An Iterative Algorithm for Translation Acquisition of Adpositions

KANAYAMA Hiroshi
Tokyo Research Laboratory, IBM Japan, Ltd.
1623-14 Shimo-tsuruma, Yamato-shi, Kanagawa 242-8502, Japan
kanayama@trl.ibm.co.jp

Abstract

This paper describes an algorithm which acquires prepositions for translation from large corpora. Corpora of both the source language and the target language are used, but they can be independent of each other. Moreover, the algorithm does not require any type of manual tagging. Using an iterative algorithm, the system selects preferred prepositions between specific verbs and nouns in the target language, and simultaneously detects compound verbs which may be obstacles to the proper selection of prepositions. This algorithm is applied for the translation of the Japanese postposition '*de*' into English.

1 Introduction

The basic goal of this research is to construct a corpus-based method for machine translation with the following two desirable characteristics:

1. *Resources for the model are easy to acquire.* Available resources are limited if sentence-aligned bilingual corpora or semantically tagged corpora are required. On the other hand, untagged monolingual corpora are much easier to obtain.

2. *Knowledge can be added by hand without distorting the consistency of the model.* Statistical models tend to be difficult for humans to adjust directly, because arbitrary change of resources make the model inconsistent. Thus models into which humans can integrate their knowledge without changing the resources are desirable.

This paper proposes a method to select proper translations of the Japanese postposition '*de*' into English prepositions. The method has the above two traits, since it uses monolingual corpora without tagging by humans, and the iterative algorithm allows us to integrate linguistic knowledge at every stage in training.

Several corpus-based methods that use bilingual or monolingual corpora have been proposed in order to overcome the problem that a high-quality machine translation system requires too much knowledge for humans to describe it comprehensively. One of the methods that most drastically reduce the knowledge which must be described by humans is a fully statistical method (Brown et al., 1993). However, the method is difficult to apply to language pairs such as Japanese and English because word-to-word correspondences in these languages are not as firm as in closer language pairs such as English and French.

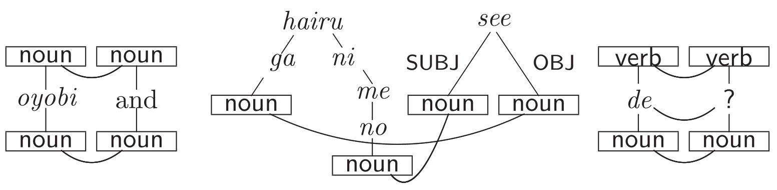

Figure 1: Patterns which should be extracted from bilingual corpora

Japanese			English		
ginkou “bank”	*de*	*au* “meet”	meet	at	the bank
kanazuchi “hammer”	*de*	*kowasu* “break”	break	with	a hammer
Osaka “Osaka”	*de*	*kurasu* “live”	live	in	Osaka

Table 1: Examples of translation of ‘*de*’

One of the feasible approaches for Japanese to English translation is automatic extraction of phrase-to-phrase correspondences from bilingual corpora (Watanabe et al., 2000). Their approach can extract any type of structural correspondences from sentence-aligned bilingual corpora, and also meets the second criterion described above, because one can intentionally add bilingual sentences which do not appear in the original corpora so that the system can find new bilingual correspondences.

Of course complete knowledge including lexical information is hard to get when relying only on bilingual corpora. Therefore, correspondences extracted from corpora should be generalized as in Figures 1, removing the specific lexical information from words which have bilingual correspondences. In Fig. 1(a) and (b), some content words are generalized and denoted as $\boxed{\text{noun}}$. After the target structure is constructed, content words in the target structure are specified as translations of related source words by using a bilingual dictionary. In Fig. 1(c), a preposition between $\boxed{\text{verb}}$ and $\boxed{\text{noun}}$ in the target phrase is not specified. This is because the Japanese postposition ‘*de*’ is translated in various ways depending on the noun and verb around it. Table 1 shows examples of the translation of ‘*de*’. ‘*De*’ doesn’t have a typical English translation, while some other postpositions do, like ‘*wo*’ which usually corresponds to a marker of the direct object.

Our algorithm selects an appropriate preposition for the translation of ‘*de*’ by collecting verb-preposition-noun tuples as underlined words in the right part of Table 1. Henceforth we call such tuples *VPN-tuples.* A VPN-tuple (v_e, p_e, n_e) consists of a verb v_e, a preposition p_e and a noun n_e where p_e modifies v_e and n_e follows p_e. Stemmed forms are used for v_e and n_e.

This paper relies on a translation system based on the corpus-based approach illustrated in Fig. 2, and the method in this paper focuses on step Ⓒ, the selection of prepositions in the target language, as the translation of *‘de’.*

In Section 2, some research on knowledge acquisition from monolingual corpora is reviewed. Section 3 presents the proposed algorithm, and its performance is evaluated in Section 4.

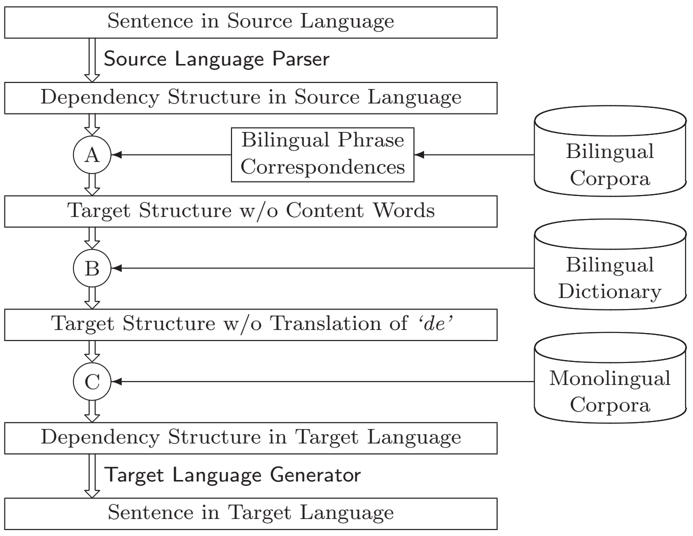
Sentence in Source Language
Source Language Parser
Dependency Structure in Source Language
A
Bilingual Phrase Correspondences
Bilingual Corpora
Target Structure w/o Content Words
B
Bilingual Dictionary
Target Structure w/o Translation of *'de'*
C
Monolingual Corpora
Dependency Structure in Target Language
Target Language Generator
Sentence in Target Language

Figure 2: Flow of translation system used in this paper

2 Related Work

Monolingual corpora contain useful information for machine translation and they are much easier to collect than bilingual corpora. Using this advantage, several methods which are based on monolingual corpora have been proposed.

Dagan and Itai (1994) showed that collocations of words in the target language can disambiguate word sense in the source language. They extract pairs of words in relation such as subject-verb or verb-object from corpora, and their frequencies are used for target word selection. The metric of statistical significance which they use is also used in the algorithm proposed in the next section.

Koehn and Knight (2000) used the EM algorithm based on translation probabilities of words and the bigram language model in the target language. Their approach handles subtle distinction in choosing of target words, however, different algorithm is required for the translation of adpositions because their bigram model is applicable to only noun sequences.

Yarowsky (1995) proposed an iterative algorithm, which gradually enhances the precision and the coverage of word-sense disambiguation, assuming "one sense per collocation" and "one sense per discourse". An ambiguous word is classified into two groups by using constraints involving collocated words, and such constraints are recognized and added based on the assumption that each word is used in one unique sense in a discourse. Since his algorithm starts with some seeds prepared by humans, linguistic knowledge is reflected in the approach. Thus his algorithm meets the two criteria described in Section 1. Though our algorithm in this paper is also iterative, the same method as Yarowsky's cannot be used, because the "one sense per discourse" assumption does not hold in the selection of prepositions. The next section presents our algorithm for the selection of prepositions.

V	P	N	frequency
come	at	average	1
come	in	average	1
come	with	average	1
compare	in	average	5
compare	on	average	8
compare	with	average	115
decline	on	average	2
do	on	average	1
exceed	by	average	1
exceed	on	average	2
exist	on	average	2
expect	on	average	15
⋮	⋮	⋮	⋮

Table 2: Extracted VPN-tuples with noun = `‘average’`

3 The Iterative Algorithm

3.1 Overview of Our Algorithm

Our system selects an appropriate English preposition between a verb and a noun as the translation of a Japanese verb phrase which consists of a noun, a postposition ‘*de*’ and a verb. As described in Section 1, this algorithm extracts VPN-tuples from an English monolingual corpus, and selects the most frequent preposition for a verb-noun pair (a *VN-tuple* henceforth).

This simple idea has some difficulties. There are exceptional usages of prepositions in corpora. Moreover, some parsing errors occur when sentences in an untagged corpus are parsed automatically. Thus the model should have the ability to ignore some VPN-tuples. In order to solve this problem, we adopt the *dynamic threshold* which Yarowsky (1995) used. This approach is described in Section 3.3.

In spite of using a large corpus, not enough VPN-tuples can be collected for every VN-tuple. To deal with this data-sparseness problem, the algorithm decides on a preposition relying only on the nouns for VN-tuples which do not appear frequently. In addition, some data is noisy: Table 2 shows VPN-tuples whose noun is `‘average’`, and their frequencies. For most verbs, `‘average’` appears most frequently with `‘on’`. Actually, Japanese ‘*heikin* (“average”) *de*’ is translated into ‘on average’ in most cases. However, the VPN-tuple (`‘compare’, ‘with’, ‘average’`) is so prominent that the most frequent preposition before `‘average’` is not `‘on’`. Taking into consideration the frequent appearance of `‘compare’` with `‘with’` regardless of the following noun, VPN-tuples which contain (`‘compare’, ‘with’`) should be removed from the VPN-tuple list when the preferred prepositions are selected for each noun. The iterative algorithm detects such verb-preposition pairs, which we name *VP-compounds*. If VPN-tuples which contain one of VP-compounds are removed, the precision of selecting prepositions for the nouns is improved, and the improved results can help detect new VP-compounds. This iterative method is described in Section 3.4.

3.2 Extraction of VPN-tuples

We construct a set of VPN-tuples E_{vpn} using both an English corpus and a Japanese corpus.

First, sentences in an English monolingual corpus are parsed by a syntactic parser and VPN-tuples are extracted from the parse trees. For example, from a sentence 'She went to the store by bus.', two VPN-tuples (`'go'`, `'to'`, `'store'`) and (`'go'`, `'by'`, `'bus'`) are extracted. All words are converted to stems and determiners are dropped.

Next, we extract VN-tuples which can be used in the translation of Japanese verb phrases which have the postposition '*de*'. That is, verb phrases which contain '*de*' are extracted from a Japanese monolingual corpus, and the noun-verb pairs in the phrases are translated into English VN-tuples by using a partial translation system[1]. For example, from the Japanese sentence '*Kare wa niwa de hashiru.*("He runs in the yard.")' a VN-tuple (v_e, n_e) = (`'run'`, `'yard'`) is extracted, because the original Japanese sentence has the phrase '*niwa* ("yard") *de hashiru* ("run")'. D_{vn} is the set of VN-tuples extracted by processing the Japanese corpus.

Finally, the VPN-tuples in the English corpus are restricted to those which can be candidates for the translation of the Japanese phrases with the postposition '*de*'. Only VPN-tuples whose verb and noun are in the set D_{vn} are chosen, and p_e is restricted to `'in'`, `'on'`, `'at'`, `'by'`, `'with'` and `'through'`. This is because more than 93% of the uses of the Japanese postposition '*de*' can be translated into one of these six prepositions[2]. In addition, VPN-tuples derived from passive sentences are removed since `'by'` is frequently found in passive sentences, but does not correspond to Japanese '*de*' in that usage.

Expressed in set notation, E_{vpn} is restricted as

$$E_{vpn} = \{(v_e, p_e, n_e) \mid (v_e, n_e) \in D_{vn}, p_e \in \{\texttt{'in'}, \texttt{'on'}, \texttt{'at'}, \texttt{'by'}, \texttt{'with'}, \texttt{'through'}\}\} \tag{1}$$

3.3 Determination of P for VN-tuples

The preferred preposition between a verb and a noun is determined by the frequencies of the VPN-tuples in E_{vpn}. If $f(v_e, p_e, n_e)$ is the frequency of a VPN-tuple (v_e, p_e, n_e) in E_{vpn}, then the preferred preposition for a VN-tuple $\pi(v_e, n_e)$ is calculated as follows:

$$\pi(v_e, n_e) = \arg\max_{p_e} \frac{f(v_e, p_e, n_e)}{\sum_p f(v_e, p, n_e)} \tag{2}$$

However, the result of Equation (2) is not reliable when $f(v_e, p_e, n_e)$ is not high enough or when the frequency of $\pi(v_e, n_e)$ is not high enough compared with that of

[1] The partial translation system consists of a Japanese parser, a bilingual phrase transducer and a bilingual dictionary. Therefore, the system outputs 'Target Structure w/o Translation of '*de*' ' shown in Figure 2.

[2] 326 instances of '*de*' in the Japanese newspaper articles are manually examined. Some exceptions are excluded such as '*koko de* ("here")' because the system has rules for such adverbial translations.

```
 1 $i := 0$;
 2 iterate
 3     $S_n^{(i)} := \phi; X := \phi; C_{vn}^{(i)} := \phi; E_{vpn}^{(i)} := E_{vpn}$;
 4     for each VPN-tuple $(v_e, p_e, n_e)$ in $E_{vpn}^{(i)}$
 5         for $j := 0$ to $i - 1$
 6             for each VP-compound $(v_c, p_c)$ in $C_{vp}^{(j)}$
 7                 if $v_e = v_c$ and $p_e = p_c$
 8                     remove $(v_e, p_e, n_e)$ from $E_{vpn}^{(i)}$;
 9             end for
10         end for
11     end for
12     for each noun $n_e$
13         compute $\pi^{(i)}(n_e)$;
14         if $\pi^{(i)}(n_e)$ exceeds the dynamic threshold
15             add $n_e$ to $S_n^{(i)}$;
16     end for
17     if $S_n^{(i)} = S_n^{(i-1)}$ exit iterate
18     for each VN-tuple $(v_e, n_e)$ in $S_{vn}$
19         if $n_e \in S_n^{(i)}$ and $\pi^{(i)}(n_e) \neq \pi(v_e, n_e)$
20             add $(v_e, n_e)$ to $X$;
21     end for
22     for each VN-tuple $(v_e, n_e)$ in $X$
23         for each VN-tuple $(v'_e, n'_e)$ in $X$
24             if $v_e = v'_e$ and $\pi(v_e, n_e) = \pi(v'_e, n'_e)$
25                     and $\pi^{(i)}(n_e) = \pi^{(i)}(n'_e)$
26                 add $(v_e, \pi(v_e, n_e))$ to $C_{vn}^{(i)}$;
27         end for
28     end for
29     $i$ :=i + 1;
30 end iterate
```

Figure 3: The pseudo-code of the iterative algorithm

other prepositions. In order to make reliable decisions, we adopt the *dynamic threshold* used by Yarowsky (1995). Taking into consideration the second most frequent preposition for a VN-tuple $\pi_2(v_e, n_e)$, the threshold is computed as

$$\ln\left(\frac{f_1}{f_2}\right) - Z_{1-\alpha}\sqrt{\frac{1}{f_1} + \frac{1}{f_2}} > \theta \tag{3}$$

where $f_1 = f(v_e, \pi(v_e, n_e), n_e)$, $f_2 = f(v_e, \pi_2(v_e, n_e), n_e)$, $Z_{1-\alpha}$ is the confidence coefficient, and θ is the threshold. Here we set $\alpha = 0.1$ ($Z_{1-\alpha} = 1.282$) and $\theta = 0.2$. This means the log odds ratio $\ln(\frac{f_1}{f_2})$ exceeds the threshold 0.2 with the confidence that 90% of the hypotheses are true.

The preposition is decided as $\pi(v_e, n_e)$ only for VN-tuples which satisfy Equation (3), and S_{vn} is defined as the set of VN-tuples for which the best preposition is successfully selected.

3.4 Determination of P for N and Detection of VP-Compounds

In the above process, prepositions are determined only for a limited number of VN-tuples. To cover the other VN-tuples, we find the preferred preposition for each noun, ignoring the verbs. This backup method works because the correct translation of Japanese postposition '*de*' often depends only on the noun before '*de*'.

Prepositions for specific nouns are determined by the iterative algorithm illustrated in Figure 3. Each step of the algorithm is described below.

In Line 3, $E_{vpn}^{(i)}$ is set to E_{vpn}. Line 4–11 is skipped here since nothing is done in the first pass. In Line 13, the preferred preposition for a noun is computed as

$$\pi^{(i)}(n_e) = \arg\max_{p_e} \frac{\sum_v f^{(i)}(v, p_e, n_e)}{\sum_p \sum_v f^{(i)}(v, p, n_e)} \tag{4}$$

where $f^{(i)}(v_e, p_e, n_e)$ is the frequency of the VPN-tuple (v_e, p_e, n_e) in $E^{(i)}_{vpn}$. Also in this case, $\pi^{(i)}(n_e)$ is determined only when Equation (3) is satisfied, where $f_1 = \sum_v f^{(i)}(v, \pi^{(i)}(n_e), n_e)$ and $f_2 = \sum_v f^{(i)}(v, \pi^{(i)}_2(n_e), n_e)$. $S^{(i)}_n$ is defined as the set of the nouns whose preposition has been determined in the i-th stage.

As noted in Section 3.1, we remove VPN-tuples which contain one of VP-compounds, pairs consisting of a verb and a preposition which conflict with $\pi^{(i)}(n_e)$. VP-compounds are detected in Line 18–28. These lines mean that $C^{(i)}_{vn}$ is a set of (v_e, p_e) which satisfies the following condition:

$$\left.\begin{array}{l}(v_e, p_e) \text{ is a VP-compound in the } i\text{-th stage iff } \exists n_1 \exists n_2 \text{ such that} \\ \quad (v_e, n_1) \in S_{vn}, (v_e, n_2) \in S_{vn}, (n_1) \in S^{(i)}_n, (n_2) \in S^{(i)}_n, \\ \quad \pi(v_e, n_1) \neq \pi^{(i)}(n_1), \pi(v_e, n_2) \neq \pi^{(i)}(n_2), \pi(v_e, n_1) = \pi(v_e, n_2), \pi^{(i)}(n_1) \neq \pi^{(i)}(n_2)\end{array}\right\} \tag{5}$$

In the next stage, the set of VPN-tuples is reconstructed according to the VP-compounds: all VPN-tuples which contain a VP-compound are removed from $E^{(i)}_{vpn}$ in Line 4–11. This operation is also expressed as

$$E^{(i)}_{vpn} = E_{vpn} - \{(v_e, p_e, n_e) \mid (v_e, p_e) \in \bigcup_{x=0}^{i-1} C^{(x)}_{vp}\} \tag{6}$$

Then, the preposition for each n_e is recomputed by using $E^{(i)}_{vpn}$. This computation is repeated until $S^{(i)}_n$ converge to a fixed set.

3.5 Example

The following example clarifies how the iterative algorithm works.

First, some prepositions are specified for VN-tuples per Equations (2) and (3). For example, π(`‘compare’`, `‘second’`)=`‘with’`, π(`‘compare’`, `‘price’`)=`‘with’`. In the 0th stage of the iteration, prepositions for nouns are determined such as π(`‘second’`) = `‘in’`, π(`‘price’`)=`‘at’`, implying (`‘second’`) $\in S^{(0)}_n$ and (`‘price’`) $\in S^{(0)}_n$. However, (`‘average’`) $\notin S^{(0)}_n$, that is, the preferred preposition for (`‘average’`) failed to be determined, because the difference of the frequencies of `‘with’` and `‘on’` is not large enough before the noun `‘average’`.

After $S^{(0)}_n$ is computed, the VP-compounds are detected. In this case, $(v_e, p_e) =$ (`‘compare’, ‘with’`) is detected as a VP-compound, because it satisfies the condition in (5) for $n_1 =$ `‘second’`, $n_2 =$ `‘price’`.

In the next stage, all VPN-tuples which equal (`‘compare’`, `‘with’`, `‘average’`) are removed from $E^{(1)}_{vpn}$ because (`‘compare’`, `‘with’`) $\in C^{(0)}_{vp}$ (see Equation (6)). Due to the absence of (`‘compare’`, `‘with’`, `‘average’`), $\pi^{(1)}$(`‘average’`) is now specified as `‘on’`. Therefore, the iterative algorithm improved the decision because (`‘average’`) $\in S^{(1)}_n$ while (`‘average’`) $\notin S^{(0)}_n$.

Africa	in	absence	in	aim	with	approval	with
Alaska	in	access	with	air	in	area	in
:		accuracy	with	airline	with	arena	in
Beijing	in	acquisition	through	airport	at	arrival	with
Boeing	by	act	in	all	at	article	in
Brazil	in	addition	in	analysis	in	assembly	at
:		address	at	answer	with	assistance	with
U.K.	in	adjustment	with	apartment	in	average	on
Washington	in	advance	in	appearance	in	back	on
Wimbledon	at	age	at	approach	with	:	

Table 3: The prepositions determined for nouns in the 5th stage of the iteration (37 out of 555, in alphabetical order)

act	on	beat	by	call	on	combine	with
advise	on	become	with	capitalize	on	come	with
agree	on	begin	with	carry	on	comment	on
aim	at	bet	on	center	on	communicate	with
appear	on	blame	on	click	on	compare	with
approve	by	boost	by	close	at	compete	with
arrive	at	brief	on	close	on	comply	with
associate	with	build	in	coincide	with	concentrate	on
bank	on	build	on	collaborate	on	continue	with
base	on	buoy	by	comb	through	:	

Table 4: The VP-compounds detected at least once in the iterative algorithm (39 out of 172, in alphabetical order)

4 Evaluation

4.1 Experimental Results

We used an English corpus which consists of about 3.2 million sentences (70.2M words) extracted from a newspaper, and a Japanese corpus which consists of about 2.2 million sentences from another newspaper. A total of 2.3 million VPN-tuples were extracted by parsing the English sentences with ESG (McCord, 1980). The Japanese parser in our partial translation system extracted 497,720 tokens (280,257 types) of verb phrases which have the postposition '*de*'. Using the bilingual lexicon, we translated the Japanese verbs and nouns in the extracted phrases into the corresponding English VN-tuples. At this stage we had 203,739 types of VN-tuples for D_{vn} and 133,654 tokens of VPN-tuples for E_{vpn}. In this experiment, we constructed S_{vn} using VPN-tuples not restricted by D_{vn} instead of E_{vpn}, in order to determine the prepositions for more VN-tuples.

With the above E_{vpn} and S_{vn}, the iterative algorithm computed $S_n^{(i)}$, which converged when $i = 5$. The prepositions for nouns determined in the final stage of the iteration, and the VP-compounds which are detected at least once in the algorithm are exemplified in Table 3 and 4, respectively.

We used the selected prepositions for the Japanese-English translation of real-world sentences. The partial translation system translates a Japanese sentence into English

Iteration	$\lvert S_{vn} \rvert$	$\lvert S_n^{(i)} \rvert$	$\lvert \bigcup_{x=0}^{i-1} C_{vp}^{(x)} \rvert$	Coverage
0	25339	634	0	840 / 1321 (63.5%)
1		569	149	880 / 1321 (66.6%)
2		553	165	886 / 1321 (67.0%)
3		557	166	887 / 1321 (67.1%)
4		555	171	886 / 1321 (67.0%)
5		555	172	886 / 1321 (67.0%)

Table 5: The selection of prepositions using Rules **A** or **B** for 1321 VN-tuples. The middle three columns denote the number of elements of S_{vn}, $S_n^{(i)}$, and the VP-compounds detected before the i-th stage, respectively.

Iteration	Partial Precision	Total Precision
0	72 / 93 (77.4%)	94 / 146 (64.4%)
5	80 / 98 (81.6%)	100 / 146 (68.5%)
Baseline	-	61 / 146 (41.8%)

Table 6: The precision of the translation of '*de*' into prepositions in 146 trials. 'Baseline' denotes the precision where `'in'` is always selected.

but the preposition to be used as the translation of '*de*' is left unspecified unless the phrase matched with a special translation pattern. When (v_e, n_e) is the verb and noun in an incomplete target phrase, the system selects a preposition according to the following rules.

A If $(v_e, n_e) \in S_{vn}$, $\pi(v_e, n_e)$ is selected.

B Otherwise, if $(n_e) \in S_n^{(i)}$, $\pi^{(i)}(n_e)$ is selected.

C Otherwise, if n_e is found in the VPN-tuples extracted from the English corpus, the most frequent preposition preceding n_e is selected.

D Otherwise, `'in'` is selected.

To evaluate the effectiveness of our algorithm, we calculated the following two values for each step of the iteration:

- *Coverage.* The percentage of VN-tuples (v_e, n_e) for which the preposition was specified as $\pi(v_e, n_e)$ or $\pi^{(i)}(n_e)$ i.e. determined in case **A** or **B**.
- *Precision.* The percentage of VN-tuples for which one of the acceptable prepositions[3] was selected. *Partial precision* means the precision of selection in case **A** or **B** and *total precision* is the precision for the whole estimation process.

[3] There are one or more acceptable prepositions for each translation, since it is difficult to determine the correct translation uniquely.

Table 5 shows that the coverage increases gradually until the 3rd step of the iteration, even though the number of elements in $S_n^{(i)}$ decreases from the initial state by 65 nouns. This phenomenon in the 1st step of the iteration is explained as follows: Due to the set of VP-compounds $C_{vn}^{(0)}$, 64 nouns which are not in $S_n^{(0)}$ were newly added to $S_n^{(1)}$, and at the same time 129 nouns in $S_n^{(0)}$ failed to exceed the dynamic threshold in the 1st step. However, some nouns in the latter group tend to collocate with the specific verbs, thus most of their appearances were covered by S_{vn}, so did not cause decrease of the coverage.

Table 6 shows the precision after and before the iteration. Given that not only the coverage but also the partial precision was enhanced by the iteration, more appropriate prepositions are predicted for nouns by removing VPN-tuples which contain VP-compounds. For example, $\pi^{(1)}$(`'success'`) = `'on'` is improved as $\pi^{(2)}$(`'success'`) = `'with'`[4]. The preposition for `'success'` is once determined as `'on'` because of the frequent VPN-tuple (`'board'`, `'on'`, `'success'`), but after that a new VP-compound (`'board'`, `'on'`) is detected, then the preposition for `'success'` is changed in the next computation.

4.2 Discussion

This analysis focuses on some cases where prepositions were not determined correctly, classifying into the following types of problems. Errors caused by inappropriate translations of nouns and verbs are not intrinsic, so we don't consider them here.

Ambiguous nouns. For the phrase '*rijikai* ("board of directors") *de youkyuu-suru* ("demand")', the system searches for the preposition for (`'demand'`, `'board'`). Since the VN-tuple is not in S_{vn}, the preposition for (`'board'`) is searched and `'on'` is returned. Apparently the output is affected by the expression 'on board'. To avoid this error, we must specify the sense of `'board'` using a Japanese thesaurus.

Translation of other postpositions. 'Work for the company' is the correct translation of '*sono kaisha* ("company") *de hataraku* ("work")'. However, the VN-tuple (`'work'`, `'company'`) prefers `'with'` since expressions like '*sono kaisha to hataraku* ("work with the company")' appear frequently in the corpus. This error occurs because our algorithm does not take into consideration Japanese phrases with other postpositions. Some VPN-tuples which may be the translations of such Japanese phrases should be removed. If this mechanism works successfully, other prepositions like `'for'` can be added to the possible preposition list for the translation of '*de*'.

Wrong VP-compounds. Though $\pi^{(0)}$(`'approach'`) was `'with'` in the initial stage, `'approach'` was removed from $S_n^{(1)}$. This is because inappropriate VP-compounds such as (`'come', 'with'`) and (`'expect', 'with'`) were added to $C_{vp}^{(0)}$. The criterion to pick up VP-compounds should be reconsidered.

[4]'*Seikou* ("success") *de owatta* ("ended")' should be translated as 'ended *with* success', thus `'with'` is better preposition than `'on'` for `'success'`.

Our supplementary experiment shows that the VP-compound set can be adjusted manually. We removed the two VP-compounds (`'come'`, `'with'`) and (`'expect'`, `'with'`), and ran the iterative algorithm again, so `'with'` is then selected for some nouns such as `'approach'`, `'service'` and `'software'`.

As another way to enhance the model, we can add other VP-compounds, or specify prepositions for specific VN-tuples or nouns. Knowledge of VP-compounds and correct prepositions are complementary to each other, therefore the manual enhancement of either can help the other.

5 Conclusion

We have introduced an iterative algorithm for selecting prepositions as the translation of the Japanese postposition '*de*'. The unsupervised method needs only untagged corpora. In tests 81.6% of the uses are translated acceptably for VN-tuples which are covered by this algorithm. The experiment proved that the VP-compounds helped refine the selection of the prepositions. Though we have not manually optimized the system yet, the automatic method solved the problem with good precision. There is a room for manual enhancement of the VP-compounds or the prepositions liked to specific VN-tuples or nouns. Therefore, one can integrate linguistic knowledge which is not determined from the given corpora into the statistical model.

References

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.

Ido Dagan and Alon Itai. 1994. Word sense disambiguation using a second language monolingual corpus. *Computational Linguistics*, 20(4):563–596.

Philipp Koehn and Kevin Knight. 2000. Estimating word translation probabilities from unrelated monolingual corpora using the EM algorithm. In *Proc. of the 17th National Conference on Artificial Intelligence (AAAI-2000)*.

Michael C. McCord. 1980. Slot grammars. *Computational Linguistics*, 6:31–43.

Hideo Watanabe, Sadao Kurohashi, and Eiji Aramaki. 2000. Finding structural correspondences from bilingual parsed corpus for corpus-based translation. In *Proc. of the 18th International Conference on Computational Linguistics (COLING 2000)*, pages 906–912.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Meeting of the Association for Computational Linguistics*, pages 189–196.