a small target domain corpus. In our experiments, we compared the LM performance of our model to the LM of the popular toolkits SRILM and KenLM. It shows that our language model works well on in-domain as well as cross-domain data. We implemented our model as a new LM in the MOSES statistical machine translation framework of Koehn et al. (2007) and evaluated our model on several major translation tasks. In those experiments we observed significant improvements in most cross-domain translation setups.

2 Related Work

There exists a wealth of different language models and evaluations of them, so we can only recall the basic antecedents of our work. Kneser and Ney (1995) presented a extension
of absolute discounting method and thereby established the popular KNESER-Ney models. Chen and Goodman (1996) proposed the modified KNESER-Ney model in their study, which quickly became the dominant $n$-gram language model. In addition, they already noted that interpolation generally works better than backoff. Brants et al. (2007) contributed stupid backoff, which is slightly cheaper to estimate. Finally, Schütze (2011) proposed a recursive DUPONT-ROSENFIELD model with polynomial discounting by interpolating class-based distributions (Brown et al., 1992) with the lower-order distributions. These models achieved improvements in perplexity when compared to the modified KNESER-Ney models. A simple and general scheme for the adaptation of stochastic language models was already presented by Kneser and Steinbiss (1993). Their adaptation method was used to improve a bi-gram language model.

Corresponding to the wealth of language models, there is also a wealth of implementations of them. We only mention IRSTLM by Federico et al. (2008) and MSRLM by Nguyen et al. (2007), which both implement several language models. We implemented our model in SRILM by Stolcke (2002). In our experiments we compare our model against the modified KNESER-Ney models implemented in SRILM and additionally KenLM, which is the recommended language model in the MOSES framework. SRILM is a popular toolkit for building and applying statistical $n$-gram-based language models and is used in speech recognition, statistical tagging and segmentation, and statistical machine translation. SRILM offers methods to compute the optimal interpolation weights for the corresponding domain models. Heafield (2011) contributed a scalable variant of the modified KNESER-Ney model that does not rely on pruning. KenLM was already evaluated in a statistical machine translation setup and significant improvements in terms of BLEU (Papineni et al., 2002) were observed (Heafield et al., 2013) at the expense of much larger language models.

3 Language Models

In this section, we recall the commonly used modified KNESER-Ney model (KN model), which is also used in our contrastive systems, and introduce the monomial discounting that we add to the KN models. This type of discounting was originally proposed for the POLKN models by Schütze (2011). Naturally, we also present the newly obtained KN models with monomial discounting in detail, which we will evaluate later on.

3.1 Modified Kneser-Ney model

The modified KN model was proposed by Chen and Goodman (1996). We present the general formulation for an $n$-gram language model. The model parameters are estimated on the training set, from which we extract occurrence counts $c(w)$ for all sequences $w \in \Sigma^\leq n$ of length at most $n$, where $\Sigma$ is our lexicon. Given $w \in \Sigma^k$ with $k \geq 1$, we let tail$(w)$ be the sub-sequence excluding just the first position; i.e., if $w = \sigma_1 \cdots \sigma_k$, then tail$(w) = \sigma_2 \cdots \sigma_k$. Instead of a single discount parameter $D$ (or a constant function $D$), they proposed to use three discount parameters $D_1, D_2, D_3$. More precisely, for every $n \geq 1$, $\sigma' \in \Sigma$, and $w' \in \Sigma^{n-1}$, let

$$p^{(n)}_{\text{KN}}(\sigma' | w') = \frac{c(w'\sigma') - D(c(w'\sigma'))}{c(w')} + \gamma^{(n)}(w') \cdot p^{(n-1)}_{\text{KN}}(\sigma' | \text{tail}(w'))$$

$$D(k) = \begin{cases} 0 & \text{if } k = 0 \\ D_1 & \text{if } k = 1 \\ D_2 & \text{if } k = 2 \\ D_3 & \text{otherwise} \end{cases}$$
where, to make the distribution sum to 1, they set
\[
\gamma^{(i)}(w') = \frac{\sum_{k \geq 0} D(k) \cdot \left| \{ \sigma \in \Sigma \mid c(w' \sigma) = k \} \right|}{\sum_{k \geq 0} \sum_{c(w') \in \Sigma} D(k) \cdot \left| \{ \sigma \in \Sigma \mid c(w' \sigma) = k \} \right|}.
\]

Kneser and Ney (1995) developed an estimate for the optimal value of their discount parameter \( D \), and Chen and Goodman (1996) derived the analogous values for the modified KN-model:
\[
D^*_i = i - (i + 1) \frac{n_1 n_i + 1}{n_1 n_i + 2 n_2 n_i}
\]
with \( i \in \{1, 2, 3\} \), where \( n_i = \left| \{ w \in \Sigma^n \mid c(w) = i \} \right| \) is the number of \( n \)-grams that appear exactly \( i \) times in the training data.

### 3.2 Our Model

In our monomial-discount domain-adaptation \( n \)-gram-based language model, we use exactly the same general approach as in the modified KN-models, but we replace the discount function by the monomial discount. Schütze (2011) proposed a polynomial discounting mechanism originally for his POLKN models with the intuition that the ideal discount \( D(k) \) in the model should grow monotonically with \( k \). More precisely, he replaced the KN-discount \( D \) by the discounting function \( E \) given for two discount parameters \( \rho \) and \( \gamma \) by \( E(k) = \rho \cdot k^\gamma \). Informally, the parameter \( \gamma \) controls the rate of growth of the discount as a function of \( k \), and the parameter \( \rho \) is a classical discount factor that can be scaled for optimal performance. We assume that \( \rho^0 = 0 \), so \( E(0) = 0 \).

We generally compute the \( n \)-th-level conditional probability \( p^{(n)}_{DA}(\sigma' \mid w') \) given the occurrence counts \( c(w' \sigma') \) and \( c(w') = \sum_{\sigma \in \Sigma} c(w' \sigma) \). In particular, we only consider lower-order levels if the \( n \)-gram was not seen in the training data (following backoff models). Note that we do not distinguish whether an \( n \)-gram occurs once or twice. The only remaining distinction is whether an \( n \)-gram occurs or not. Naturally, the number of occurrences modifies the discount \( E \). We denote our tunable language model \( p_{DA} \) and define it for every \( n \geq 1 \), \( \sigma' \in \Sigma \), and \( w' \in \Sigma^{n-1} \) as follows.

\[
p^{(n)}_{DA}(\sigma' \mid w') = \begin{cases} 
\frac{c(w' \sigma') - E(c(w' \sigma'))}{c(w')} & \text{if } c(w' \sigma') \neq 0 \\
\beta(w') \cdot p^{(n-1)}_{DA}(\sigma' \mid \text{tail}(w')) & \text{otherwise}
\end{cases}
\]

To make the distribution sum to 1, we let
\[
\beta(w') = \sum_{\sigma \in \Sigma} \frac{E(c(w' \sigma'))}{c(w')} \left( \sum_{\sigma \in \Sigma : c(w' \sigma) = 0} p^{(n-1)}_{DA}(\sigma' \mid \text{tail}(w')) \right)^{-1}.
\]

Overall, this LM is a simple, recursive model with monomial discount. We use a simple backoff scheme distinguishing only occurring and non-occurring \( n \)-grams. The discount parameters \( \rho \) and \( \gamma \) are optimized on a development set before we apply the model. For this tuning step we use heuristic grid search.

To apply our LM to a sentence, we simply multiply the conditional probabilities obtained for the various windows as usual. Let \( w = \sigma_1 \cdots \sigma_k \) be the input sentence. Then
\[
p^{(n)}_{DA}(w) = \prod_{i=1}^{k} p^{(n)}_{DA}(\sigma_i \mid \sigma_{i-k+1} \cdots \sigma_{i-1})
\]
where $\sigma_\ell = \text{NULL}$ for all $\ell \leq 0$. The same NULL-tokens are also used in the counts $c(w)$.

While the idea of the monomial discount derives from the POLKN model by Schütze (2011), which is a class-based interpolation model, we apply it without classes as a backoff model. Without the classes, our LM only relies on $n$-grams and is thus cheaper and easier to generate in ARPA format, which can be processed by SRILM. Since our goal was a tunable LM that can be used for domain adaptation in machine translation, our model needs to be compatible with a toolkit that is supported by the MOSES framework. We selected the SRILM toolkit, which can import our LM given in ARPA format. Naturally, our model can be used for different $n$-gram orders. An implementation of a class-based model or interpolation models remains future work.

### 3.3 Parameter Optimization

Our model has two discount parameters: $\rho$ and $\gamma$. As already mentioned, we use a tuning step with a small amount of development data to set those parameters. We use perplexity (Jelinek et al., 1977) as the measure of language model performance. In natural language processing, perplexity is the inverse probability of the test set, normalized by the number of words (i.e., the inverse of the geometric mean of the individual word predictions). Let $w$ be a test sentence. We assume that a language model $p$ estimates the probability $p(w')$ of each sequence $w'$. The perplexity of the LM $p$ on the test sentence $w$ is then defined to be

$$\text{Perplexity}(p, w) = \left(\frac{\sqrt[w]{p(w)}}{w}\right)^{-1}.$$  

Consequently, lower perplexity means that the LM is better at predicting the individual words in the test sentence.

Now that we have established our target function, we simply use heuristic grid search to optimize our parameters. More precisely, the parameters $\rho$ and $\gamma$ are selected from the range $(0, 1)$, and we explore the search space for the optimal discounting parameters with step-size 0.01 and map all the development set perplexities to a grid. To simplify this procedure, we start with step-size 0.1, which yields 81 settings in the straightforward way. Figures 1 and 2 show the perplexities of the development set in those 81 settings. In this way, we obtain the best general areas (brighter is better), for which we then lower the step-size to 0.01 to fine-tune the parameters. We found that even finer step-sizes (such as 0.001) have no effect on performance since the perplexities converge beforehand.

The obtained parameter space looks rather smooth, so we expect our obtained parameters to be close to optimal. We can naturally imagine more refined search methods for the ideal pa-
rameters, but we do not expect major improvements since our parameters should be almost optimal, whereas traditional methods often only yield local optima. The optimized parameters for different domains are shown in Table 1. As expected, we observe a drastic change of the parameter values for $\gamma$ comparing the in-domain scenario (English: 0.03; Chinese: 0.10; German: 0.11) to the cross-domain scenario (English: 0.59; Chinese: 0.57; German: 0.54). Thus in the cross-domain scenario, our model reserves more probability mass for the unobserved $n$-grams in comparison to the in-domain scenario and the modified Kneser-Ney models. The parameters for the German cross-domain LM are particularly large ($\rho = 0.75$ and $\gamma = 0.54$). We can only speculate that the huge difference between the training set (European Parliament proceedings) and the development set (bio-medical data) needs huge discounts to allow for many unseen $n$-grams. Surprisingly, the cross-domain parameters for English ($\rho = 0.63$ and $\gamma = 0.59$) and Chinese ($\rho = 0.64$ and $\gamma = 0.57$) are very similar. Further evaluations are necessary to detect a trend here, so at present, we do not know the significance of this observation. However, we can observe that high $\gamma$-values generally indicate a domain change between the training set and the development set.

Finally, we performed a series of experiments to establish reasonable sizes for the development set. To this end, we optimized the discount parameters for different sizes of the development set. Figures 3 and 4 show the obtained perplexity on the test set for English in relation to the size of the development set. If the development set is tiny ($\leq 10$ sentences), then we cannot find reasonable discount parameters. However, already at sizes of 20–50 sentences, we find optimal discount parameters that yield very good perplexities also on the test set. For example, in the in-domain experiment we just need 20 sentences to find the parameters $\rho = 0.6$ and $\gamma = 0.1$. With those parameters, we achieve the perplexity 92.32 on the test set, which is already better than standard SRILM, which achieves 92.42. It might be argued that our model had access to additional training examples, but adding, for example, 100 sentences of the development set to the training set for the SRILM models does not influence their perplexity (92.42) since the training data is huge (1.6 million sentences) in comparison to those 100 sentences. In summary, the very small development data does not help as additional training data, but it is enough for our model to optimize the discount parameters, which offer an alternative way to improve the performance. The same observations are true for the cross-domain experiments (and the other languages). In all cases, approximately 100 sentences are sufficient to discover good discount parameters.

### 3.4 Domain Adaption

We already mentioned that even huge in-domain training data is typically insufficient to combat cross-domain data sparseness. In addition, we have seen that adding the cross-domain development set to the training set is ineffective for small development sets. The standard solution to this problem interpolates the LM for the training set with an additional LM for the target

<table>
<thead>
<tr>
<th>Language</th>
<th>Domain</th>
<th>Training</th>
<th>Development</th>
<th>$\rho$</th>
<th>$\gamma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>in-domain</td>
<td>WSJ</td>
<td>WSJ</td>
<td>0.61</td>
<td>0.03</td>
</tr>
<tr>
<td>Chinese</td>
<td>in-domain</td>
<td>MultiUN</td>
<td>MultiUN</td>
<td>0.59</td>
<td>0.10</td>
</tr>
<tr>
<td>German</td>
<td>in-domain</td>
<td>EuroParl</td>
<td>EuroParl</td>
<td>0.65</td>
<td>0.11</td>
</tr>
<tr>
<td>English</td>
<td>cross-domain</td>
<td>MultiUN</td>
<td>NIST</td>
<td>0.63</td>
<td>0.59</td>
</tr>
<tr>
<td>Chinese</td>
<td>cross-domain</td>
<td>MultiUN</td>
<td>NIST</td>
<td>0.64</td>
<td>0.57</td>
</tr>
<tr>
<td>German</td>
<td>cross-domain</td>
<td>EuroParl</td>
<td>khreshmoi</td>
<td>0.75</td>
<td>0.54</td>
</tr>
</tbody>
</table>

Table 1: Parameters of our models. The corpora are presented in Table 2.
domain. These mixture models weight the individual LM and these weights are tuned on the development set. This approach utilizes both types of data independently, which can be beneficial. However, it requires estimating an LM for the target domain, which requires substantial training data in the target domain to be effective. In our setups, in which the target domain development set is small (few thousand sentences), this approach is ineffective since the obtained target domain LM is not useful enough. In contrast, our method only needs to optimize the discount parameters on the development data. Recall that we do not update the occurrence counts of the $n$-grams. In addition, we actually only need a very small development set (100 sentences) to optimize our discount parameters. It is known that such very small development sets do not help the other models. To confirm these statements, we ran a preliminary experiment. We did not observe any improvements using interpolation with a target domain LM trained on less than 2,000 sentences. In addition, at 2,000 sentences our model still outperforms the interpolated models in terms of test set perplexity. Moreover, the same interpolation approach can be applied to our model, and we observe the same improvements as the development set size increases beyond 2,000 sentences.

4 Experimental Setup

4.1 Corpora

We perform two experiment types: (i) language model experiments for English, Chinese, and German evaluated by perplexity as well as (ii) machine translation experiments for the two language pairs English–Chinese and English–German evaluated by BLEU (Papineni et al., 2002). We summarize the used corpora in Table 2.

For the machine translation experiments on English–Chinese, we use the special IWSLT 2011 release of the sentence-aligned MultiUN corpus of Eisele and Chen (2010) as training data. It is a multilingual parallel corpus extracted from official documents published by the United Nations from 2000 to 2009. This corpus is available in German and all 6 official languages of the United Nations. It contains roughly 300 million words per official language. We use 2 million Chinese–English sentence pairs as training data (48,933,848 English tokens and 47,222,992 Chinese tokens) from the special release provided for IWSLT 2011. For tuning and testing we use the official NIST data provided by LDC (catalog numbers LDC-2010-T10, · · · -T12, · · · -T14, · · · -T17, and · · · -T21). Note that our test sets contain multiple reference translations. The NIST data consists of Chinese news-wire documents, human transcriptions of broadcast news as well as web newsgroup documents. Obviously, the domains of the MultiUN and the NIST data are quite different.

![Figure 3: In-domain optimization.](image1)

![Figure 4: Cross-domain optimization.](image2)
Table 2: Used corpora (parts of the tuning data serve as development data for the LM).

<table>
<thead>
<tr>
<th>Corpus</th>
<th>Domain</th>
<th>Usage</th>
<th>Sentences</th>
<th>Lang.</th>
<th>Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wall Street Journal</td>
<td>news</td>
<td>training</td>
<td>1,634,529</td>
<td>English</td>
<td>39,027,486</td>
</tr>
<tr>
<td>MultiUN</td>
<td>official documents</td>
<td>training</td>
<td>8,820,000</td>
<td>English</td>
<td>215,096,536</td>
</tr>
<tr>
<td>NIST 2002, -4, -6</td>
<td>news</td>
<td>tuning</td>
<td>4,139</td>
<td>English</td>
<td>104,228</td>
</tr>
<tr>
<td>NIST 2005</td>
<td>news</td>
<td>test</td>
<td>1,082</td>
<td>English</td>
<td>139,144</td>
</tr>
<tr>
<td>NIST 2008</td>
<td>news</td>
<td>test</td>
<td>1,859</td>
<td>Chinese</td>
<td>188,402</td>
</tr>
<tr>
<td>EuroParl</td>
<td>parliament</td>
<td>training</td>
<td>1,886,260</td>
<td>English</td>
<td>50,406,502</td>
</tr>
<tr>
<td>News Commentary</td>
<td>news</td>
<td>training</td>
<td>200,112</td>
<td>German</td>
<td>5,020,146</td>
</tr>
<tr>
<td>Common Crawl</td>
<td>web</td>
<td>training</td>
<td>2,376,881</td>
<td>German</td>
<td>51,889,104</td>
</tr>
<tr>
<td>khreshmoi</td>
<td>medical, biology</td>
<td>tuning</td>
<td>500+1,000</td>
<td>English</td>
<td>10,350+21,450</td>
</tr>
</tbody>
</table>

The corresponding experiments for English-to-German use all data present in EuroParl version 7 of Koehn (2010a) as training data. EuroParl contains the proceedings of the European parliament in 21 European languages. We use three corpora as training data for the German LM: EuroParl version 7, News Commentary and Common Crawl. The News Commentary corpus contains news text and commentaries from Project Syndicate. It is provided as training data for the shared tasks offered by the workshop on statistical machine translation (WMT). The Common Crawl corpus, which was collected from web sources, was provided as a new data resource for WMT 2013. As tuning and test data we use the bio-medical data of the khreshmoi project provided by the WMT 2014 shared task. Again, the domains of the training data and tuning and test data are vastly different.

4.2 Setup

As contrastive language models we use the standard modified KN language models provided by the toolkits SRILM of Stolcke (2002) and KenLM of Heafield et al. (2013). KenLM uses a no-pruning strategy, which it compensates for with its high efficiency allowing it to handle the resulting large models. Since our model works essentially as the models in SRILM, which relies on pruning to reduce the size of the models, we select the modified KN model implemented in SRILM as baseline. We currently employ the same pruning strategy as SRILM, so our models are small compared to models of KenLM and have essentially the same size as the standard SRILM models. It remains to be seen whether the reported advantages can also be obtained using a no-pruning strategy as in KenLM together with our model. For completeness’ sake, we also report scores for other models.

All systems are used to generate 5-gram language models in ARPA format. We use the full monolingual data available in the training corpus (e.g., 8.8 million English sentences from
Table 3: Perplexity and size of the improved Kneser-Ney, interpolated modified Kneser-Ney, backoff version of modified Kneser-Ney and our models on the in-domain data.

<table>
<thead>
<tr>
<th>Toolkit</th>
<th>Model</th>
<th>Smoothing Method</th>
<th>Size in GB</th>
<th>English Perplexity</th>
<th>Chinese Perplexity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Dev. Test</td>
<td>Dev. Test</td>
</tr>
<tr>
<td>IRSTLM</td>
<td>improved KN</td>
<td></td>
<td>208.6</td>
<td>102.54 102.44</td>
<td>179.2 92.49 94.67</td>
</tr>
<tr>
<td>KenLM</td>
<td>interpolated mKN</td>
<td></td>
<td>621.2</td>
<td>91.41 91.53</td>
<td>483.4 82.89 86.11</td>
</tr>
<tr>
<td>SRILM</td>
<td>backoff mKN</td>
<td></td>
<td>217.7</td>
<td>94.38 94.34</td>
<td>187.2 84.08 85.31</td>
</tr>
<tr>
<td>SRILM</td>
<td>interpolated mKN</td>
<td></td>
<td>217.8</td>
<td>92.48 92.42</td>
<td>187.0 82.52 83.95</td>
</tr>
<tr>
<td>SRILM</td>
<td>our</td>
<td></td>
<td>217.2</td>
<td>92.37 92.32</td>
<td>187.1 83.00 84.58</td>
</tr>
</tbody>
</table>

MultiUN). Our language model is implemented as a variant of SRILM that implements the different discounting. Our implementation is available on the homepage of the first author (JUNFEI GUO). As mentioned earlier, we use heuristic grid search with step size 0.01 during tuning to discover the optimal discount parameters (see Section 3.3) for our model. An illustration of the results of such a search is presented in Figure 1 and 2.

All the machine translation experiments use the MOSES framework of Koehn et al. (2007). It offers support for phrase-based and hierarchical phrase-based translation models and contains all tools needed to train and execute these models. In particular, it supports the ARPA format of our language models. The word segmentation of the Chinese sentences was achieved with the Stanford Word Segmenter of Chang et al. (2008). GIZA++ (Och and Ney, 2003) with the heuristic grow-diag-final-and (Koehn et al., 2005) was used to obtain the word alignments. All the translation models were trained on approximately 1.8 million parallel sentences after standard length-ratio filtering. They were tuned using MERT (Och, 2003) on their respective tuning sets using BLEU (Papineni et al., 2002) as score, which is also the score that we report for the test sets. Finally, the pairwise bootstrap resampling method of Koehn (2004) is used for significance testing.

5 Language Model Perplexity Experiments

First we evaluate the various language models in isolation using perplexity (Jelinek et al., 1977). Since the new feature of our model is the ability to tune the discount parameters, we perform two types of experiments: in-domain and cross-domain. In the in-domain experiments, the tuning and test data are similar to the training data, whereas in the cross-domain scenario the tuning and test data are still similar, but different to the training data. Obviously, we focus on cross-domain experiments since we expect our model to perform well there. A summary of the obtained results (together with the model sizes) is presented in Tables 3 and 4 for the in-domain and the cross-domain scenario, respectively.

5.1 In-domain

The first experiment investigates the performance of the different language models on in-domain (news) data from the Wall Street Journal. The training set contains more than 1.6 million sentences and both the validation and the test set have roughly 100,000 sentences. We use the same number of sentences for Chinese from the MultiUN corpus. Table 3 shows the performance of the language models (measured by perplexity) together with their size. The IRSTLM models, which are simplified versions of the improved KN model, are the smallest in size, but have the highest (i.e., worst) perplexity. The large unpruned KenLM models have the lowest perplexities for English, but at the expense of significantly larger sizes. The SRILM models using backoff
Table 4: Perplexity and size (in GB) of the models on the cross-domain data.

<table>
<thead>
<tr>
<th>Toolkit</th>
<th></th>
<th>ENGLISH</th>
<th></th>
<th>CHINESE</th>
<th></th>
<th>GERMAN</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Perplexity</td>
<td>Size</td>
<td>Perplexity</td>
<td>Size</td>
<td>Perplexity</td>
<td>Size</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dev.</td>
<td>Test</td>
<td>Val.</td>
<td>Test</td>
<td>Dev.</td>
<td>Test</td>
</tr>
<tr>
<td>KenLM</td>
<td>9.40</td>
<td>296.51</td>
<td>317.94</td>
<td>11.26</td>
<td>950.07</td>
<td>840.15</td>
<td>10.20</td>
</tr>
<tr>
<td>SRILM</td>
<td>2.20</td>
<td>289.92</td>
<td>312.30</td>
<td>2.20</td>
<td>729.40</td>
<td>639.42</td>
<td>1.29</td>
</tr>
<tr>
<td>our</td>
<td>2.19</td>
<td>271.20</td>
<td>286.53</td>
<td>2.20</td>
<td>669.16</td>
<td>584.79</td>
<td>1.27</td>
</tr>
</tbody>
</table>

(without interpolation) score consistently worse than those using interpolation, which was already observed by Chen and Goodman (1996). Our model (without interpolation) outperforms the SRILM interpolated models for English (92.48 vs. 92.37 and 92.42 vs. 92.32 on the development and test set, respectively), but is beaten by the KenLM models (91.41 vs. 92.37 and 91.53 vs. 92.32 on the development and test set, respectively). Overall, the differences between these models are rather small. In the Chinese experiments we observed similar performances. IRSTLM models are again the worst in terms of perplexity, and our model performs slightly worse than SRILM models with interpolation but always better than SRILM models with back-off only. Overall, these results suggest that in-domain our monomial discount model achieves the same performance as the modified KN models implemented in SRILM when using interpolation. Consequently, in all other experiments we use SRILM models with interpolation, which is also recommended for use in MOSES.

5.2 Cross-domain

For the cross-domain experiments we use three languages: English, Chinese, and German. The results are reported in Table 4. Before we discuss the results, let us quickly describe the experiments (see Table 2).

- For the English experiment, we train the language models on the English data of the MultiUN corpus of Eisele and Chen (2010), which contains roughly 8.8 million sentences. For the cross-domain evaluation, we use NIST data, which includes news-wire, broadcast news, and web data, so it is quite different (in style and language) from the contract documents contained in MultiUN.

- For the Chinese experiment, we use the same resources, but now the Chinese data contained in those corpora.

- For the German experiment, which we did in order to cover a morphologically rich language, the language models are trained on EuroParl version 7, News Commentary, and the Common Crawl corpus (overall 4.4 million sentences). We use the khreshmoi data for development and test. The data in khreshmoi were sampled from summaries of English medical documents.

Comparing the perplexities reported in Tables 3 and 4, we immediately observe that they increase from \( \leq 100 \) to \( \geq 200 \) (sometimes a lot more), which shows that the cross-domain development and test data is rather different from the training data. The results in Table 4 indicate that our model can achieve considerable perplexity improvements for cross-domain data. While our models retain the size of the models generated by SRILM, we often improve the perplexity (English: from 312.30 to 286.53; Chinese: from 639.42 to 584.79; and German: from 501.28 to 469.78). For all experiments, the perplexities computed for the development and the test set are similar because we chose similar validation and test sets. Overall, in all performed experiments, our model outperforms both the modified KN model in SRILM and KenLM (both
using interpolation). The models produced by KenLM are generally much larger (> 9 GB) than the models produced by SRILM or our variant (< 3 GB). We report the perplexity results for KenLM in this experiment since KenLM models are very popular in machine translation. Since KenLM does not prune, the KenLM models have larger vocabularies, which can be both beneficial and harmful. This might be an explanation for the poor perplexities that the KenLM models achieve. The tuning of the discount parameters in our model on the cross-domain development set seems to help our model adapt well to the new domain. Together with the results from the in-domain experiment, we can conclude that our model seems to perform as well as SRILM on in-domain data and outperforms SRILM on cross-domain data. The improvements are more pronounced the more distant the development and test data is from the training data.

### 6 Machine Translation Experiments

Following our LM perplexity experiments, we also want to confirm that the theoretical advantage that our model enjoys in terms of perplexity translates into an application area. Here we select statistical machine translation as an application, so we want to confirm that systems using our model achieve better BLEU-scores (Papineni et al., 2002) in a variety of translation tasks. We compare the different language models on both the phrase-based translation models [PBMT] by Zens et al. (2002) and the hierarchical phrase-based translation models [HPBMT] of Chiang (2005). Both types of translation models are implemented in MOSES toolkit of Koehn et al. (2007). The results of our evaluation are reported in Table 5.

For the Chinese-to-English experiments, we use roughly 2 million sentence pairs from the MultiUN corpus and the tuning and test data consists of the classical NIST data. Overall, the models supported by KenLM achieved the best BLEU scores and significantly beat the SRILM-based baseline, but they do not significantly outperform the models supported by our new language model. Together with our new language model, both the phrase-based and the hierarchical phrase-based models significantly outperform the SRILM-based baselines (from 20.05 to 20.35 for PBMT and from 20.64 to 20.95 for HPBMT). In this experiment, our improvement in terms of perplexity compared to the SRILM-based baseline translates well into an advantage in BLEU-score. This is not true for the perplexity advantage compared to KenLM-based models, which achieve even (insignificantly) better BLEU-scores despite worse perplexity.

For the experiments translating English to Chinese, we use the same training data, but the NIST 2008 test data, which has multiple Chinese references. The results (see Table 5) show a similar picture with one exception. The phrase-based model did not benefit from KenLM and achieves the same performance as the SRILM-based model. Otherwise, KenLM-based models and models based on our new language model achieve similar performance and both significantly outperform the SRILM-based baseline. For our models, the scores consistently improve (from 17.04 to 17.41 for PBMT and from 17.68 to 17.94 for HPBMT). Due to the particularity already mentioned, we even significantly outperform the KenLM-based phrase-based model in this task. So far, our perplexity improvements consistently yielded improvements in translation.

### Table 5: BLEU-scores for the various translation experiments. Stars indicate significant improvements over the baseline SRILM (at confidence level 95%).

<table>
<thead>
<tr>
<th>Language model</th>
<th>CHINESE → ENGLISH</th>
<th>ENGLISH → CHINESE</th>
<th>ENGLISH → GERMAN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PBMT</td>
<td>HPBMT</td>
<td>PBMT</td>
</tr>
<tr>
<td>KenLM</td>
<td>20.59*</td>
<td>21.12*</td>
<td>17.06</td>
</tr>
<tr>
<td>SRILM</td>
<td>20.05</td>
<td>20.64</td>
<td>17.04</td>
</tr>
<tr>
<td>our</td>
<td>20.35*</td>
<td>20.95*</td>
<td>17.41*</td>
</tr>
</tbody>
</table>
quality when measured by BLEU.

Finally, we run experiments translating English to German. In this case, the training data is EuroParl and the tuning and test sets are from the WMT 2014 bio-medical data khreshmoi. The results of Table 5 show minute differences, of which none are significant. In this machine translation task, we observe no significant improvements in translation quality (measured by BLEU) even though we observed sizable LM improvements in terms of perplexity. The huge difference between the training set (European Parliament proceedings) and the test set (bio-medical data) might be the reason. The overall translation quality is very poor and potentially too poor to yield reasonable translations, which allows us to speculate that the impact of the language model might be minimal in this setup.

Overall, we demonstrated that our new language model does not harm the translation quality, but rather offers significant improvements in a number of cases. However, the improvements in terms of perplexity do not necessarily translate into BLEU-score improvements. Nevertheless, we often significantly outperformed SRILM-based models in cross-domain evaluations, which shows a nice benefit of our new language model.

7 Summary

In this paper, we introduced a tunable language model which can easily be tuned to new domains and is thus ideally suited for domain adaptation. Perplexity shows that our model outperforms the baseline model especially in domain adaptation scenarios. We implemented our model as a new language model in the MOSES statistical machine translation framework and evaluated it in machine translation task. Also there we observed significant improvements.

In future work we plan to improve the parameter optimization algorithm and implement our model with interpolation. We would also like to investigate translation from German to English and apply our model to other morphologically rich target languages.

Acknowledgments

JUNFEI GUO and ANDREAS MALETTI gratefully acknowledge the financial support by the German Research Foundation (DFG) grant MA / 4959 / 1-1. JUNFEI GUO acknowledges the support by Chinese Scholarship Council (CSC) during his PhD studies at the University of Stuttgart. All authors want to sincerely thank the colleagues at the University of Stuttgart and anonymous reviewers for their helpful comments.

References


