An empirical method for identifying and translating technical terminology

Sayori Shimohata
Research & Development Group,
Oki Electric Industry Co., Ltd.
Crystal Tower 1-2-27 Shiromi,
Chuo-ku, Osaka 540-6025 Japan
shimohata245@oki.co.jp

Abstract

This paper describes a method for retrieving patterns of words and expressions frequently used in a specific domain and building a dictionary for machine translation(MT). The method uses an untagged text corpus in retrieving word sequences and simplified part-of-speech templates in identifying their syntactic categories. The paper presents experimental results for applying the words and expressions to a pattern-based machine translation system.

1 Introduction

There has been a continuous interest in corpus-based approaches which retrieve words and expressions in connection with a specific domain (we call them technical terms hereafter). They may correspond to syntactic phrases or components of syntactic relationships and have been found useful in various application areas, including information extraction, text summarization, and machine translation. Among others, a knowledge of technical terminology is indispensable for machine translation because usage and meaning of technical terms are often quite different from their literal interpretation.

One approach for identifying technical terminology is a rule-based approach which learns local syntactic patterns from a training corpus. A variety of methods have been developed within this framework, (Ramshaw, 1995) (Argamon et al., 1999) (Cardie and Pierce, 1999) and achieved good results for the considered task. Surprisingly, though, little work has been devoted to learning local syntactic patterns besides noun phrases. Another drawback of this approach is that it requires substantial training corpora, in many cases with part-of-speech tags.

An alternative approach is a statistical one which retrieves recurrent word sequences as collocations (Smadja, 1993)(Haruno et al., 1996)(Shimohata et al., 1997). This approach is robust and practical because it uses plain text corpora without any information dependent on a language. Unlike the former approach, this approach extracts various types of local patterns at the same time. Therefore, post-processing, such as part of speech tagging and syntactic category identification, is necessary when we apply them to NLP applications.

This paper presents a method for identifying technical terms from a corpus and applying them to a machine translation system. The proposed method retrieves local patterns by utilizing the n-gram statistics and identifies their syntactic categories with simple part-of-speech templates. We make a machine translation dictionary from the retrieved patterns and translate documents in the same domain as the original corpus.

In the next section, we briefly describe a pattern-based machine translation. The following section explains how the proposed method works in detail. We then present experimental results and conclude with a discussion.

2 Pattern-based MT system

A pattern-based MT system uses a set of bilingual patterns(CFG rules) (Abeille et al., 1990) (Takeda, 1996) (Shimohata et al., 1999). In the parsing process, the engine performs a CFG-parsing for an input sentence and rewrites trees by applying the source patterns. Terminals and non-terminals are processed under the same framework but lexicalized patterns have priority over symbolized patterns [1] . A plausible parse

[1] We define a symbolized pattern as a pattern without a terminal and a lexicalized pattern as that with more than one terminal. we prepares 1000 symbolized patterns and 130,000 lexicalized patterns as a system

tree will be selected among possible parse trees by the number of patterns applied. Then the parse tree is transferred into target language by using target patterns which correspond to the source patterns.

Figure 1 shows an example of translation patterns between English and Japanese. Each English pattern(a left-half CFG rule) has corresponding Japanese pattern(a right-half CFG rule). Non-terminals are bracketed with index numbers which represents correspondence of non-terminals between the source and target pattern.

S	←[1:NP] [2:VP]	S	←[1:NP] が(subj) [2:VP]
NP	← a [1:NP]	NP	←[1:NP]
VP	←[1:VT] [2:NP]	VP	←[2:NP] を(dobj) [1:VT]
VP	← take [1:NP]	VP	←[1:NP] を(dobj) する(“do”)
VP	← take a bath	VP	←風呂(“bath”) に(“in”) 入る(“enter”)
V	← take	V	←とる(“take”)
N	← bath	N	←風呂(“bath”)

Figure 1: translation patterns

The pattern format is simple but highly descriptive. It can represent complicated linguistic phenomena and even correspondences between the languages with quite different structures. Furthermore, all the knowledge necessary for the translation, whether syntactic or lexical, are compiled in the same pattern format. Owing to these features, we can easily apply the retrieved technical terms to a real MT system.

3 Algorithm

Figure 2 shows an outline of the proposed method. The input is an untagged monolingual corpus, while the output is a domain dictionary for machine translation. The process is comprised of 3 phases: retrieving local patterns, assigning their syntactic categories with part-of-speech(POS) templates, and making translation patterns. The dictionary is used when an MT system translates a text in the same domain as the corpus.

We assume that the input is an English corpus and the dictionary is used for an English-Japanese MT system. In the remainder of this section, we will explain each phase in detail with English and Japanese examples.

dictionary.

3.1 Retrieving local patterns

We have already proposed a method for retrieving word sequences (Shimohata et al., 1997). This method generates all n-character (or n-word) strings appearing in a text and filters out fragmental strings with the distribution of words adjacent to the strings. This is based on the idea that adjacent words are widely distributed if the string is meaningful, and are localized if the string is a substring of a meaningful string.

The method introduces entropy value to measure the word distribution. Let the string be str, the adjacent words $w_1...w_n$, and the frequency of str $freq(str)$. The probability of each possible adjacent word $p(w_i)$ is then:

$$p(w_i) = \frac{freq(w_i)}{freq(str)} \tag{1}$$

At that time, the entropy of str $H(str)$ is defined as:

$$H(str) = \sum_{i=1}^{n} -p(w_i)\log p(w_i) \tag{2}$$

Calculating the entropy of both sides of str, the lower one is used as $H(str)$. Then the strings whose entropy is larger than a given threshold are retrieved as local patterns.

3.2 Identifying syntactic categories

Since the strings are just word sequences, the process gives them syntactic categories. For each str str,

1. assign part-of-speech tags t_1, ... t_n to the component words w_1, ... w_n
2. match tag sequence t_1, ... t_n with part-of-speech templates T_i
3. give str corresponding syntactic category SC_i, if it matches T_i

3.2.1 Assigning part-of-speech tags

The process uses a simplified part-of-speech set shown in table 1. Function words are assigned as they are, while content words except for adverb are fallen into only one part of speech **word**. Four kinds of words “*be*”, “*do*”, “*not*”, and “*to*” are assigned to special tags **be**, **do**, **not**, and **to** respectively.

There are several reasons to use the simplified POS tags:

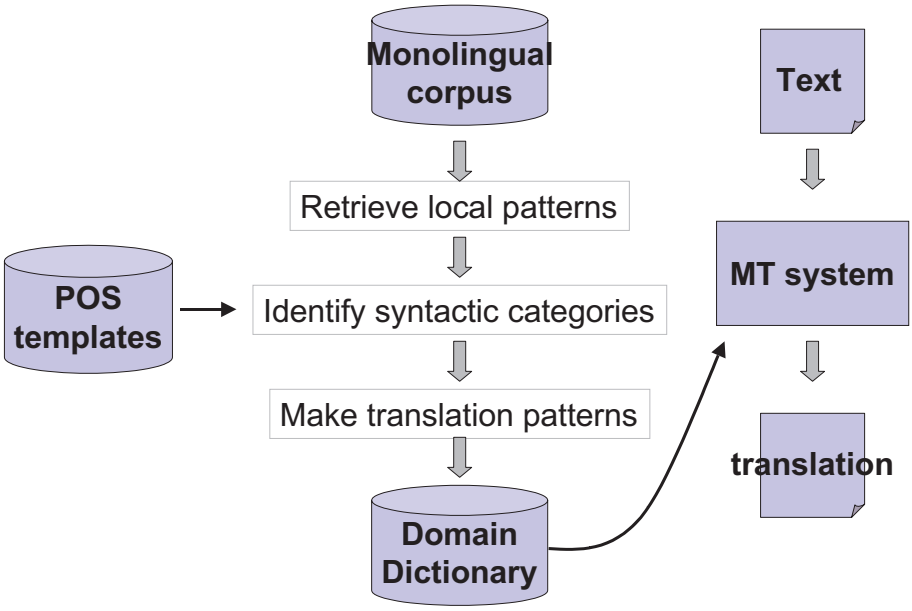

Figure 2: outline

POS tag	part of speech
art	article
adv	adverb
aux	auxiliary verb
conj	conjunction
det	determiner
prep	preposition
prn	pronoun
punc	punctuation
be	"*be*"
do	"*do*"
not	"*not*"
to	"*to*"
word	the others

Table 1: part-of-speech tags

- it may sometimes be difficult to identify precise parts of speech in such a local pattern.
- words are often used beyond parts of speech in technical terminology
- it is empirically found that word sequences retrieved through n-gram statistics have distributional concentration on several syntactic categories.

Therefore, we think the simplified POS tags are sufficient to identify syntactic categories.

The word sequence w_1, ... w_n is represented for a part-of-speech tag sequence t_1, ... t_i. Figure 3 shows examples of POS tagging. Italic

the fuel tank
art word word

do this step :
do det,prn word punc

to oprn the
to word art

Figure 3: examples of POS tagging

lines are given word sequences and bold lines are POS tag sequences. If a word falls into two or more parts of speech, all possible POSs will be assigned like "this" in the second example.

3.2.2 Matching POS templates

The process identifies a syntactic category(SC) of *str* by checking if *str*'s tag sequence t_1, ... t_n matches a given POS template T_i. If they

match, *str* is given a syntactic category SC_i corresponding to T_i. Table 2 shows examples of POS templates and corresponding SCs [2] .

SC	POS template
N	$(art)\,(word \mid conj)*\,(word)$
N+prep	$(art)\,(word)+\,(prep \mid to)\,(art)*$
VT	$(aux \mid to \mid prn)*\,(word)+\,(art)$
V-ed	$(be)\,(word)+\,(prep)\,(art)*$
V	$(aux \mid to \mid prn)\,(word)$
FUNC	$(art \mid aux \mid conj \mid det \mid prep \mid prn)+$

Table 2: POS templates and corresponding SCs

The templates are described in the form of regular expressions(RE) [3] . The first template in table 2, for example, matches a string whose tag sequence begins with an article, contains 0 or more repetitions of content words or conjunctions, and ends with a content word. "the fuel tank" in figure 3 is applied to this templates and given a SC "N".

3.3 Making translation patterns

The process converts the strings into translation patterns. The problem here is that we need to generate bilingual translation patterns from monolingual strings. We use heuristic rules on borrowing words from foreign languages [4] .

Figure 4 is an example of conversion rules for generating English-Japanese translation patterns. To give an example, "to open the" in figure 3, whose SC is VT, is converted into the following patterns in accordance with the second rule in figure 4.

VP ← open [1:NP]
VP ← [1:NP] を(dobj) open する("do")

If SC is N, delete **art** and generate:
NP ← *str*
NP ← *str*

If SC is VT, delete (**aux|to|prn**) and **art** and generate:
VP ← *str* [1:NP]
VP ← [1:NP] を(dobj) *str* する("do")

If SC is V, delete (**aux|to|prn**) generate:
V ← *str*
V ← *str* する("do")

Figure 4: conversion rules for generating translation patterns

4 Evaluation

We have tested our algorithm in building a domain dictionary and making a translation with it. A corpus used in the experiment is a computer manual comprising 167,023 words (in 22,041 sentences).

The corpus contains 24,137 n-grams which appear more than twice. Among them, 7,616 strings are extracted over the entropy threshold 1. Table 3 is a list of top 20 strings (except for single words and function word sequences) retrieved from the test corpus.

These strings are categorized into 1,239 POS patterns. Table 4 is a list of top 10 POS patterns and the numbers of strings classified into them. In this experiment, the top 10 POS patterns account for 49.4 % of all POS patterns. It substantiates the fact that the retrieved strings tend to concentrate in certain POS patterns.

freq	POS
1886	word
553	word word
368	art word
229	art word word
160	word prep
158	word art
121	word word word
108	to word
101	prep art word
81	prep word

Table 4: top 10 POS patterns

[2] Note that the POS templates are strongly dependent on the features of n-gram strings.

[3] "*" causes the resulting RE to match 0 or more repetitions of the preceding RE. "+" causes the resulting RE to match 1 or more repetitions of the preceding RE. "|" creates a RE expression that will match either right or left of "|". "(...)" indicates the start and end of a group.

[4] In Japanese, foreign words, especially in technical terminology, are often used as they are in *katakana* (the phonetic spelling for foreign words) followed by function words which indicate their parts of speech For example, English verbs are followed by "*suru*", a verb which means "*do*" in English.

$H(str)$	$freq(str)$	str	$H(str)$	$freq(str)$	str
5.51	247	see also	3.55	552	the client
4.48	1499	the server	3.54	209	use the
4.46	100	click OK .	3.46	209	the user
3.92	106	use this function	3.46	168	click the
3.79	163	the function	3.44	172	the catalog agent
3.76	309	the following	3.36	192	the request
3.67	297	the file	3.29	132	on page
3.58	36	in the Server Manager ,	3.23	213	a specified
3.56	169	using the	3.22	71	if you want to
3.55	180	CGI programs	3.22	575	your server

Table 3: top 20 strings

In the matching process, we prepared 15 templates and 6 SCs. Table 5 is a result of SC identification. 2,462 strings(32.3 %) are not matched to any templates. The table indicates that most strings retrieved in this method are identified as N and NP. It is quite reasonable because the majority of the technical terms are supposed to be nouns and noun phrases.

SC	number of patterns
NP	722
N+prep	200
VP	32
VP+prep	10
VT	177
V	78

Table 5: result of SC identification

The retrieved translation patterns total 1,219. Figure 5 shows an example of translation patterns retrieved by our method.

We, then, converted them to an MT dictionary and made a translation with and without it. Table 6 summarizes the evaluation results translating randomly selected 1,000 sentences from the test corpus. Compared with the translations without the dictionary, the translations with the dictionary improved 571 in parsing and word selection.

Figure 6 illustrates changes in translations. Each column consists of an input sentence, a translation without the dictionary, and a translation with the dictionary. Bold English words correspond to underlined Japanese.

improved in parsing	104
improved in word selection	467
about the same	160
same	212
not improved	57
total	1000

Table 6: Translation evaluation results

First two examples show improvement in word selection. The translations of "*map*(verb)" and "*exec*" are changed from word-for-word translations to non-translation word sequences. Although "to make a map" and "exective" are not wrong translations, they are irrelevant in the computer manual context. On the contrary, the domain dictionary reduces confusion caused by the wrong word selection.

Wrong parsing and incomplete parsing are also reduced as shown in the next two examples. In the third example, "*Next*" should be a noun, while it is usually used as an adverb. The domain dictionary solved the syntactic ambiguity properly because it has exclusive priority over system dictionaries. In the forth example, "*double-click*" is an unknown word which could cause incomplete parsing. But the phrase was parsed as a verb correctly.

The last one is an wrong example of Japanese verb selection. That was a main cause of errors and declines. The reason why the undesirable Japanese verbs were selected is that

NP ← fully-qualified domain name	NP ← fully-qualified domain name
NP ← text search engine	NP ← text search engine
NP ← access log for [1:NP]	NP ← [1:NP] の(“of”) access log
VP ← save [1:NP]	VP ← [1:NP] を(dobj) save する(“do”)
V ← deallocate	V ← deallocate する(“do”)

Figure 5: the retrieved translation patterns

Type the URL prefix you want to **map**.
あなたが地図("map")を(dobj)作り("make") たいURL 接頭辞をタイプしなさい。
あなたがmap し(“perform mapping”)たいURL prefix をタイプしなさい。

The **exec tag** allows an HTML file to execute an arbitrary program on the server;
幹部タグ(“exective‘s tag”) はサーバーで HTML ファイルが任意なプログラムを実行するのを許す;
exec tag は HTML ファイルが server の任意なプログラムを実行するのを許す;

Type the full name of your server, and then click **Next**.
次に("then") あなたのサーバーの完全な名前をタイプしてクリックしなさい。
あなたの server の完全な名前をタイプして Next を click しなさい。

Go to the Control Panel and **double-click** the Services icon.
Control Panel へ行きなさい、そうすれば二重な・("double-") は Services アイコンをクリックする("click") 。
Control Panel へ行って Services icon をdouble-click し("double-click")なさい。

Setting additional document directories
追加のドキュメントディレクトリを置く(“put, place”) こと
追加の document に directory を課する("assign, impose")こと

Figure 6: example sentences in the test corpus

the method added default semantic information to the retrieved nouns and noun phrases. We hope to overcome it by a model that classifies noun phrases, for example using verb-noun or adjective-noun relations.

5 Related work

As mentioned in section 1, there are two approaches in corpus-based technical term retrieval: a rule-based approach and a statistical approach. Major differences between the two are:

- the former uses a tagged corpus while the latter uses an untagged one.
- the former retrieves words and phrases with a designated syntactic category while the latter retrieves that with various syntactic categories at the same time.

Our method uses the latter approach because we think it more practical both in resources and in applications.

For comparison, we refer here to Smadja's method (1993) because this method and the proposed method have much in common. In both cases, technical terms are retrieved from an untagged corpus with n-gram statistics and given syntactic categories for NLP applications. The methods are different in that Smadja uses a

parser for syntactic category identification while we use POS templates. A parser may add more precise syntactic category than POS templates. However, we consider it not to be critical under the specific condition that the variety of input patterns is very small. In terms of portability, the proposed method has an advantage. Actually, adding POS templates is not so time consuming as developing a parser.

We have applied the translation patterns retrieved by this method to a real MT system. As a result, 57.1 % of translations were improved with 1,219 translation patterns. To our knowledge, little work has gone into quantifying its effectiveness to NLP applications. We recognize that the method leaves room for improvement in making translation patterns. We, therefore, plan to introduce techniques for finding translational equivalent from bilingual corpora (Melamed, 1998) to our method.

6 Conclusion

We have presented a method for identifying technical terminology and building a domain dictionary for MT. Applying the method to a technical manual in English yielded positive results. We have found that the proposed method would dramatically improve the performance of translation. In the future work, we plan to investigate the availability of POS patterns which are not categorized into any SCs.

References

Abeille A., Schabes Y., and Joshi A. K. 1990. "Using Lexicalized Tags for Machine Translation". In *Proceedings of the International Conference on Computational Linguistics(COLING)*, pages 1–6.

Argamon, S., Dagan, I., and Krymolowski, Yuval. 1999. A Memory-Based Approach to Learning Shallow Natural Language Patterns. In *Proceedings of the 17th COLING and the 36th Annual Meeting of ACL*, pages 67–73.

Cardie, C. and Pierce, D. 1999. The Role of Lexicalization and Pruning for Base Noun Phrase Grammars In *Proceedings of the 16th National Conference on Artificial Intelligence*, pages 423–430.

Haruno, M., Ikehara, S., and Yamazaki, T. 1996. Learning Bilingual Collocations by Word-Level Sorting. In *Proceedings of the 16th COLING*, pages 525–530.

Melamed,I.D. 1998. Empirical Methods for MT Lexicon Development In Gerber, L. and Farwell, D. Eds. *Machine Translation and the Information Soup*, Springer-Verlag.

Ramshaw, L.A., and Marcus, M.P. 1995. Text Chunking using Transformation-Based Learning In *Proceedings of the 3rd Workshop on Very Large Corpora*, pages 82–94.

Shimohata,S., Sugio,T., and Nagata,J. 1997. Retrieving Collocations by Co-occurrences and Word Order Constraints. In *Proceedings of the 35th Annual Meeting of ACL*, pages 476–481.

Shimohata, S. et al. 1999. "Machine Translation System PENSEE: System Design and Implementation," In *Proceedings of Machine Translation Summit VII*, pp.380–384.

Smadja,F.A. 1993. Retrieving Collocations from Text: Xtract. In *Computational Linguistics, 19(1)*, pages 143–177.

Takeda K. 1996. "Pattern-Based Context-Free Grammars for Machine Translation". In *Proceedings of the 34th Annual Meeting of ACL*, pages 144–151.