Improving English-to-Chinese Translation for Technical Terms Using Morphological Information

Xianchao Wu[†] **Naoaki Okazaki**[†] **Takashi Tsunakawa**[†] **Jun'ichi Tsujii**[†‡]

[†]Computer Science, Graduate School of Information Science and Technology, University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

[‡]School of Computer Science, University of Manchester
National Centre for Text Mining (NaCTeM)
Manchester Interdisciplinary Biocentre, 131 Princess Street, Manchester M1 7DN, UK

{wxc, okazaki, tuna, tsujii}@is.s.u-tokyo.ac.jp

Abstract

The continuous emergence of new technical terms and the difficulty of keeping up with neologism in parallel corpora deteriorate the performance of statistical machine translation (SMT) systems. This paper explores the use of morphological information to improve English-to-Chinese translation for technical terms. To reduce the morpheme-level translation ambiguity, we group the morphemes into morpheme phrases and propose the use of domain information for translation candidate selection. In order to find correspondences of morpheme phrases between the source and target languages, we propose an algorithm to mine morpheme phrase translation pairs from a bilingual lexicon. We also build a cascaded translation model that dynamically shifts translation units from phrase level to word and morpheme phrase levels. The experimental results show the significant improvements over the current phrase-based SMT systems.

1 Introduction

Statistical machine translation (SMT) provides an impressive framework in which a machine translation system can be built, only if a parallel corpus is available. Koehn (2005) collected a corpus of parallel text in 11 languages and trained SMT systems for 110 language pairs within three weeks. However, the experimental results also revealed several language-specific issues: morphologically-rich languages (e.g., German) were more difficult to translate into than from, and two distant languages (e.g., Finnish and English) that have discrepant morphologies (word structures) were difficult to deal with. These issues arise because SMT systems usually employ words as the minimum units of translation, even when some elements represented by individual words in one language are included in the morphology of another language.

Numerous researchers have proposed a variety of approaches that make use of morphological information in machine translation (Popović and Ney, 2004; Goldwater and McClosky, 2005; Oflazer and El-Kahlout, 2007; Virpioja et al., 2007; Oflazer, 2008). Most studies assume that the input language (e.g., Arabic, Catalan, Czech, and Spanish) is morphologically richer than the output language (English) because translating from an information-rich language into an information-poor language is easier than the other way around (Koehn 2005). Some recent studies (Oflazer and El-Kahlout, 2007; Oflazer, 2008) explored the opposite direction (e.g., English to Turkish), but more case studies are necessary.

Moreover, a number of new technical terms are emerging daily in English, which is the dominant international language in communications, science, business, politics, etc. Due to the difficulty of keeping up with neologism in parallel corpora, SMT systems suffer from the data sparseness problem and the out-of-vocabulary (OOV) problem. Thus, translating English technical terms into other languages is a crucial challenge in SMT and useful for various natural language applications including cross-lingual information retrieval (CLIR).

In this paper, we present an approach that leverages morphological information for translating En-

Domain	抗 (*Kang*)	反 (*Fan*)	防 (*Fang*)	Other translations	# of 63 translations in one domain
Immunology	**265**, antibody(抗体)	0	0	0	265
Army	2, anti-static(抗静电)	**90**, anti-reconnaissance(反侦察)	19, antiaircraft(防空)	18	129
Pharmaceutics	**76**, antitoxin(抗毒素)	0	0	29	105
Mathematics	0	**81**, anti-automorphism(反自同构)	0	22	103
Physics	8, anti-reflection(抗反射)	**56**, antimicrophonic(反颤噪声的)	11, antiswing(防摆动)	19	94
Auxiliaries	27, antioxidant(抗氧化剂)	0	**55**, antioxidant(防老剂)	12	94
Other domains	287	266	280	-	-
# of one translation in 109 domains	665	493	365	-	-

Table 1: English prefix $anti$ corresponds to 63 Chinese characters in 109 domains in our MPTP set. In which 3 Chinese characters and 6 domains together with example words are listed here. The numbers stand for the frequency counts.

glish technical terms into Chinese. These terms tend to contain a large amount of Latin and Greek word roots, concatenated into words such as $supernova$ and $trichloroethane$. Words of this type have a rich derivational morphology, despite the fact that English has a poor inflectional morphology. Even though English has a discrepant morphology compared with Chinese, we report significant improvements in translating English technical terms into Chinese over the current phrase-based SMT systems.

2 Issues in technical term translation

We address several key research issues in utilizing morphological information for translating English technical terms into Chinese. First, English morphemes and Chinese characters do not always provide a consistent translation unit. The *morpheme collapse problem* on both language sides (i.e., a morpheme in one language may correspond to zero, one, or multiple morpheme(s) in another language, and vice versa[1]) is the motivation for doing morpheme grouping on both language sides (Virpioja et al., 2007) instead of on a single side.

Second, translation disambiguation is still essential on morpheme level translation. Despite the fact that Chinese can be seen as a morphologically-poor language, ideographic Chinese characters complicate the translation disambiguation problem. Table 1 illustrates the diversity of translating one English prefix $anti$: 63 Chinese characters are possible translations appearing in 109 technical/scientific domains. Even if we resort to the n-gram (character) language model (LM), it further suffers from the data sparseness problem. The essential observation is that corresponding to one English morpheme, one or several Chinese characters are always dominant translations in specific domains. For example, $Kang$ is the only possible translation of $anti$ in the **Immunology** domain.

Third, morpheme-based translation is mainly aiming at the OOV problem and the data sparseness problem. Generally, numerous works have shown that integrating different levels of translation units, from morpheme to word, and to phrase, is crucial for achieving high accuracy SMT (Popović and Ney, 2004; Oflazer and El-Kahlout, 2007; Virpioja et al., 2007; Oflazer, 2008).

In order to handle the first problem, we group the correspondences between English morphemes and Chinese characters. We denote these translation-oriented correspondences as morpheme phrase translation pairs (MPTPs). We attach domain information to MPTPs for choosing correct Chinese characters to deal with the second problem. We also build a maximum entropy (ME) model that predicts the possible domains of a given English term. In answer to the third problem, we build a cascaded translation model that dynamically shifts translation units from phrase level to word and morpheme phrase levels. A beam search style cascaded decoding algorithm is designed for this translation model.

[1]e.g., $supernova(ChaoXinXing) : super \sim Chao$ (1 English prefix∼1 Chinese character), $nova \sim Xin\ Xing$ (1 English stem∼2 Chinese characters); $reaction(FanYing) : re\ act \sim Fan\ Ying$ (2 English morphemes∼2 Chinese characters), -$(t)ion \sim$ (1 English suffix∼0 Chinese character)

The remainder of this paper is organized as follows. Section 3 describes the cascaded model for translating English technical terms and the ME model for estimating possible domains of a given source term. Section 4 presents an algorithm to mine domain specific MPTPs from a bilingual lexicon. In Section 5, we describe the decoding algorithm for the translation model. Section 6 reports our experimental setup and results. After reviewing the related work briefly in Section 7, we conclude this paper in Section 8.

3 Cascaded translation model

Our cascaded translation model is a direct translation model using a LM for ranking. It can be regarded as a reversal of the order of the source and target languages in the noisy-channel model (Brown et al., 1993).

Let E and C denote terms in the source and target languages. The translation model produces the optimal term C^* given E,

$$C^* = \arg\max_C p(C|E)p(C). \quad (1)$$

In order to choose a target term C by making use of the domain information of the source term E, we integrate the domain D in the channel model,

$$p(C|E) = \sum\nolimits_D p(C|E,D)p(D|E). \quad (2)$$

We estimate D of E using a log-linear ME model:

$$p_\Lambda(D|E) = \frac{1}{Z_\Lambda(E)} \exp\left\{\sum_k \lambda_k h_k(D,E)\right\}, \quad (3)$$

$$Z_\Lambda(E) = \sum\nolimits_D \exp\left\{\sum_k \lambda_k h_k(D,E)\right\}.$$

Here, $p_\Lambda(D|E)$ is computed by the feature vector $\{h_k\}$, and the feature weight vector $\Lambda = \{\lambda_k\}$. We use the words and the morphemes inside E as features, where the morphemes are gained by using Morfessor[2], an unsupervised language-independent morphological analyzer (Creutz and Lagus, 2007). The feature weights are estimated by using a maximum *a posteriori* (MAP) estimator:

$$\mathcal{L}(\Lambda) = \sum_{(E,D)\in\mathcal{D}} \log p_\Lambda(D|E) - \sum\nolimits_k \left(\frac{\lambda_k}{\sigma}\right)^2. \quad (4)$$

[2]http://www.cis.hut.fi/projects/morpho/

L_2 regularization is employed here, and the regularization parameter σ is tuned using a development set. We maximize Equation 4 using L-BFGS (Nocedal and Wright, 1999).

In Equation 2, when we express the alignment A explicitly, the translation probability under domain D is written as

$$\begin{aligned} p(C|E,D) &= \sum\nolimits_A p(C,A|E,D) \\ &\simeq p(C,A^*|E,D). \end{aligned} \quad (5)$$

Here, $A^* = \{a_i\}$ denotes the *Viterbi* (the most probable) alignment between E and C for simplifying the model (maximum approximation). n-gram phrases of E are first considered during decoding where n ranges over N down to 1 and N is the maximum phrase length (see Section 5 for decoding detail). We approximate the translation probability of E to C under A^*, given D, by the product of the translation probabilities of E's individual n-gram phrase $\overrightarrow{e}_i$ to C's individual phrase $\overrightarrow{c}_{a_i}$:

$$p(C,A^*|E,D) \simeq \prod\nolimits_i p(\overrightarrow{c}_{a_i}|\overrightarrow{e}_i,D). \quad (6)$$

Only when failing in the phrase/word level, does the decoding algorithm use the morpheme phrase level. Suppose that the current word e_i ($|\overrightarrow{e}_i| = 1$) is an OOV word, we consider all possible partitions s of e_i:

$$p(c_{a_i}|e_i,D) = \sum\nolimits_s p(c_{a_i}|e_i,s,D)p(s|e_i,D). \quad (7)$$

We assume uniform distribution among partitions; thus $p(s|e_i,D)$ can be omitted during decoding. Furthermore,

$$\begin{aligned} p(c_{a_i}|e_i,s,D) &= p(\overrightarrow{\omega}^s_{a_i}|\overrightarrow{\epsilon}^s_i,D) && (8) \\ &\simeq \prod\nolimits_k p(\omega^s_{a_i,k}|\epsilon^s_{i,k},D). && (9) \end{aligned}$$

In Equation 8, $\overrightarrow{\epsilon}^s_i$ denotes the English morpheme phrase (ϵ) sequence of e_i under s, and $\overrightarrow{\omega}^s_{a_i}$ denotes the Chinese sequence of characters (ω) of c_{a_i} under s. Formula 9 assumes that e_i and c_{a_i} should have the same number of morpheme phrases. During training, we introduce an empty morpheme symbol $\star$ to Chinese, in case some English morphemes (e.g., morphological suffixes) do not correspond to any Chinese character. The appearances of the symbol $\star$ are removed from the decoding outputs.

4 Domain specific MPTP mining

In this section, we describe the domain specific MPTP mining algorithm to estimate $p(\omega|\epsilon, D)$ in the cascaded translation model. The algorithm consists of a set of word-splitting heuristic rules, which group the correspondences between English morphemes and Chinese characters.

4.1 Preprocessing

We mine MPTPs from word level translation pairs. The available dictionary is phrasal in which Chinese words are not segmented beforehand. For the reason that more than half a million technical terms (Table 4) are included in the dictionary, Chinese word segmentation programs based on existing dictionaries may not perform well. Instead, we utilize English words as a clue for segmenting Chinese phrases into words. We use the EM-based GIZA++ (Och and Ney, 2000), and extract a word level dictionary.

We first tokenize English terms by spaces and hyphens, and lowercase the words. Chinese phrases are divided into character sequences. We use GIZA++ for aligning English words to Chinese characters. Then, Chinese characters aligned to the same English word are considered as a word. Let W^0 denote the word level dictionary. Each entry $\boldsymbol{w} \in W^0$ is represented by a 4-tuple $\boldsymbol{w} = \langle e, c, d, n\rangle$, where e, c, d, and n present an English string (word or morpheme), a Chinese string, the domain of the translation pair, and the frequency of the occurrences of the entry in the dictionary. We represent each item in $\boldsymbol{w}$ with $\boldsymbol{w}_e$, $\boldsymbol{w}_c$, $\boldsymbol{w}_d$, and $\boldsymbol{w}_n$, respectively.

4.2 MPTP mining algorithm

The mining algorithm is illustrated in Algorithm 1. We extract the entries in W^0 that consist of only one Chinese character as the initial MPTP set V^1 (line 1). Then, $V^i (i \geq 1)$ is applied to the current remaining dictionary W^i to split the words and to generate new MPTPs (lines 2∼13).

Generally, at iteration $i \geq 1$, suppose $\boldsymbol{w} \in W^i$ and $\boldsymbol{v} = \langle \epsilon, \omega, d, n\rangle \in V^i$, the new generated MPTP set is written as (lines 4∼9)

$$V^{i'} \quad = \quad \{\boldsymbol{v}' \mid \boldsymbol{v}' \in \texttt{split}(\boldsymbol{w}, \boldsymbol{v})\}$$

(the `split` function is described later). We then *unify* ($\sqcup$) $V^{i'}$ with V^i by cleaning up the overlapping entries, i.e. adding up their n and removing the duplicated ones (line 10). Simultaneously, considering that a $\boldsymbol{w}$ may fail to be split by any $\boldsymbol{v} \in V^i$, and the new generated MPTPs may be split again by other MPTPs (thus $|\boldsymbol{v}_\omega|$ and $|\boldsymbol{v}_\epsilon|$ should ≥ 2, line 8 and 11), new remaining dictionary is written as

$$\begin{aligned} W^{i+1} \quad = \quad & \{\boldsymbol{w} \mid \texttt{split}(\boldsymbol{w}, \boldsymbol{v}) = \phi\} \sqcup \\ & \{\boldsymbol{v} \mid \boldsymbol{v} \in V^{i'} \wedge |\boldsymbol{v}_\omega| \geq 2 \wedge |\boldsymbol{v}_\epsilon| \geq 2\}. \end{aligned}$$

The unified MPTP set V^{i+1} can be used again for mining new MPTPs from W^{i+1}, or be unified to generate the last MPTP set (line 14). In order to avoid overlapped frequency counting in the $\sqcup$ operations, we mark each $\boldsymbol{v} \in V^{i'}$ that returns back to W^{i+1} (line 11). Marked entries' n will not be accumulated during the $\sqcup$ operation. We cancel the mark when $\boldsymbol{v}$ is split again during the iterations (line 7).

We now describe the `split` function in detail. We split an entry $\boldsymbol{w} \in W^i$ when the following condition holds:

$$\exists \boldsymbol{v} \in V^i,\ s.t.\ \boldsymbol{v}_\epsilon \in \boldsymbol{w}_e \ \wedge\ \boldsymbol{v}_\omega \in \boldsymbol{w}_c \ \wedge\ \boldsymbol{v}_d = \boldsymbol{w}_d.$$

The domain of each new generated MPTP is set to be $\boldsymbol{w}_d$, and the frequency count is set to be $\boldsymbol{w}_n$. When more than one $\boldsymbol{v}$ is available for one $\boldsymbol{w}$, we choose the $\boldsymbol{v}$ that has the highest frequency (line 6).

In actuality, we use a slightly stricter condition than simple inclusion, regarding the position of ϵ and ω in e and c. Let L and R denote non-empty substrings, there are 4 possible positions where ϵ can appear in e: exact match ($e = \epsilon$), left ($e = \epsilon R$), right ($e = L\epsilon$), and middle ($e = L\epsilon R$), analogous with ω in c.

Algorithm 1 MPTP mining

Require: W^0, $N \triangleright W^0$, initial word level dictionary; N, iteration number

1: $V^1 \leftarrow \{\boldsymbol{w} | \boldsymbol{w} \in W^0 \wedge |\boldsymbol{w}_c| = 1\}, W^1 \leftarrow W^0 \setminus V^1$
2: $i \leftarrow 1$
3: **while** $i \leq N$ **do**
4: $\quad V^{i'} \leftarrow \{\}, W^{i+1} \leftarrow \{\}$
5: $\quad$ **for** $\boldsymbol{w} \in W^i$ **do**
6: $\quad\quad \boldsymbol{v} \leftarrow argmax_{\{\boldsymbol{v} | \boldsymbol{v} \in V^i \wedge \boldsymbol{v}_\epsilon \in \boldsymbol{w}_e \wedge \boldsymbol{v}_\omega \in \boldsymbol{w}_c \wedge \boldsymbol{v}_d = \boldsymbol{w}_d\}} \boldsymbol{v}_n$
7: $\quad\quad V^{i'} \leftarrow V^{i'} \sqcup \{\boldsymbol{v}' | \boldsymbol{v}' \in \texttt{split}(\boldsymbol{w}, \boldsymbol{v})\}$
8: $\quad\quad W^{i+1} \leftarrow W^{i+1} \sqcup \{\boldsymbol{w} \mid \texttt{split}(\boldsymbol{w}, \boldsymbol{v}) = \phi\}$
9: $\quad$ **end for**
10: $\quad V^{i+1} \leftarrow V^i \sqcup V^{i'}$
11: $\quad W^{i+1} \leftarrow W^{i+1} \sqcup \{\boldsymbol{v} \mid \boldsymbol{v} \in V^{i'} \wedge |\boldsymbol{v}_\omega| \geq 2 \wedge |\boldsymbol{v}_\epsilon| \geq 2\}$
12: $\quad i \leftarrow i + 1$
13: **end while**
14: MPTP $\leftarrow V^{i+1} \sqcup W^{i+1}$ $\triangleright$ the last MPTP set
15: **return** MPTP $\triangleright$ the output

$e \backslash c$	ω	$\omega R'$	$L'\omega$	$L'\omega R'$
ϵ	$\langle e, c\rangle$	$\langle e, c\rangle$	$\langle e, c\rangle$	$\langle e, c\rangle$
ϵR	$\langle e, c\rangle$	$\langle \epsilon, \omega\rangle$ $\langle R, R'\rangle$	-	$\langle \epsilon, L'\omega\rangle$ $\langle R, R'\rangle$
$L\epsilon$	$\langle e, c\rangle$	-	$\langle L, L'\rangle$ $\langle \epsilon, \omega\rangle$	$\langle L, L'\rangle$ $\langle \epsilon, \omega R'\rangle$
$L\epsilon R$	$\langle e, c\rangle$	$\langle L\epsilon, \omega\rangle$ $\langle R, R'\rangle$	$\langle L, L'\rangle$ $\langle \epsilon R, \omega\rangle$	$\langle L, L'\rangle$ $\langle \epsilon, \omega\rangle$ $\langle R, R'\rangle$

Table 2: The heuristic rules used to split a entry ($\langle e, c, d, n\rangle$) by a MPTP ($\langle \epsilon, \omega, d, n\rangle$) to generate new MPTPs. d and n are omitted for simplicity. In the diagonal conditions, the same MPTP is generated again.

Table 2 presents the rules used to split words and generate new MPTPs. For example, suppose $\epsilon = super$, $\omega = Chao$, $e = supernova$ ($\epsilon nova$), and $c = ChaoXinXing$ ($\omega XinXing$), then the new MPTPs would be like $\langle nova, XinXing, d, n\rangle$ and $\langle super, Chao, d, n\rangle$. In addition, we prohibit cross-aligning morpheme phrases inside words for decoding convenience. Thus, two position combinations are ignored in Table 2: ϵ appears in the left part of e but ω appears in the right part of c, and vice versa.

The mining algorithm can iterate until V^{i+1} and W^{i+1} converge, ideally. In our experiments, we just iterate constant ($N = 5$) times to save computing time.

One limitation of this algorithm is that it entirely depends on the initial MPTP set V^1. When V^1 is too small compared to the dictionary W^0, few new MPTPs will be generated. In order to alleviate the dependency on single Chinese characters (line 1), we used Morfessor to segment English words in our training data into morphemes. We used GIZA++ on both directions between English morphemes and Chinese characters, and took the intersection of *Viterbi* alignments to extract another initial MPTP set. For the English morphemes that do not align with any Chinese character, we assign them with the empty morpheme $\star$. In order to reduce ambiguity, MPTPs with $\star$ are not used for splitting words.

In Section 6.2, we compare three strategies for generating initial MPTP sets: single Chinese character based, Morfessor based, and their combination. In the combination process, when one MPTP appears in both initial sets, only the one with the bigger frequency count is kept. We run our mining algorithm the same constant times on the three initial MPTP sets. Maximum likelihood estimations are respectively applied to the last three MPTP sets to determine $p(\omega|\epsilon, D)$ in the translation model.

Algorithm 2 cascaded decoding

Require: $EWords[]$, D, **T**, MPTP, N, b $\triangleright EWords[]$, source language term; D, domain label; **T**, phrase translation (hash)table; N, maximum phrase length in **T**; b, beam size.

1: $start \leftarrow 1$,$end \leftarrow \min\{N, |EWords|\}$
2: **while** $start \leq |EWords|$ **do**
3: $isInT \leftarrow false$
4: **for** $end' \leftarrow end$ downto $start$ **do**
5: $ephrase \leftarrow EWords[start..end']$
6: **if** $ephrase \in$ **T**.keys **then**
7: $CPhrase\_p[] \leftarrow$**T**.get($ephrase, b$) $\triangleright$ $CPhrase\_p[]$,a list storing current target language phrases and their probabilities.
8: $isInT \leftarrow true$
9: $start \leftarrow end' + 1$
10: $end \leftarrow \min\{start + N, |EWords|\}$
11: **break** $\triangleright$ jump out **for**
12: **end if**
13: **end for**
14: **if** $isInT = false$ **then**
15: $CPhrase\_p[] \leftarrow$ ***decodeWord*** ($EWords[start]$, D, MPTP) $\triangleright$morpheme phrase level decoding. D is dynamically predicted or given beforehand.
16: $start \leftarrow start + 1$
17: $end \leftarrow \min\{end + 1, |EWords|\}$
18: **end if**
19: $Result[] \leftarrow$ ***append***$(Result[], CPhrase\_p[], b)$ $\triangleright$ a list storing target language phrases and their probabilities
20: **end while**
21: **return** $Result[]$ $\triangleright$ the output

5 Cascaded decoding algorithm

5.1 Decoding algorithm

Algorithm 2 illustrates the pseudocode of our beam search style cascaded decoding algorithm. The algorithm first decodes at the n-gram phrase level, where the indexes $start$, end and end' are used to segment the source language term into existing n-gram phrase sequence (lines 3$\sim$13). If this fails, the algorithm splits current OOV word into all possible morpheme phrase sequences, and search translation candidates in the last MPTP set (lines 14$\sim$18). When the domain label D is given beforehand, we set $p(D|E)$ to 1 in Equation 2. Otherwise, we use the log-linear model (Equation 3) to estimate the probabilities of the possible domains (line 15).

We integrate an n-gram character LM in morpheme phrase level decoding and an n-gram word

LM in phrase level decoding[3] (lines 15 and 19).

Let us consider the process in which an English term $superconducting\ quantum\ magnetometer$ is translated. The reference Chinese term is $ChaoDao\ LiangZi\ CiQiangQi$. Suppose $N = 2$ and $D =$ **physics**, we first segment the source term into two phrases: $superconducting\ quantum$ and $magnetometer$. We check if both of them appear in the phrase translation table **T** (refer to line 6 and Equation 6). We terminate the decoding algorithm if translation candidates are found in **T** under some segmentation in phrase/word level. Otherwise, the term would contain at least one OOV word. Assume that $superconducting$ is an OOV word, we consider all possible partitions of it (refer to Equation 7), e.g., $super\ con\ duct\ ing$, $superconduct\ ing$, etc. When considering a partition $superconduct\ ing$, we express $\overrightarrow{\epsilon} = (superconduct, ing)$ (refer to Equation 8) and morpheme phrase $\epsilon \in \overrightarrow{\epsilon}$ (refer to Formula 9). The translation candidates for each ϵ are searched among entries associated with **physics** domain in the MPTP set. $ChaoDao$ and $\star$ are the possible translation candidates for $superconduct$ and ing; thus, one possible translation could be $ChaoDao$ for $superconducting$. The remaining words in the given term are translated in a similar manner.

5.2 Reordering patterns

The alignment distortion problem is easy to settle since the number of words in an English technical term is generally small. We extract reordering patterns by observing the *Viterbi* alignments gained from GIZA++, inspired by the synchronous CFG rule extraction approach presented in (Chiang, 2007). According to our preliminary experimental results, approximately 10% of entries are reordered in the training set. Especially of caused about 35% of the reordering: $x\ of\ y \rightarrow y\ [de]\ x$. The Chinese genitive auxiliary word de is optional here but we just omit it for simplification. We only keep the top-5 patterns active. The other 4 patterns are: $type\ x\ y \rightarrow x\ Xing\ y$, $group\ x\ y \rightarrow x\ Zu\ y$, $class\ x\ y \rightarrow x\ Lei\ y$, and $element\ x \rightarrow x\ YuanSu$. x is limited to be one word in the middle three patterns in order to separate x from y.

[3]Hypothesis combinations are also checked and applied in these two steps referring to the ideas in Pharaoh (Koehn, 2004).

Domain	# of entries	%
Computer Science	55,902	.101
Mathematics	28,743	.052
Mechanics	22,914	.042
Electronics	17,764	.032
Electrology	15,210	.028
Anatomy	14,233	.026
Organization	14,003	.025
Mycology	12,896	.023
Science and Technology	11,461	.021
Architectonics	10,222	.019
total (551,570 entries)	203,348	.369

Table 3: The statistical information of the top-10 domains in the Wanfang dictionary.

	training	testing DS	testing MD
# of entries	540,570	1K×10	1K
# of *E* words	113,958	11,686	1,670
# of OOV *E* words	-	957	107
avg. *E* phrase length (word)	2.37	2.59	2.30
avg. *C* phrase length (character)	4.59	4.66	4.49

Table 4: The statistical information of the training and testing sets. E and C stand for English and Chinese respectively. Duplicate appearances of a word are counted only once.

For example, $type\ I\ supernova$ is reordered into $I\ type\ supernova$. These reordering patterns are optional to reorder the English phrases before decoding.

6 Experiments

6.1 Setups

The Wanfang Chinese-English technical term dictionary[4] with domain information, which contains 204 domains and 525,259 entries in total, was used for training and testing. In this dictionary, some entries are attached with more than one domain label. We obtained 551,570 entries by duplicating an entry with k domain labels into k different entries with each domain label. Table 3 illustrates that the top-10 domains cover 36.9% of the entries in the dictionary.

We randomly selected 1K from the 551,570 entries as the mixed domain (MD) test set and then selected 1K×10 entries from the top-10 domains (1K in each domain) as the domain specific (DS) test sets. The remaining 540,570 entries were taken as

[4]http://www.wanfangdata.com.cn/Search/ResourceBrowse.aspx

the training set. Table 4 shows some statistics on our training and testing sets.

We created a phrase translation table similar to most phrase-based systems (Och and Ney, 2004). We built a trigram character LM from the training data and a bigram word LM trained by SRILM[5] on the data where the Chinese phrases have been segmented into words by GIZA++ (Section 4.1). Considering that the number of words in a technical term is far smaller than a typical sentence, we empirically determined the beam size to be 50 for fast decoding and used the top-1 output for evaluating. The translation results were evaluated using 4-gram BLEU score (Papineni et al., 2002), WER (Word Error Rate), PER (Position independent word Error Rate), and EMatch (exact match)[6]. To take synonymic translations into account, we manually evaluate the translation of English terms that contain OOV words.

6.2 Effectiveness of domain specific MPTP mining approach

We compare our approach with a baseline system, which is similar to the work of (Virpioja et al., 2007). Their system employed Moses[7] (Koehn et al., 2007) trained on the data in which each word was segmented into morpheme sequences. In order to implement their method, we split Chinese phrases into characters and segmented English words into morphemes by using Morfessor. We produced three variants of the baseline system by setting the maximum phrase length in Moses to 2, 3 and 4, respectively. SRILM was used to train respective gram character LMs.

While the outputs of baseline variants are characters, the outputs of our system are words. For direct comparison, we also segment our word outputs into sequences of individual characters, and preprocess the reference sets in the same way.

As mentioned in the end of Section 4.2, we also compared three strategies for generating initial MPTP sets: single Chinese character based (*single*), Morfessor based (*morf*), and their combination (*comb*). We mined 485,647 MPTPs using *single*, 488,117 MPTPs using *morf*, and 809,847 MPTPs using *comb*[8]. Table 1 is actually an extract from MPTPs of *comb* setting. Table 5 shows that in the DS test sets, the performance of *single* and *morf* are comparable. In the MD test set, *morf* performs a little better than *single* dealing with OOV translation. Generally, *comb* performs the best in all the criteria owing to its higher coverage.

	BLEU	WER	PER	EMatch	OOV	ID
DS avg.	.0873	.8430	.7730	.0471	.019	*Moses_2*
	.2638	.6250	.5239	.0978	.022	*Moses_3*
	.2689	.6206	.5228	.1000	.023	*Moses_4*
	.3595	.5114	.3981	.1891	.195	*morf_pD*
	.3448	.5145	.3951	.1882	.256	*morf_gD*
	.3763	.4923	.3695	.2031	.155	*single_pD*
	.3131	.5576	.3882	.1654	.266	*single_gD*
	.4056	.4783	.3756	**.2076**	.218	*comb_pD*
	.3955	**.4763**	**.3691**	.2057	**.317**	*comb_gD*
MD	.1039	.8256	.7690	.056	.065	*Moses_2*
	.2799	.6283	.5508	.096	.056	*Moses_3*
	.2890	.6158	.5405	.100	.065	*Moses_4*
	.3788	.5050	.4077	.206	.196	*morf_pD*
	.3643	.5008	.4002	.201	**.346**	*morf_gD*
	.3558	.5004	.3957	.208	.178	*single_pD*
	.3033	.5530	.4181	.187	.243	*single_gD*
	.4068	.4849	.3937	.214	.262	*comb_pD*
	.3981	**.4768**	**.3805**	**.215**	**.346**	*comb_gD*

Table 5: Comparison of Moses and our translation models. OOV stands for the translation accuracy of English terms containing OOV words.

Table 5 also reports the significant improvements of our models compared with the baseline variants. The BLEU score was improved from *Moses_4*'s 0.2689 to *comb_pD* (predicted D)'s 0.4056 on average in the DS test sets, and from 0.2890 to 0.4068 in the MD test set. The improvements were also confirmed in the reductions of WER and PER. The numbers of the EMatch were approximately doubled in all test sets. We found that, in the DS test sets, *comb_pD* could successfully translate 21.8% of the terms with OOV words while *Moses_4* could achieve only 2.3%.

Independently, we verified the log-linear model for domain prediction (Equation 3) using the MD and SD test sets: top-1 accuracy of 62.9% and top-10 accuracy of 90.6%. The practicability of our approach and the predictability of the log-linear model were also verified by looking at the comparable re-

[5] http://www.speech.sri.com/projects/srilm/

[6] To avoid the influence from Chinese word segmentation problem for exact matching, spaces between Chinese characters in systems' outputs and the references are removed beforehand.

[7] http://www.statmt.org/moses/

[8] We use *single*, *morf*, and *comb* to delegate our model variants hereafter.

sults of *comb_gD* (given D), *comb_pD* and other pairs. In addition, *comb_gD* achieved about 30% better accuracies for OOV term translation compared with *comb_pD*.

6.3 Effectiveness of cascaded translation model

We compared with Pharaoh (Koehn, 2004) to verify the effectiveness of our cascaded translation model. Table 6 shows the comparison results on conditions of domain information used or not, given or not. *noD* denotes domain information is totally not employed during decoding. We implemented this by setting all the domain labels to be the same in the last MPTP set and executing the *unify* operation on it. *+p/-p* denotes with/without using the phrase translation table, *+r/-r* denotes with/without using the reordering patterns. BLEU, WER and PER were evaluated at word level. In addition, *+p* and *-r* are taken as the experiment configuration in Table 5.

Our *comb* model outperformed Pharaoh significantly both in DS and MD test sets, regardless of the availability of domain information. Under *noD*, our model was better than Pharaoh when *+p* while worse when *-p* by comparing the BLEU score and EMatch. In the MD test set, under *noD*, *pD*, and *gD*, we achieved the best BLEU score of 0.2299, 0.2425, and 0.2621 respectively, 33.0%, 40.3%, and 51.6% relatively better than Pharaoh's 0.1729. In addition, the effectiveness of domain information was proved again by comparing all the evaluation criteria of *noD*, *pD* and *gD*.

Furthermore, the improvement by using the phrase translation table was verified in the BLEU score and EMatch criteria, while it was difficult to conclude that in the WER or PER. The influence of reordering patterns on the four criteria are summarized as follows: negative influence in EMatch and WER, positive influence in PER, and the influence is not monotonic in the BLEU score (*+p*, *+r* is better. *-p*, *-r* is better under *gD* and *pD* while *+r* is better under *noD*).

6.4 Error analysis

Even though our translation system outperformed the baseline systems obviously, it still have room for improving. We selected the test output of our system (*comb*) on the MD testing set for error analysis (refer to the last rows in Table 5 and Table 6). For comparison, we also manually evaluated the test outputs of *Moses_4* and Pharaoh. The results are listed in Table 7.

	BLEU	WER	PER	EMatch	ID
DS avg.	.1385	.6989	.6656	.1411	Pharaoh
	.0813	.6708	.6421	.1247	*-p-r, noD*
	.0895	.6839	.6421	.1247	*-p+r, noD*
	.1503	.6576	.6181	.1540	*+p+r, noD*
	.1425	.6449	.6202	.1577	*+p-r, noD*
	.1731	.5785	.5507	.1988	*-p-r, pD*
	.1560	.6025	.5502	.1872	*-p+r, pD*
	.1936	.5977	.5462	.2010	*+p+r, pD*
	.1898	.5830	.5508	**.2076**	*+p-r, pD*
	.1772	**.5736**	.5451	.1968	*-p-r, gD*
	.1619	.5971	.5447	.1865	*-p+r, gD*
	.1905	.5957	**.5440**	.1996	*+p+r, gD*
	.1867	.5813	.5488	.2057	*+p-r, gD*
MD	.1729	.6884	.6709	.157	Pharaoh
	.1325	.6794	.6630	.128	*-p-r, noD*
	.1418	.6856	.6630	.128	*-p+r, noD*
	.2299	.6394	.6160	.172	*+p+r, noD*
	.2128	.6315	.6166	.174	*+p-r, noD*
	.1837	.5923	.5778	.201	*-p-r, pD*
	.1710	.6073	.5775	.196	*-p+r, pD*
	.2425	.6048	.5753	.210	*+p+r, pD*
	.2293	.5953	.5780	.214	*+p-r, pD*
	.2161	**.5727**	**.5598**	.207	*-p-r, gD*
	.2032	.5904	**.5598**	.201	*-p+r, gD*
	.2621	.5928	.5630	.211	*+p+r, gD*
	.2464	.5834	.5680	**.215**	*+p-r, gD*

Table 6: Comparison of Pharaoh and our *comb* model.

We found, actually, 80% of our system's output were correct translations while *Moses_4* and Pharaoh achieved only 43.6% and 43% respectively. Synonymous words or phrases share nearly half of *comb*'s correct translations, while most of them are not represented by the evaluation criteria like BLEU. Missing or redundant (m/r) characters were also critical. If this only happens on some subset of function words, which are sometimes dispensable, we took the candidate as correct.

Nearly half of the wrongly translated entries in *comb* were caused by the problem of m/r characters. Wrong meaning selection of a polysemous word caused the wrong translation of the term, referring to the example in the polysemous problem in Table 7: $cell$ in $specific\ gravity\ cell$ was translated into $XiBao(biologic\ cell)$ while the correct translation is $DianChi(battery)$. In addition, abbreviation and transliteration in the test sets also possibly led to wrong translations.

The m/r character problem and the polysemy problem could be alleviated by using larger or do-

(# of *comb* | M | P)	type	# of *comb* | M | P	examples (test source/reference/system output) selected from *comb*
correct (800 | 436 | 430)	exact match	215 | 100 | 157	1 - hydroxy - 2 - propanone/１－羟基－２－丙酮/１－羟基－２－丙酮
	synonymy	384 | 121 | 88	decompression chamber/减压 舱/减压 室; rose/玫瑰/蔷薇
	m/r characters	201 | 215 | 185	current - mode logic/电流 型 逻辑/电流 － 型 逻辑
wrong (200 | 564 | 570)	m/r characters	89 | 284 | 394	light antiaircraft artillery/小高 炮/轻 高射 炮兵
	polysemous	28 | 47 | 49	specific gravity cell /比 重 电池/比 重 细胞
	OOV	42 | 194 | 97	autohemorrhage/自出血/autohemorrhage
	abbreviation	24 | 20 | 14	grc/盖尔研究公司/grc
	transliteration	17 | 19 | 16	kallman syndrome/卡尔曼 综合征/kallman 综合征

Table 7: Error analysis on translating the MD test set. M and P denote *Moses_4* and Pharaoh respectively.

main specific n-gram character/word LMs. In addition, integrating abbreviation lexicons and transliteration models are also needed.

7 Related work

Koehn and Knight (2003a) investigated several empirical methods for splitting German compounds. Even though they did not apply morphology analysis to German compounds, their work integrated the idea of word-splitting into a phrase-based translation system. They reported significant results in translating German base noun phrases into English. We built our cascaded translation model inspired by their work. The differences are that we further employed morphological analysis to technical terms, grouped the morpheme correspondences, and proposed using domain information for morpheme-level translation disambiguation.

Virpioja et al. (2007) suggested grouping morphemes on both morphologically-rich language sides. They used Morfessor to segment words into morphemes and employed Moses to automatically find morpheme phrases on three Nordic languages (Danish, Finnish, and Swedish). Unfortunately, they reported a worse BLEU score compared to the standard word-based approach. We implemented their method as baseline to confirm the effectiveness of our MPTP mining approach (Section 6.2). In contrast, our domain specific MPTP-based translation system outperforms the baseline and a word/phrase-based translation system (Section 6.3).

Other works like (Popović and Ney, 2004; Goldwater and McClosky, 2005) all focused on translating morphologically-rich languages (e.g., Spanish, Catalan, Serbian, Czech) into English. A number of language specific characteristics are employed in their methods which prohibit the direct comparison with our work. Oflazer and El-Kahlout (2007) and Oflazer (2008) proposed the morpheme-based English to Turkish translation with selective morpheme-grouping on the Turkish side. They also used Moses to automatically group morphemes similar to (Virpioja et al. 2007).

8 Conclusion and future work

We investigated the feasibility of using morpheme phrases on the task of English-to-Chinese technical term translation that suffers from the data sparseness problem and the OOV problem. We verified the effectiveness of our domain specific MPTP mining algorithm through our translation models under three MPTP sets compared with Moses, which takes morpheme level training and testing sets. Ultimately, our mining approach shows a novel way of bridging monolingual morphology analysis to morpheme-based SMT. In general, external morphological analyzers are needed for transferring our mining approach to other language pairs.

We testified the effectiveness of our cascaded translation model under four configurations, i.e. *+p/-p* or *+r/-r*, compared with Pharaoh. As proved by Koehn and Knight (2003b), who integrated a base noun phrase translation subsystem into a SMT system, we argue that it is straightforward to integrate our cascaded translation model into English-to-Chinese SMT systems for technical text translation.

Considering the limited coverage on most domains of the dictionary used in our experiments, mining domain specific MPTPs from Web pages is

taken as one of our future works. It has been shown to be feasible for building large-scale bilingual dictionaries from Web pages by several papers such as (Lin et al., 2008; Cao et al., 2007). Another future work is to build a log-linear translation model in order to integrate richer feature sets for technical term translation, such as domain specific abbreviation lexicons, spelling variance features, and even transliteration models.

Acknowledgments

This work was partially supported by Grant-in-Aid for Specially Promoted Research (MEXT, Japan) and Japanese/Chinese Machine Translation Project in Special Coordination Funds for Promoting Science and Technology (MEXT, Japan). We would like to thank Takuya Matsuzaki for invaluable assistance and the anonymous reviewers for their constructive comments.

References

Peter F. Brown, Stephen 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.

Guihong Cao, Jianfeng Gao and Jian-Yun Nie. 2007. A system to Mine Large-Scale Bilingual Dictionaries from Monolingual Web Pages. In *Machine Translation Summit XI*, pages 57–64, Copenhagen, Denmark.

David Chiang. 2007. Hierarchical Phrase-Based Translation. *Computational Linguistics*, 33(2):201–228.

Mathias Creutz and Krista Lagus. 2007. Unsupervised Models for Morpheme Segmentation and Morphology Learning. *ACM Transactions on Speech and Language Processing*, 4(1):Article 3.

Sharon Goldwater and David McCloskey. 2005. Improving Statistical MT Through Morphological Analysis. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pages 676–683, Vancouver, British Columbia, Canada.

Philipp Koehn and Kevin Knight. 2003a. Empirical Methods for Compound Splitting. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics (EACL)*, pages 187–193, Budapest, Hungary.

Philipp Koehn and Kevin Knight. 2003b. Feature-Rich Statistical Translation of Noun Phrases. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL)*, pages 311–318, Sapporo, Japan.

Philipp Koehn. 2004. Pharaoh: A Beam Search Decoder for Phrase-based Statistical Machine Translation Models. In *Meeting of the American Association for Machine Translation (AMTA)*, pages 115-124, Washington DC, USA.

Philipp Koehn. 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Machine Translation Summit X*, pages 79–86, Phuket, Thailand.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the ACL 2007 Demo and Poster Sessions*, pages 177–180, Prague, Czech.

Dekang Lin, Shaojun Zhao, Benjamin Van Durme and Marius Paşca. 2008. Mining Parenthetical Translations from the Web by Word Alignment. In *Proceedings of ACL-08:HLT*, pages 994–1002, Columbus, Ohio, USA.

Jorge Nocedal and Stephen J. Wright. 1999. Numerical Optimization. Springer.

Franz Josef Och and Hermann Ney. 2004. The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, 30:417–449.

Kemal Oflazer and Durgar El-Kahlout. 2007. Exploring Different Representational Units in English-to-Turkish Statistical Machine Translation. In *Proceedings of the Second ACL Workshop on Statistical Machine Translation*, pages 25–32, Prague, Czech.

Kemal Oflazer. 2008. Statistical Machine Translation into Morphologically Complex Language. *Alexander F. Gelbukh (Ed.): Computational Linguistics and Intelligent Text Processing, 9th International Conference, CICLing 2008, Proceedings. Lecture Notes in Computer Science (4919). Springer 2008,* pages 376–387, Haifa, Israel.

Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318, Philadelphia, USA.

Maja Popović and Hermann Ney. 2004. Towards the Use of Word Stems and Suffixes for Statistical Machine Translation. In *4th International Conference on Language Resources and Evaluation (LREC)*, pages 1585–1588, Lisbon, Portugal.

Sami Virpioja, Jaakko J. Väyrynen, Mathias Creutz and Markus Sadeniemi. 2007. Morphology-Aware Statistical Machine Translation Based on Morphs Induced in an Unsupervised Manner. In *Machine Translation Summit XI*, pages 491–498, Copenhagen, Denmark.