

Mining the Web for Domain-Specific Translations

Jian-Cheng Wu¹ Peter Wei-Huai Hsu² Chiung-Hui Tseng³ Jason S. Chang⁴

Department of Computer Science¹³⁴

Institute of Information Systems
and Applications⁴

National Tsing Hua University National Tsing Hua University
101 Kuang Fu Road, HsinChu, Taiwan 101 Kuang Fu Road, HsinChu, Taiwan
{wujc86, hsn97526, smilet, jason.jschang}@gmail.com

Abstract

We introduce a method for learning to find domain-specific translations for a given term on the Web. In our approach, the source term is transformed into an expanded query aimed at maximizing the probability of retrieving translations from a very large collection of mixed-code documents. The method involves automatically generating sets of target-language words from training data in specific domains, automatically selecting target words for effectiveness in retrieving documents containing the sought-after translations. At run time, the given term is transformed into an expanded query and submitted to a search engine, and ranked translations are extracted from the document snippets returned by the search engine. We present a prototype, *TermMine*, which applies the method to a Web search engine. Evaluations over a set of domains and terms show that *TermMine* outperforms state-of-the-art machine translation systems.

1 Introduction

Increasingly, phrases, passages, and Web pages are being translated using desktop software (e.g., *Systran*) or Web-based services (e.g., *Yahoo! Babel Fish*¹ and *Google Translate*²). These texts usually contain domain-specific terms, which may not be handled properly by a general-domain machine translation system. State-of-the-art machine translation systems typically use a general bilingual dictionary either manually compiled or learned from a parallel corpus. However, such dictionaries may not have sufficient coverage of

domain-specific information, leading to ineffective handling of terms that have domain-specific translations (DST), or simply out of vocabulary (OOV).

It is difficult to compile, manually or automatically, a bilingual dictionary with comprehensive coverage of words, domains, and domain-specific translations with existing parallel corpora, which are limited in quantity and domain variety. These domain-specific translations could be obtained more effectively, if we search the Web and extract the translations from domain-specific mixed-code Web pages. Consider the following sentence:

- (1) Options are financial instruments that convey the right, but not the obligation, to engage in a future transaction on some underlying security, or in a futures contract.”

A typical machine translation system (i.e., *Google Translate*) probably translates this domain-specific sentence into Sentence (2), where “option” is translated as “選擇” and “security” as “安全性.”

- (2) 選擇的金融工具，賦予權力但沒有義務，在未來從事交易潛在的安全性或期貨合約。

However, the best way to translate Sentence (1) probably involves using FINANCE-related translations, “選擇權” and “證券.” These DSTs may be found in domain-specific mixed-code Web pages. Intuitively, by requiring a search engine to retrieve documents containing the source term (e.g., “security”) and some target-language (TL) FINANCE keywords (e.g., “市場”, or “價格”), we can bias the search engine towards returning top-ranking snippets of mixed-code documents such as Snippets (3) and (4):

- (3) 科技產業提供員工股票選擇權 (stock option) 或分紅配股 ...
- (4) 台灣證券市場股票價格 ... The Information Content of Security Prices in Taiwan ...

¹ <http://babelfish.yahoo.com/>

² http://www.google.com/language_tools?hl=en

We present a new terminology translation system, *TermMine*, that automatically learns to find domain-specific term translations. *TermMine* uses an unsupervised training method to learn effective query expansions (QE) automatically during training by analyzing a set of terms for each domain of interest, and extracts indicative target words for each individual domain. For example, *TermMine* learns that “市場” and “價格” are important TL keywords for the FINANCE domain because it occurs frequently in mixed-code snippets returned by the search engine for many FINANCE terms submitted as query. We describe the training process in more details in Section 3.

At runtime, *TermMine* accepts a term and domain as input, and transforms the given term into an expanded query. The query is submitted to a search engine to retrieve mixed-code documents. After that, *TermMine* extracts candidate translations in the returned document snippets and ranks them according to surface patterns, frequency, and distance from the given term. In our prototype, *TermMine* returns the term translations to the user directly; alternatively, these term translations can be used as additional input to a traditional machine translation system. Some statistical machine translation tools, such as Moses (Koehn et al, 2007), accept the source sentence with preselected word translation candidates. In this way, we may be able to use DSTs to improve the performance of MT systems.

The rest of the paper is organized as follows. We review the related work in the next section. Then we present our method for automatically learning to expand a given term into an effective query for a domain (Section 3). We describe the experiments carried out to assess the proposed method. As part of our evaluation, we compare the quality of the translations by *TermMine* against a state-of-the-art machine translation system and other IR-based term translation methods (Section 4). Finally, we summarize our approach and point out future research directions (Section 5).

2 Related Work

Machine translation (MT) has long been an active research area. The traditional machine translation researches are discussed in the survey of MT history by Hutchins (1995). More technical survey on machine translation can be found in Dorr et al.

(1993), including the statistical approach pioneered by the IBM group (Brown et al. (1990); Brown et al. (1993)). Lopez (2006) provided a more up-to-date survey of state-of-the-art statistical machine translation approach, including methods that use comparable and Web corpora. In our work, we address an aspect of machine translation and machine-assisted translations, emphasizing using the Web as corpus. We consider a subproblem in machine translation where the goal is to find domain-specific translations for a given term.

More specifically, we focus on the part of translations of technical terms via Web search, namely, retrieving promising passages that are likely to contain translations of a given term, extracting, and ranking the translations. Lexical translation has been a popular topic of statistical machine translation (SMT) research. Traditional SMT approaches hinge on automatically learning a lexical translation model from a parallel corpus. The lexical translation model provides candidates for translating individual words, and subsequently the word translation ambiguity is resolved by using an n-gram model of the target language. Brown et al. (1993) described how to automatically align words and translations in the parallel sentences and build word-based statistical models. While SMT systems establish a general translation model from a parallel corpus, we propose a method for learning to find domain-specific translations via Web search.

Parallel corpora are considered better data sources for quality translation information but are limited by lower availability. As an effort to cope with the data sparseness problem, Fung and Yee (1998) advocated using comparable corpora, which are considered more readily available in large quantity. Shao and Ng (2004) described a similar method for extracting new word translations from comparable corpora. Munteanu et al. (2004) also described a system that extracts parallel sentences from comparable corpora as additional training data for SMT systems.

Recent research has begun to emphasize phrase translation to improve on word-based SMT approach. Cao and Li (2002) proposed an approach for translating short, base noun phrases based on a bilingual dictionary. The translation candidates for words in a phrase are combined and validated using Web page counts. Koehn and Knight (2003) described a system that builds a noun phrase-based translation subsystem leading to further

improvement of phrase-based machine translation systems. Koehn (2004) introduced the phrase-based machine translation approach based on phrase-to-phrase translations induced from word alignment information. The phrase-based approach significantly improves the translation quality. However, in the situations of single-word noun phrases, phrase-based approach has problem producing correct translation. In contrast, we propose to use domain information, provided by the user or automatically derived from the (sentential or paragraph) context, to help find more appropriate translations.

Round the same time, researchers began to turn to the Web and proposed new methods for translating phrases. In a study more closely related to our work, Nagata et al. (2001) introduced a system for extracting the English translations of a given Japanese technical term by collecting and scoring translation candidates co-occurring with the given term in mixed-code Web pages. In addition to using Web texts, Lu et al. (2002) proposed a new method that uses anchor texts and hyperlink structure to find term translations for cross-language information retrieval. More recently, Wu et al. (2005) introduced a method for learning source-target surface patterns to find translation of proper names and technical terms on the Web. Our setting, approach, and evaluation are substantially different from other Web-based translation approach.

More recently, Web-based term translation systems have begun to expand queries to increase the chance of retrieving snippets that contain translations. Huang et al. (2005) proposed a pseudo relevance feedback approach for improving search results by augmenting the second round query with translations of high-frequency words found in the first-round returned snippets. Similarly, Su (2006) presented a method for automatically expanding the queries by augmenting the query with the high-frequency TL words in the snippets returned in the first round. Another recent method presented by Wu and Chang (2007) also expands queries based on cross-language phonemic relationships learned from a set of training data, in order to find transliteration of a given name on the Web. Both Su (2006) and Wu and Chang (2007) perform query expansion on a query-by-query basis. The presented method attempts to learn in advance a set

of domain-specific TL keywords for a set of query terms in the same domain.

In contrast to the previous research in machine translation of sentences and phrases, we present a method that automatically learns query expansions with the goal of maximizing the probability of a Web search engine returning snippets that contain domain-specific translations of a given term. We mine the Web to exploit the vast amount of Web data for finding domain-specific translations.

3 The *TermMine* System

General translation systems usually do not work well in translating domain-specific terms. Those systems typically return general translations (unless the term is a multiword phrase encountered in the training phrase). Unfortunately, such translations may not be what the user expects, or fit well with the sentential context containing the term. To obtain domain-specific translations, a promising approach is to bias the search engine to return snippets of domain-specific mixed-code Web document that are likely to contain the given term and the sought-after translations.

3.1 Problem Statement

We focus on the subproblem of the machine translation: extracting domain-specific translations for a given term. The returned translations can be examined by a user directly, or passed on to the sophisticated decoding model of a translation system (e.g., a phrase-based statistical machine translation as described in Koehn (2004)). It is crucial that the translations are relevant to the given domain. At the same time, several alternative translations should be extracted. Therefore, our goal is to return a small set of translations that are appropriate for the given term under the given domain. We now formally state the problem that we are addressing.

Problem Statement: We are given a domain-specific source language (SL) term T under the domain D and a general-purpose information retrieval system SE that operates over a mixed-code documents collection (e.g., the Web). Our goal is to extract domain-specific translations for T from the collection via SE . For this, we transform T into a new query with a set of target-language (TL) keywords, w_1, \dots, w_m , which are related to D . We submit the expanded query to SE and extract

domain-specific translations of T in the snippets returned by SE .

In the rest of this section, we describe our solution to this problem. First, we show how to learn a set of TL domain keywords using a set of SL terms for each domain (Section 3.2). After that, we show how *TermMine* accepts a term and domain from the user, expands it into a new query, and extracts domain-specific translations from the Web snippets returned by SE (Section 3.3).

- | |
|--|
| (1) Generating and Selecting Terms for Training from a Terminology Bank (Section 3.2.1)
(2) Generating and Filtering Candidate Keywords from Selected Training Data (Section 3.2.2)
(3) Weighting and Ranking Candidates to Determine the Sets of Keywords (Section 3.2.3) |
|--|

Figure 1. Outline of the process used to train *TermMine*.

3.2 Learning Domain Keywords for QE

We attempt to learn transformations to convert the given term into effective queries for domain-specific translations. The transformations essentially consist of TL words expected to appear in domain-related snippets from the Web. In this subsection, we first define a strategy of learning a set of TL keywords for each domain. Figure 1 shows the learning process.

The strategy relies on a training set of domain-specific SL terms. We then describe how to select terms for best training effect (Section 3.2.1). After that, we give the procedure of generating, filtering, and ranking TL keywords (Section 3.2.2 and 3.2.3).

3.2.1 Selecting Training Terms

In the first stage of the training process, we describe how to select domain source terms for training. The input to this stage is a set of monolingual terms for each domain under consideration. As we will describe in Section 4.1, we selected 100 terms from a list of topics for each domain in Wikipedia³ for this purpose. Some example domains and SL terms are shown in Figure 2.

From the original lists, we select multi-word, medium-frequency terms, that are more representative and specific. Singleton terms (e.g., “port”) tend to be ambiguous, conveying different meanings in different domains (e.g., COMPUTER

and GEOGRAPHY). An ambiguous term tends to co-occur with words relevant to many domains, thus defeating the purpose of learning keywords for a specific domain.

We choose not to use sophisticated techniques to filter out ambiguous terms. For simplicity, we just use the multi-word SL terms, which are considerably less ambiguous.

The output of this stage is sets of domain-specific terms that produce better training results. Figure 3 shows some example terms, automatically selected from the original lists.

Domain Name	Example Terms
COMPUTER	<i>computer engineering, data structures, port, loader, option</i>
FINANCE	<i>capital market, common stock, personal finance, option, capital</i>
BIOLOGY	<i>memory cell, structural biology, culture, fitness, vaccine</i>
GEOGRAPHY	<i>natural resource, mountain range, population density, culture, port</i>

Figure 2. Example domains and terms.

Domain Name	Selected Terms
COMPUTER	<i>computer engineering, computer file, data structures</i>
FINANCE	<i>personal finance, common stock, capital market</i>
BIOLOGY	<i>memory cell, structural biology, population genetics</i>
GEOGRAPHY	<i>natural resource, mountain range, population density</i>

Figure 3. Example domains and terms selected.

Although alternative approaches can be used to generate TL keywords—using more source terms (thousands instead of one hundred), or additional information of translations (e.g., “記憶細胞” for “memory cell” in the BIOLOGY domain)—our method does have some advantages. The approach only requires a small amount of monolingual SL terms, making it easier to obtain the required data for new or fine-grained domains for which it may not be easy to find a lot of technical terms and translations. Additionally, since the method does not rely on translations, it is easily to port to other target languages (e.g., finding term translations in Mandarin or Japanese).

³ <http://www.wikipedia.org/>

3.2.2 Generating and Filtering Keywords

In the second stage of the training process, we generate candidate TL keywords that are potentially effective as query expansion (QE) terms. We apply a filtering procedure to rule out ineffective TL words from a set of candidate keywords CK , generated for each domain based on the source terms obtained in the first stage. Figure 4 shows the procedure for generating CK for each domain D , consisting of several steps—retrieving Web mixed-code snippets, word segmentation, calculating features, ranking and selecting TL words. Each step of the algorithm is described below in detail.

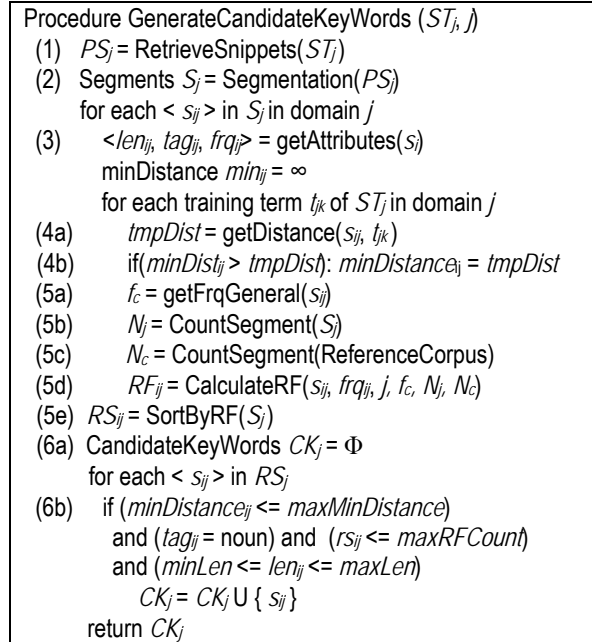


Figure 4. Algorithm for generating candidate keywords

We begin in Step (1) with a set of selected terms ST for each domain D obtained from the previous stage. In Step (2), a list of TL words is obtained from the snippets returned for queries, consisting of ST for each domain. Each word is therefore associated with the relevant domain.

Once we have a list of TL words as candidate keywords for each domain, we calculate their features, including length, POS tag, and frequency in Step (3). To obtain more effective TL words to bias SE towards returning domain-relevant search results, we use a simple heuristic that worked well in our training experiments. We observed that nouns are much more likely to be domain-specific

than other parts of speech, especially in the range of length from $minLen$ to $maxLen$. Additionally, for each noun, we calculate the minimal distance $MinDistance$ between the noun and the relevant source term (Steps (4a) and (4b)). An upper bound $maxMinDistance$ is set for the purpose of discarding ineffective keywords. Typically, a word that never appears near to the source term is unlikely to be relevant to the domain under consideration.

In addition to the distance feature, we also calculate the frequency aiming to filter out candidate keywords that appear relatively less frequently in the returned snippets. We calculate relative top $maxRFRank$ candidates. For that purpose, in Step (5a) to (5c), we calculate the occurrence counts of words in returned snippets and their counts in general reference corpus. In Step (5d), we use these counts to calculate relative frequencies:

$$RF_{ij} = f_{ij} - N_j * (f_c / N_c) \quad (1)$$

where RF_{ij} is the relative frequency of segment s_{ij} with absolute frequency f_{ij} in the snippets of size N_j , and f_c is the frequency of s_{ij} in the general corpus of size N_c . In Step (5e), we sort the segments in a decreasing order of RF . Intuitively, this frequency-based feature is effective in finding the keywords in the TL from the snippets.

Finally in Step (6a), we set some limitations to help filter out ineffective candidate keywords. Sets of CK are initialized as empty for each domain. In Step (6b), we check and retain candidate keywords CK_j that satisfy all filtering constraints for the domain D_j .

Example keywords generated after this step is shown in Figure 5. As shown in the figure, the filtering in general does a reasonable job in keeping TL words that are closely related to the intended domain. These keywords are not the direct translations of the source terms, but rather typical domain-specific keywords that occur in the vicinity of their translations. Recall that effective query expansion requires a set of keywords that are written in the TL and reflect broadly the domain, yet the number of keywords should be small enough as not to overburden the search engine. Therefore we still need a further stage to filter and rank the keywords.

Domain	Candidate Keywords
COMPUTER	軟體 系統 網路 電腦 資料 教科書 物件 原文 標題 使用者 模式 概念

Figure 5. Sample CK for the COMPUTER domain.

3.2.3 Ranking Candidate Keywords for QE

At the final stage of the training process, we cut down further the number of keywords for each domain, because typical search engines only allow for a limited number of keywords in a query. We assess the effectiveness of candidate keywords and retain a small number of candidates. Figure 6 presents the algorithm for ranking and selecting a set of candidates for QE keywords.

We calculate a score for each candidate key ck under domain D_j based on the concept of *entropy* in the information theory. In order to apply the concept of entropy, we need a notion of probability—how the probability of finding ck_{ij} in PS_j for a particular term ST_j . The entropy as a measure of uncertainty or information content corresponds intuitively how the instances of ck_{ij} spread across the snippets for all source terms under the domain D_j .

Procedure DetermineKeyWords (CK, PS_j)	
for each candidate keywords ck_{ij} for domain D_j	
$Spread_{ij} = 0$	
$N_i = 0$	
for each ps_{jk} of PS_j in Domain D_j	
(1a)	$f_{ijk} = \text{getFrqInSnippet}(ck_{ij}, ps_{jk})$
(1b)	$N_{jk} = \text{getSegNumInSnippet}(ps_{jk})$
(1c)	$N_i = N_i + f_{ijk}$
(1d)	$p_{ijk} = f_{ijk} / N_{jk}$
(1e)	$Spread_{ij} = Spread_{ij} - p_{ijk} * \log_2(p_{ijk})$
$Concentrate_i = 0$	
for each domain D_j	
(2a)	$f_{ij} = \text{getFrqNumInDomain}(ck_{ij}, D_j)$
(2b)	$p_{ij} = f_{ij} / N_i$
(2c)	$Concentrate_i = Concentrate_i - p_{ij} * \log_2(p_{ij})$
(3)	$Score_{ij} = Spread_{ij} / Concentrate_i$
(4)	Sort items in CK_j in decreasing order of $Score_{ij}$
(5)	Return the top $numKeywords$ candidate keywords

Figure 6 Algorithm for determining QE keywords.

Therefore, in addition to measure spread across all the source terms in a domain, we also need to measure spread across all domains under consideration. For our purpose, the entropy associated with the source terms within a domain need to be high so as to make ck_{ij} applicable for all

terms under a domain, while the entropy associated a keyword across all domains should be low so as to make ck_{ij} effective in query expansion for this particular domain.

Consider two keywords “基金” and “內容,” both of which spread evenly across the terms under the FINANCE domain. However, the occurrences of “基金” concentrate more in the FINANCE snippets, while the occurrences of “內容” scatter across snippets for difference domains such as COMPUTER and LITERATURE. We would like to keep “基金” and discard “內容” as keywords for QE for the FINANCE domain.

Hence, we calculate two entropy-based scores *Spread* and *Concentrate* for each candidate keyword ck_{ij} . The score *Spread* measures how the instances of ck_{ij} spread out in snippets for the all source terms under a domain, while *Concentrate* shows how instances of ck_{ij} concentrate in snippets for a few domains. *Spread* and *Concentrate* for ck_{ij} is defined as:

$$P_{ijk} = f_{ijk} / N_{jk}, \text{ and } P_{ij} = \sum_k f_{ijk} / N_i \quad (2)$$

$$Spread_{ij} = -\sum_k P_{ijk} \text{Log}_2 P_{ijk} \quad k \text{ in } PS_j \text{ for } D_j, \quad (3)$$

$$Concentrate_{ij} = -\sum_i P_{ij} \text{Log}_2 P_{ij} \quad j \text{ in } D, \quad (4)$$

where f_{ijk} is the frequency of ck_{ij} in the k -th snippet, N_{jk} is total frequency of ck_{ij} in all snippets PS_j for domain D_j , and N_i is total frequency of ck_{ij} in all snippets for all domain D . Subsequently, each ck_{ij} will be ranked by these two entropy values. Obviously there are at least two ways of combine these two scores to filter keywords:

$$Spread_{ij} / Concentrate_{ij} \quad (5)$$

$$Spread_{ij} - Concentrate_{ij} \quad (6)$$

After close examination, we found that division is more useful.

With that in mind, we calculate the frequency f_{ijk} of ck_{ij} occurring in each ps_{jk} for Domain D_j (Step (1a) in Figure 6), and N_{jk} , the total number of word segments in ps_{jk} (Step (1b)). In Step (1d), the generated probability p_{ijk} are involved for calculating entropy values.

In Steps (2a) and (2b), we calculate these frequency counts and related probabilities. In Steps (1e) and (2c), we use the probabilistic values to calculate *Spread* and *Concentrate* values. In Step

(3), we calculate the ratio of $Spread_{ij}$ and $Concentrate_i$ scores to filter keywords. In Steps (4) and (5), we rank ck_{ij} by $Score_{ij}$ and return the top $numKeywords$ keywords as the final result. Therefore, the output contains a set of $numKeywords$ keywords for each of domains.

3.3 Translation Extraction

Recall that the main purpose of the research is to learn keywords for domain-specific query expansions in order to find and extract domain-specific translations. For the last step of extracting translations, we adopt the translation extraction model proposed by Wu et al. (2005). One of the reasons is that the method performs reasonably well in extracting general translations. Additionally, the method does not expand the query, making it easy to incorporate our query expansion approach.

Here, we briefly describe the model proposed by Wu et al. (2005). They learn source-target surface patterns from a set of training data and use those surface patterns to extract translation candidates from the returned summaries, and subsequently rank candidates based on statistical information like occurrence frequency, and position between source query and translation candidates in returned mixed-code pages.

4 Experiments and Setting

In this section, we first describe the implementation and training process of the proposed method (Section 4.1). Then, we describe the systems being evaluated and compared (Section 4.2). We describe the data used in the evaluation (Section 4.3), and report the evaluation results (Section 4.4).

4.1 Training *TermMine*

We used a set of 26 domains, with each domain containing up to several hundreds of English terms for training or testing. We obtained the domains and terms from the “List of Topics” pages in Wikipedia, a free, multilingual, open content project operated by the non-profit Wikipedia Foundation. Figure 7 shows all domains tested.

We select 100 terms from each domain for training *TermMine*. We performed some experiments with the different values of various system parameters, the resulting parameters are shown in Figure 8. We have not tested all

parameters exhaustively and completely. If a parameter has been tuned in some testing and observations, we would have a brief explanation in the figure.

4.2 Term Translation Systems Compared

We compared the results produced by *TermMine* under different settings of query expansion and relevancy feedback. Additionally, we also compared these results against the results produced by a state-of-the-art machine translation system. The systems being evaluated and compared are:

- (1) ***TermMine QE-*** (Baseline): The system employs the approach proposed by Wu et al. (2005) to extract translations based on surface patterns. No query expansion is performed.
- (2) ***TermMine***: The proposed method described in Section 3.
- (3) ***Web-based Relevance Feedback (WRF)***: The system processes the input the same way as in System *TermMine QE-*. After that it uses TL keywords in the snippets to expand query and perform another round of search as proposed in (Su, 2006).
- (4) ***TermMine+***: This system proceeds as *TermMine* in the first round. After that, pseudo relevance feedback is performed.
- (5) ***Google Translate***: An on-line translating system developed by Google.

Huang et al. (2005) and Su (2006) propose to use *Pseudo relevance Feedback* (PRF) as the query expansion strategy to enhance the performance of finding translation on the Web. The method involves selecting keywords in the returned snippets by a search engine to perform QE for the second round of search. In our experiment, we select TL word tokens based on frequencies and distances from the given term. We select the top 10 of the most frequent words that appear near the given term for QE. Some example keywords and domains are shown in Figure 9.

<i>law</i>	<i>economics</i>	<i>management</i>
<i>sports</i>	<i>chemistry</i>	<i>mathematics</i>
<i>music</i>	<i>geography</i>	<i>literary</i>
<i>biology</i>	<i>marketing</i>	<i>health science</i>
<i>finance</i>	<i>linguistics</i>	<i>physical science</i>
<i>history</i>	<i>agriculture</i>	<i>communication</i>
<i>nutrition</i>	<i>philosophy</i>	<i>computer science</i>
<i>sociology</i>	<i>architecture</i>	<i>mechanical engineering</i>
<i>education</i>	<i>psychology</i>	

Figure 7. The list of domains.

Parameters/values	Description and tuning
numDomains = 26	Number of domains under consideration
maxMinDistance = 5	The Min. distance between a segment and training terms
minRFRank = 200	Minimal rank by RF for selecting candidate keywords. Values selected by testing 50, 100, 150 and 200
minLen = 2	Minimal length of TL keyword in characters, selected by testing values from 1 to 10
maxLen = 3	Maximal length of TL keyword in characters, selected by testing values from 1 to 10
numKeywords = 10	Number of QE keywords produced, selected by considering the limitation of common search engines

Figure 8. Training parameters for *TermMine*.

Domain	Target Language Keywords
<i>History</i>	哲學 帝國 歷史 時代 社會 遊戲 文化 著作 時期 戰爭
<i>Biology</i>	細胞 基因 蛋白質 生物 植物 化學 病毒 分子 生態 物種
<i>Finance</i>	股票 利率 債券 基金 金融 證券 現金 財務 報酬 率 選擇權
<i>Literature</i>	小說 主義 文學 戲劇 人物 詩人 喜劇 史詩 運動 故事

Figure 9. Sample keywords for different domains

4.3 Test Data and Evaluation Procedure

We used four sets of test data for 26 domains:

- (1) A set of 40 source terms in English for each domain. We randomly select 40 terms from each domain for testing. All testing terms are single-word and not used in the training procedure.
- (2) The set of out of vocabulary (OOV) terms from Set (1). We use an online English-Mandarin dictionary⁴ offered by Yahoo! to check each terms in Data Set (1). If no translation is found, we consider the term OOV and put it in Data Set (2).
- (3) The set of terms with domain-specific translations (DST) from Set (1). We searched each term in Data Set (1) in Wikipedia. If there are disambiguation links on the first returned pages, we consider the term to have DSTs. Wikipedia terms that need to be disambiguated

⁴ <http://tw.dictionary.yahoo.com>

tend to have different meaning due to domain distinction, thus are likely to have domain-specific translation different from their frequent, general translations.

- (4) A set of sentences containing the 40 source terms under the first three domain tests (BIOLOGY, CHEMISTRY, and PHYSICS). We selected the first sentence of the Wikipedia entry for the term in question.

Recall our goal is to find domain-specific translations for a given term. Therefore, our experimental evaluation involves both translations and domain. The evaluation was performed by two judges. For example, the term “port” is used in both COMPUTER and GEOGRAPHY domains with the respective translations of “埠” and “港口.” Therefore, when evaluating the system output for the COMPUTER domain, the judges considered “埠” correct and “港口” incorrect. Likewise, for the GEOGRAPHY domain, “埠” was considered incorrect, while “港口” was considered correct.

4.4 Evaluating Domain Translations

We now describe the evaluation procedures and the overall performances. The performances of all systems compared are influenced by domain information. We discovered in our experiments that the QE keywords generated based on the filtering and ranking method described Section 3.2.3 is reasonably effective in general. As expected, these QE terms are especially more effective in the case of finding translation for OOV and DST cases.

All Terms	Top 1	Top 2	Top 3
<i>TermMine QE-</i>	65.4	75.7	77.3
<i>TermMine</i>	73.4	83.1	87.0
<i>TermMine +</i>	76.6	85.8	88.9
<i>Google Translate</i>	69.0	-	-
<i>WRF</i>	56.4	64.0	66.7

Figure 10. The overall evaluation results.

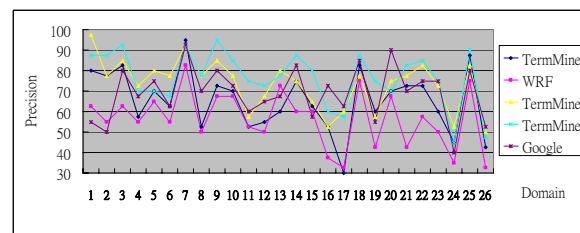


Figure 11. The performances by domain.

Figure 10 shows that *TermMine*, and the improved version *TermMine+* deliver the best performances for most of the domains evaluated. This suggests that domain information is useful and the proposed approach based on query expansion is a reasonable way to exploit the domain information. Additionally, the training process is effective in learning TL domain keyword for QE. As indicated in Figure 11, the proposed learning method consistently learns reasonable and general domain keywords across domains studied in the experiments. Additionally, for the learning procedure to be effective, it takes as little as 100 source terms for each domain.

As indicated in Figures 12 and 13, query expansion is very effective in improving the accuracy rate of top one translation for both OOV and DST cases. For the OOV cases, there is a significant increase (over 5%) in the average accuracy rate. For the DST case, the increase in the accuracy rate is even more dramatic (almost 20%).

The effectiveness of a second round of pseudo relevance feedback is mixed. For the OOV cases, relevance feedback results in a small drop in accuracy rate, while for the DST cases it results in marginal improvement (about 2%). However, this improvement in the DST cases tapers off for the Top 2 and 3 translations.

We also used Data Set (4) to examine whether sentential context will indirectly provide the domain information and help *Google Translate* select the correct DSTs. Figure 14 shows the average precision rates of term translation with and without sentential context for three domains.

As indicated in the figure, the results are mixed. When given the sentential context, the precision rate improves for the CHEMISTRY domain, but deteriorates slightly for the BIOLOGY domain. The reason is that the sentential contexts certain may help in the DST case. For example, the BIOLOGY term, “culture” is likely to be translated into “培養” correctly, given the context information. But for OOV BIOLOGY terms (e.g., “bromouracil” and “microevolution”), *Google Translate* tends to leave the terms not translated, regardless a term or a sentence is submitted as input. Overall, the context information does not offer much help to increase the translating performance.

In general, *Google Translate* offers impressive performances on translating common terms,

especially when translations are domain-independent. The OOV and DST problems seem to be the weak points upon which state-of-the-art MT systems, such as *Google Translate* can be improved.

Systems	Top 1	Top 2	Top 3
<i>TermMine QE-</i>	49.0	61.5	65.6
<i>TermMine</i>	56.2	63.5	66.7
<i>TermMine +</i>	51.0	61.5	65.6
<i>Google Translate</i>	12.5	-	-
<i>WRF</i>	42.7	51.0	55.2

Figure 12. The performances in the OOV cases

Systems	Top 1	Top 2	Top 3
<i>TermMine QE-</i>	51.1	57.4	57.5
<i>TermMine</i>	68.8	78.5	81.7
<i>TermMine +</i>	71.0	76.3	78.5
<i>Google Translate</i>	36.6	-	-
<i>WRF</i>	43.8	46.9	51.0

Figure 13. The performances in the DST cases

Domain	# of correct translation	Prec. (term)	Prec. (sent.)
BIOLOGY	21	0.55	0.53
CHEMISTRY	22	0.50	0.55
PHYSICS	32	0.80	0.80

Figure 14. The number of correctly translated terms and average accuracy rates for 40 terms in each of the three domains tested.

5 Conclusion

Many avenues exist for future research and improvement of the proposed method. For example, the number of domains employed can be increased to allow fine-grained domain-specific translations. Domain information, currently provided by the user, can be automatically derived from the sentence or paragraph context of the given term using text categorization techniques. Additionally, such categorization methods can also be applied to filter out the returned snippets not in the intended domain before extracting translations. We used monolingual data for training query expansion so it is easy to repurpose the system for another language. It will be interesting to see if using translations in addition to the source would lead to better keywords and better performances in retrieving domain-specific translations.

In summary, we have introduced a method for extracting domain-specific translations by taking

the domain information into consideration. The method involves selecting the source terms in various domains, generating domain keywords for query expansion, and extracting domain-specific translations in retrieved mixed-code document snippets. We have implemented and thoroughly evaluated the method for translating English terms into Chinese translations. In the comparative evaluations with comparing different systems, we have shown that the method substantially outperforms other translating systems or approaches in domain-specific term translations.

References

- Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2): 79-85.
- Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter Estimation. *Computational Linguistics*, 19(2): 263-311.
- Yunbo Cao and Hang Li. 2002. Base noun phrase translation using web data and the EM algorithm. In *Proceedings of COLING*, 1-7.
- Bonnie J. Dorr, Pamela W. Jordan, and John W. Benoit. 1998. A Survey of current paradigms in machine translation. LAMP-TR-027/UMIACS-TR-98-72/CS-TR-3961, *Computer Science Department, University of Maryland*.
- Pascale Fung and Lo Yuen Yee. 1998. An IR approach for translating new words from nonparallel, comparable texts. In *Proceedings of ACL*, 414-420.
- Fei Huang, Ying Zhang, and Stephan Vogel. 2005. Mining key phrase translations from web corpora through crosslingual query expansion. In *Proceedings of the 28th annual international ACM SIGIR*, 669-670.
- John W. Hutchins. 1995. Machine translation: A brief history. In *Concise history of the language sciences: from the Sumerians to the cognitivists*. Koerner E.F. Konrad and Asher E. Ronald, Eds. Oxford: Pergamon Press, 431-445.
- Philipp Koehn, and Kevin Knight. 2003. Feature-rich statistical translation of noun phrases. In *Proceedings of ACL*, 311-318.
- Philipp Koehn. 2004. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Proceedings of AMTA*, 115-124.
- Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, 177-180.
- Adam Lopez. 2006. A survey of statistical machine translation. LAMP-TR-135/CS-TR-4831/UMIACS-TR-2006-47, *Computer Science Department, University of Maryland*.
- Wen-Hsiang Lu, Lee-Fung Chien, and Hsi-Jian Lee. 2002. Translation of web queries using anchor text mining. *ACM Transactions on Asian Language Information Processing*, 1(2): 159-172.
- Dragos Stefan Munteanu, Alexander Fraser and Daniel Marcu. 2004. Improved machine translation performance via parallel sentence extraction from Comparable Corpora. In *Proceedings of HLT-NAACL*, 265-272.
- Masaaki Nagata, Teruka Saito, and Kenji Suzuki. 2001. Using the web as a bilingual dictionary. In *Proceedings of the workshop on Data-driven methods in machine translation*, 1-8.
- Li Shao and Hwee Tou Ng. 2004. Mining new word translations from comparable corpora. In *Proceedings of COLING*, 618-624.
- Chuan-Yao Su. 2006. Bilingual proper nouns extraction through web mining. Master thesis, National Chao Tung University, Taiwan.
- Jian-Cheng Wu, Tracy Lin, and Jason S. Chang. 2005. Learning source-target surface patterns for web-based terminology translation. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, 37-40.
- Jian-Cheng Wu, and Jason S. Chang. 2007. Learning to find English to Chinese transliterations on the Web. In *Proceedings of EMNLP-CoNLL*, 996-1004.