A Comparative Study on Translation Units for Bilingual Lexicon Extraction

Kaoru Yamamoto† and **Yuji Matsumoto**† and **Mihoko Kitamura**†‡
Graduate School of Information Science, Nara Institute of Science and Technology†
8916-5 Takayama, Ikoma, Nara, Japan
{kaoru-ya, matsu, mihoko-k}@is.aist-nara.ac.jp
Research & Development Group, Oki Electric Industry Co., Ltd.‡
kita@kansai.oki.co.jp

Abstract

This paper presents on-going research on automatic extraction of bilingual lexicon from English-Japanese parallel corpora. The main objective of this paper is to examine various N-gram models of generating translation units for bilingual lexicon extraction. Three N-gram models, a baseline model (Bound-length N-gram) and two new models (Chunk-bound N-gram and Dependency-linked N-gram) are compared. An experiment with 10000 English-Japanese parallel sentences shows that Chunk-bound N-gram produces the best result in terms of accuracy (83%) as well as coverage (60%) and it improves approximately by 13% in accuracy and by 5-9% in coverage from the previously proposed baseline model.

1 Introduction

Developments in statistical or example-based MT largely rely on the use of bilingual corpora. Although bilingual corpora are becoming more available, they are still an expensive resource compared with monolingual corpora. So if one is fortune to have such bilingual corpora at hand, one must seek the maximal exploitation of linguistic knowledge from the corpora.

This paper presents on-going research on automatic extraction of bilingual lexicon from English-Japanese parallel corpora. Our approach owes greatly to recent advances in various NLP tools such as part-of-speech taggers, chunkers, and dependency parsers. All such tools are trained from corpora using statistical methods or machine learning techniques. The linguistic "clues" obtained from these tools may be prone to some error, but there is much partially reliable information which is usable in the generation of translation units from unannotated bilingual corpora.

Three N-gram models of generating translation units, namely Bound-length N-gram, Chunk-bound N-gram, and Dependency-linked N-gram are compared. We aim to determine characteristics of translation units that achieve both high accuracy and wide coverage and to identify the limitation of these models.

In the next section, we describe three models used to generate translation units. Section 3 explains the extraction algorithm of translation pairs. In Sections 4 and 5, we present our experimental results and analyze the characteristics of each model. Finally, Section 6 concludes the paper.

2 Models of Translation Units

The main objective of this paper is to determine suitable translation units for the automatic acquisition of translation pairs. A word-to-word correspondence is often assumed in the pioneering works, and recently Melamed argues that one-to-one assumption is not restrictive as it may appear in (Melamed, 2000). However, we question his claim, since the tokenization of words for non-segmented languages such as Japanese is, by na-

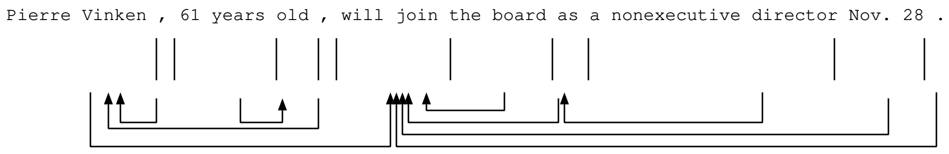

Figure 1: sample sentence

```
Pierre                             Vinken
Pierre-Vinken                      Vinken-years
Pierre-Vinken-years                Vinken-years-old
Pierre-Vinken-years-old            Vinken-years-old-join
Pierre-Vinken-years-old-join       Vinken-years-old-join-board

years                              old
years-old                          old-join
years-old-join                     old-join-board
years-old-join-board               old-join-board-nonexecutive
years-old-join-board-nonexecutive  old-join-board-nonexecutive-director

join                               board
join-board                         board-nonexecutive
join-board-nonexecutive            board-nonexecutive-director
join-board-nonexecutive-director   board-nonexecutive-director-Nov
join-board-nonexecutive-director-Nov.

nonexecutive                       director
nonexecutive-director              director-Nov
nonexecutive-director-Nov          Nov
```

Figure 2: Bound-length N-gram

ture, ambiguous, and thus his one-to-one assumption is difficult to hold. We address this ambiguity problem by allowing 'overlaps' in generation of translation units and obtain single- and multi-word correspondences simultaneously.

Previous works that focus on multi-word correspondences include (Kupiec, 1993) where NP recognizers are used to extract translation units and (Smadja et al., 1996) which uses the XTRACT system to extract collocations. Moreover, (Kitamura and Matsumoto, 1996) extracts an arbitrary length of word correspondences and (Haruno et al., 1996) identifies collocations through word-level sorting.

In this paper, we compare three N-gram models of translation units, namely Bound-length N-gram, Chunk-bound N-gram, and Dependency-linked N-gram. Our approach of extracting bilingual lexicon is two-staged. We first prepare N-grams independently for each language in the parallel corpora and then find corresponding translation pairs from both sets of translation units in a greedy manner. The essence of our algorithm is that we allow some overlapping translation units to accommodate ambiguity in the first stage. Once translation pairs are detected during the process, they are decisively selected, and the translation units that overlaps with the found translation pairs are gradually ruled out.

In all three models, translation units of N-gram are built using only content (open-class) words. This is because functional (closed-class) words such as prepositions alone will usually act as noise and so they are filtered out in advance.

A word is classified as a functional word if it matches one of the following conditions. (The Penn Treebank part-of-speech tag set (Santorini, 1991) is used for English, whereas the ChaSen part-of-speech tag set (Matsumoto and Asahara, 2001) is used for Japanese.)

part-of-speech(J) “名詞-代名詞”, “名詞-数”, “名詞-非自立”, “名詞-特殊”, “名詞-接尾-助動詞語幹”, “名詞-接尾-副詞可能”, “名詞-接尾-助動詞”, “接頭詞”, “動詞-接尾”, “動詞-非自立”, “助詞”, “助動詞”, “形容詞-非自立”, “形容詞-接尾”, “記号”

part-of-speech(E) “CC”, “CD”, “DT”, “EX”, “FW”, “IN”, “LS”, “MD”, “PDT”, “PR”, “PRS”, “TO”, “WDT”, “WD”, “WP”

stemmed-form(E) “be”

```
Pierre                                  Vinken
Pierre-Vinken
years                                   old
join                                    board
nonexecutive                            director
nonexecutive-director                   Nov
```

Figure 3: Chunk-bound N-gram

symbols punctuations and brackets

We now illustrate the three models of translation units by referring to the sentence in Figure 1.

Bound-length N-gram

Bound-length N-gram is first proposed in (Kitamura and Matsumoto, 1996). The translation units generated in this model are word sequences from uni-gram to a given length N. The upper bound for N is fixed to 5 in our experiment. Figure 2 lists a set of N-grams generated by Bound-length N-gram for the sentence in Figure 1.

Chunk-bound N-gram

Chunk-bound N-gram assumes prior knowledge of chunk boundaries. The definition of "chunk" varies from person to person. In our experiment, the definition for English chunk task complies with the CoNLL-2000 text chunking tasks and the definition for Japanese chunk is based on "bunsetsu" in the Kyoto University Corpus.

Unlike Bound-length N-gram, Chunk-bound N-gram will not extend beyond the chunk boundaries. N varies depending on the size of the chunks[1]. Figure 3 lists a set of N-grams generated by Chunk-bound N-gram for the sentence in Figure 1.

Dependency-linked N-gram

Dependency-linked N-gram assumes prior knowledge of dependency links among chunks. In fact, Dependency-linked N-gram is an enhanced model of the Chunk-bound model in that, Dependency-linked N-gram extends chunk boundaries via dependency links. Although dependency links could be extended recursively in a sentence, we limit the use to direct dependency links (i.e. links of immediate mother-daughter relations) only. Two chunks of dependency linked are concatenated and treated as an extended chunks. Dependency-linked N-gram generates translation units within the extended boundaries. Therefore, translation units generated by Dependency-linked N-gram (Figure 4) become the superset of the units generated by Chunk-bound N-gram (Figure 3).

```
Pierre                    Vinken
Pierre-Vinken             Vinken-join
Pierre-Vinken-join
years                     old
years-old                 Pierre-Vinken-old
join                      board
                          join-board
nonexecutive              director
nonexecutive-director
Nov.
join-Nov
```

Figure 4: Dependency-linked N-gram

The distinct characteristics of Dependency-linked N-gram from previous works are two-fold. First, (Yamamoto and Matsumoto, 2000) also uses dependency relations in the generation of translation units. However, it suffers from data sparseness (and thus low coverage), since the entire chunk is treated as a translation unit, which is too coarse. Dependency-linked N-gram, on the other hand, uses more fine-grained N-grams as translation units in order to avoid sparseness. Second, Dependency-linked N-gram includes "flexible" or non-contiguous collocations if dependency links are distant in a sentence. These collocations cannot be obtained by Bound-length N-gram with any N.

3 Translation Pair Extraction

We use the same algorithm as (Yamamoto and Matsumoto, 2000) for acquiring translation pairs. The algorithm proceeds in a greedy manner. This means that the translation pairs found earlier (i.e. at a higher threshold) in the algorithm are regarded as decisive entries. The threshold acts as the level of confidence. Moreover, translation units that partially overlap with the already found translation pairs are filtered out during the algorithm.

The correlation score between translation units

[1]The average number of words in English and Japanese chunks are 2.1 and 3.4 respectively for our parallel corpus.

model	English	Japanese
Bound-length	8479	11587
Chunk-bound	4865	5870
Dependency-linked	8716	11068

Table 1: Number of Translation Units

p_e and p_j is calculated by the weighted Dice Coefficient defined as:

$$sim(p_e, p_j) = (log_2 f_{ej}) \frac{2 f_{ej}}{f_e + f_j}$$

where f_j and f_e are the numbers of occurrences of p_e and p_j in Japanese and English corpora respectively, and f_{ej} is the number of co-occurrences of p_e and p_j.

We repeat the following until the current threshold f_{curr} reaches the predefined minimum threshold f_{min}.

1. For each pair of English unit p_e and Japanese unit p_j appearing at least f_{curr} times, identify the most likely correspondences according to the correlation scores.
   - For an English pattern p_e, obtain the correspondence candidate set PJ = { p_{j1}, p_{j2}, ..., p_{jn} } such that sim(p_e,p_{jk}) $> log_2\ f_{curr}$ for all k. Similarly, obtain the correspondence candidate set PE for a Japanese pattern p_j
   - Register (p_e,p_j) as a translation pair if

$$p_j = \underset{p_{jk} \in PJ}{\texttt{argmax}} sim(p_e, p_{jk})$$

$$p_e = \underset{p_{ek} \in PE}{\texttt{argmax}} sim(p_j, p_{ek})$$

     The correlation score of (p_e,p_j) is the highest among PJ for p_e and PE for p_j.

2. Filter out the co-occurrence positions for p_e, p_j, and their overlapped translation units.

3. Lower f_{curr} if no more pairs are found.

4 Experiment and Result

4.1 Experimental Setting

Data for our experiment is 10000 sentence-aligned corpus from English-Japanese business expressions (Takubo and Hashimoto, 1995). 8000 sentences pairs are used for training and the remaining 2000 sentences are used for evaluation.

Since the data are unannotated, we use NLP tools (part-of-speech taggers, chunkers, and dependency parsers) to estimate linguistic information such as word segmentation, chunk boundaries, and dependency links. Most tools employ a statistical model (Hidden Markov Model) or machine learning (Support Vector Machines).

f_{curr}	e	c	acc	e'	c'	acc'
100.0	0	0	n/a	0	0	n/a
50.0	0	0	n/a	0	0	n/a
25.0	1	1	1.0000	1	1	**1.0000**
12.0	10	9	0.9000	11	10	**0.9090**
10.0	9	9	1.0000	20	19	**0.9500**
9.0	7	7	1.0000	27	26	**0.9629**
8.0	10	10	1.0000	37	36	**0.9729**
7.0	12	11	0.9166	49	47	**0.9591**
6.0	25	25	1.0000	74	72	**0.9729**
5.0	29	28	0.9655	103	100	**0.9708**
4.0	70	68	0.9714	173	168	**0.9710**
3.0	114	109	0.9561	287	277	**0.9651**
2.0	646	490	0.7585	933	767	**0.8220**
1.9	58	54	0.9310	991	821	**0.8284**
1.8	67	60	0.8955	1058	881	**0.8327**
1.7	186	131	0.7043	1244	1012	**0.8135**
1.6	105	93	0.8857	1349	1105	**0.8191**
1.5	220	161	0.7318	1569	1266	**0.8068**
1.4	267	182	0.6816	1836	1448	**0.7886**
1.3	309	228	0.7378	2145	1676	**0.7813**
1.2	459	312	0.6797	2604	1988	**0.7634**
1.1	771	404	0.5239	3375	2392	**0.7087**

Table 2: Accuracy(Bound-Length N-gram)

Translation units that appear at least twice are considered to be candidates for the translation pair extraction algorithm described in the previous section. This implies that translation pairs that co-occur only once will never be found in our algorithm. We believe this is a reasonable sacrifice to bear considering the statistical nature of our algorithm. Table 1 shows the number of translation units found in each model. Note that translation units are counted not by token but by type.

We adjust the threshold of the translation pair extraction algorithm according to the following equation. The threshold f_{curr} is initially set to 100 and is gradually lowered down until it reaches the minimum threshold f_{min} 2, described in Section 3. Furthermore, we experimentally decrement the threshold f_{curr} from 2 to 1 with the remaining uncorrelated sets of translation units, all of which appear at least twice in the corpus. This means that translation pairs whose correlation score is $1 >$ sim(p_e,p_j) > 0 are attempted to find correspondences[2].

[2]Note that f_curr plays two roles: (1) threshold for the co-occurrence frequency, and (2) threshold for the correlation score. During the decrement of f_curr form 2 to 1,

f_{curr}	e	c	acc	e'	c'	acc'
100.0	1	1	1.0	1	1	**1.0**
50.0	13	12	0.9230	14	13	**0.9285**
25.0	26	25	0.9615	40	38	**0.95**
12.0	63	61	0.9682	103	99	**0.9611**
10.0	35	35	1.0	138	134	**0.9710**
9.0	20	20	1.0	158	154	**0.9746**
8.0	17	16	0.9411	175	170	**0.9714**
7.0	40	39	0.975	215	209	**0.9720**
6.0	38	37	0.9736	253	246	**0.9723**
5.0	84	84	1.0	337	330	**0.9792**
4.0	166	160	0.9638	503	490	**0.9741**
3.0	198	195	0.9848	701	685	**0.9771**
2.0	870	816	0.9379	1571	1501	**0.9554**
1.9	112	106	0.9464	1683	1607	**0.9548**
1.8	109	105	0.9633	1792	1712	**0.9553**
1.7	266	239	0.8984	2058	1951	**0.9480**
1.6	155	139	0.8967	2213	2090	**0.9444**
1.5	292	253	0.8664	2505	2343	**0.9353**
1.4	365	327	0.8958	2870	2670	**0.9303**
1.3	448	391	0.8727	3318	3061	**0.9225**
1.2	599	483	0.8063	3917	3544	**0.9047**
1.1	890	481	0.5404	4807	4025	**0.8373**

Table 3: Accuracy(Chunk-bound N-gram)

f_{curr}	e	c	acc	e'	c'	acc'
100.0	1	1	1.0	1	1	**1.0**
50.0	13	12	0.9230	14	13	**0.9285**
25.0	26	25	0.9615	40	38	**0.95**
12.0	62	60	0.9677	102	98	**0.9607**
10.0	32	31	0.9687	134	129	**0.9626**
9.0	20	23	1.15	158	152	**0.9620**
8.0	16	16	1.0	174	168	**0.9655**
7.0	43	43	1.0	217	211	**0.9723**
6.0	40	39	0.975	257	250	**0.9727**
5.0	85	83	0.9764	342	333	**0.9736**
4.0	166	162	0.9759	508	495	**0.9744**
3.0	205	201	0.9804	713	696	**0.9761**
2.0	949	849	0.8946	1662	1545	**0.9296**
1.9	115	107	0.9304	1777	1652	**0.9296**
1.8	105	103	0.9809	1882	1755	**0.9325**
1.7	268	230	0.8582	2150	1985	**0.9232**
1.6	156	145	0.9294	2306	2130	**0.9236**
1.5	288	244	0.8472	2594	2374	**0.9151**
1.4	373	300	0.8042	2967	2674	**0.9012**
1.3	434	344	0.7926	3401	3018	**0.8873**
1.2	576	383	0.6649	3977	3401	**0.8551**
1.1	855	417	0.4877	4832	3818	**0.7901**

Table 4: Accuracy(Dependency-linked N-gram)

$$f_{curr} = \begin{cases} f_{curr}/2 & (f_{curr} > 20) \\ 10 & (20 \geq f_{curr} > 10) \\ f_{curr} - 1 & (10 \geq f_{curr} \geq 2) \\ f_{curr} - 0.1 & (2 > f_{curr} > 1) \end{cases}$$

The result is evaluated in terms of accuracy and coverage. Accuracy is the number of correct translation pairs over the extracted translation pairs in the algorithm. This is calculated by type. Coverage measures "applicability" of the correct translation pairs for unseen test data. It is the number of tokens matched by the correct translation pairs over the number of tokens in the unseen test data. Acuracy and coverage roughly correspond to Melamed's precision and percent correct respectively (Melamed, 1995). Accuracy is calculated on the training data (8000 sentences) manually, whereas coverage is calculated on the test data (2000 sentences) automatically.

4.2 Accuracy

Stepwise accuracy for each model is listed in Table 2, Table 3, and Table 4. "f_{curr}" indicates the threshold, i.e. stages in the algorithm. "e" is the number of translation pairs found at stage "f_{curr}", and "c" is the number of correct ones found at stage "f_{curr}". The correctness is judged by an English-Japanese bilingual speaker. "acc" lists accuracy, the fraction of correct ones over extracted ones by type. The accumulated results for "e", "c" and "acc" are indicated by '.

4.3 Coverage

Stepwise coverage for each model is listed in Table 5, Table 6, and Table 7. As before, "f_{curr}" indicates the threshold. The brackets indicate language: "E" for English and "J" for Japanese. "found" is the number of content tokens matched with correct translation pairs. "ideal" is the upper bound of content tokens that should be found by the algorithm; it is the total number of content tokens in the translation units whose co-occurrence frequency is at least "f_{curr}" times in the original parallel corpora. "cover" lists coverage. The prefix "i_" is the fraction of found tokens over ideal tokens and the prefix "t_" is the fraction of found tokens over the total number of both content and functional tokens in the data. For 2000 test parallel sentences, there are 30255 tokens in the English half and 38827 tokens in the Japanese half. The gap between the number of "ideal" tokens and that of total tokens is due to filtering of functional words in the generation of translation units.

the effect is solely on the latter threshold (for the correlation score), and the former threshold (for the co-occurrence frequency) does not alter and remains 2.

f_{curr}	found(E)	ideal(E)	i_cover(E)	t_cover(E)	found(J)	ideal(J)	i_cover(J)	t_cover(J)
100.0	0	445	0	0	0	486	0	0
50.0	0	1182	0	0	0	1274	0	0
25.0	46	2562	**0.0179**	0.0015	46	2564	**0.0179**	0.0011
12.0	156	4275	**0.0364**	0.0051	146	4407	**0.0331**	0.0037
10.0	344	4743	**0.0725**	0.0113	334	4935	**0.0676**	0.0086
9.0	465	4952	**0.0939**	0.0153	455	5247	**0.0867**	0.0117
8.0	511	5242	**0.0974**	0.0168	501	5593	**0.0895**	0.0129
7.0	577	5590	**0.1032**	0.0190	567	5991	**0.0946**	0.0146
6.0	744	5944	**0.1251**	0.0245	734	6398	**0.1147**	0.0189
5.0	899	6350	**0.1415**	0.0297	891	6894	**0.1292**	0.0229
4.0	1193	6865	**0.1737**	0.0394	1195	7477	**0.1598**	0.0307
3.0	1547	7418	**0.2085**	0.0511	1549	8257	**0.1875**	0.0398
2.0	2594	8128	**0.3191**	0.0857	2617	9249	**0.2829**	0.0674
1.9	2686	8128	**0.3304**	0.0887	2713	9249	**0.2933**	0.0698
1.8	2831	8128	**0.3483**	0.0935	2858	9249	**0.3090**	0.0736
1.7	2952	8128	**0.3631**	0.0975	2983	9249	**0.3225**	0.0768
1.6	3180	8128	**0.3912**	0.1051	3214	9249	**0.3474**	0.0827
1.5	3387	8128	**0.4167**	0.1119	3423	9249	**0.3700**	0.0881
1.4	3587	8128	**0.4413**	0.1185	3628	9249	**0.3922**	0.0934
1.3	3836	8128	**0.4719**	0.1267	3901	9249	**0.4217**	0.1004
1.2	4106	8128	**0.5051**	0.1357	4184	9249	**0.4523**	0.1077
1.1	4470	8128	**0.5499**	0.1477	4558	9249	**0.4928**	0.1173

Table 5: Coverage(Bound-length N-gram)

f_{curr}	found(E)	ideal(E)	i_cover(E)	t_cover(E)	found(J)	ideal(J)	i_cover(J)	t_cover(J)
100.0	52	1374	**0.0378**	0.0017	52	1338	**0.0388**	0.0013
50.0	371	2813	**0.1318**	0.0122	372	2643	**0.1407**	0.0095
25.0	695	5019	**0.1384**	0.0229	696	4684	**0.1485**	0.0179
12.0	1251	7129	**0.1754**	0.0413	1246	6873	**0.1812**	0.0320
10.0	1478	7629	**0.1937**	0.0488	1463	7441	**0.1966**	0.0376
9.0	1607	7917	**0.2029**	0.0531	1590	7715	**0.2060**	0.0409
8.0	1690	8208	**0.2058**	0.0558	1673	8075	**0.2071**	0.0430
7.0	1893	8535	**0.2217**	0.0625	1879	8463	**0.2220**	0.0483
6.0	2023	8939	**0.2263**	0.0668	2015	8854	**0.2275**	0.0518
5.0	2464	9390	**0.2624**	0.0814	2445	9318	**0.2623**	0.0629
4.0	2893	9800	**0.2952**	0.0956	2882	9891	**0.2913**	0.0742
3.0	3425	10380	**0.3299**	0.1132	3439	10625	**0.3236**	0.0885
2.0	4702	11020	**0.4266**	0.1220	4737	11439	**0.4141**	0.1220
1.9	4869	11020	**0.4418**	0.1609	4906	11439	**0.4288**	0.1263
1.8	5020	11020	**0.4555**	0.1659	5057	11439	**0.4420**	0.1302
1.7	5177	11020	**0.4697**	0.1711	5214	11439	**0.4558**	0.1342
1.6	5388	11020	**0.4889**	0.1780	5423	11439	**0.4740**	0.1396
1.5	5621	11020	**0.5100**	0.1857	5676	11439	**0.4961**	0.1461
1.4	5907	11020	**0.5360**	0.1952	5971	11439	**0.5219**	0.1537
1.3	6227	11020	**0.5650**	0.2058	6298	11439	**0.5505**	0.1622
1.2	6513	11020	**0.5910**	0.2152	6589	11439	**0.5760**	0.1697
1.1	6787	11020	**0.6158**	0.2243	6874	11439	**0.6009**	0.1770

Table 6: Coverage(Chunk-bound N-gram)

f_{curr}	found(E)	ideal(E)	i_cover(E)	t_cover(E)	found(J)	ideal(J)	i_cover(J)	t_cover(J)
100.0	52	1370	**0.0379**	0.0017	52	1334	**0.0389**	0.0013
50.0	370	2806	**0.1318**	0.0122	371	2629	**0.1411**	0.0095
25.0	693	5010	**0.1383**	0.0229	694	4675	**0.1484**	0.0178
12.0	1238	7117	**0.1739**	0.0409	1233	6845	**0.1801**	0.0317
10.0	1429	7611	**0.1877**	0.0472	1424	7428	**0.1917**	0.0366
9.0	1583	7906	**0.2002**	0.0523	1576	7714	**0.2043**	0.0405
8.0	1689	8201	**0.2059**	0.0558	1682	8074	**0.2083**	0.0433
7.0	1945	8522	**0.2282**	0.0642	1925	8455	**0.2276**	0.0495
6.0	2083	8930	**0.2332**	0.0688	2064	8854	**0.2331**	0.0531
5.0	2481	9376	**0.2646**	0.0820	2458	9317	**0.2638**	0.0633
4.0	2918	9792	**0.2979**	0.0964	2901	9893	**0.2932**	0.0747
3.0	3473	10367	**0.3350**	0.1147	3490	10633	**0.3282**	0.0898
2.0	4736	11011	**0.4301**	0.1565	4769	11450	**0.4165**	0.1228
1.9	4893	11011	**0.4443**	0.1617	4926	11450	**0.4302**	0.1268
1.8	5032	11011	**0.4569**	0.1663	5063	11450	**0.4421**	0.1303
1.7	5155	11011	**0.4681**	0.1703	5192	11450	**0.4534**	0.1337
1.6	5369	11011	**0.4876**	0.1774	5398	11450	**0.4714**	0.1390
1.5	5630	11011	**0.5113**	0.1860	5672	11450	**0.4953**	0.1460
1.4	5908	11011	**0.5365**	0.1952	5963	11450	**0.5207**	0.1535
1.3	6205	11011	**0.5635**	0.2050	6275	11450	**0.5480**	0.1616
1.2	6415	11011	**0.5825**	0.2120	6487	11450	**0.5665**	0.1670
1.1	6657	11011	**0.6045**	0.2200	6744	11450	**0.5889**	0.1736

Table 7: Coverage(Dependency-linked N-gram)

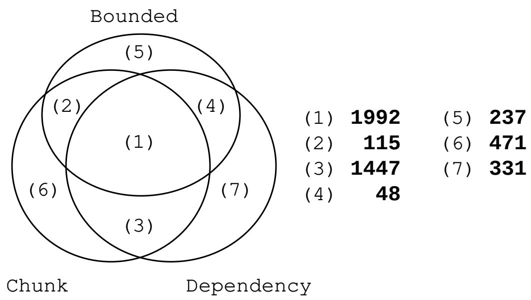

Figure 5: Venn diagram

model	English	Japanese
B	look forward	楽しみにする
B	look forward	期待-する
B	look forward	心-待ち
B	do not hesitate	遠慮-なく
C	Hong Kong	香港
C	San Diego	サンディエゴ
C	Parker	パーカー
D	free (of) charge	無料
D	point out	指摘
D	go press	印刷 (-に-) 回す
D	affect	影響 (-を-) 与える

Table 8: correct translation pairs

5 Discussion

Of the three models, Chunk-bound N-gram yields the best performance both in accuracy (83%) and in coverage (60%)[3]. Compared with the Bound-length N-gram, it achieves approximately 13% improvement in accuracy and 5-9% improvement in coverage at threshold 1.1.

Although Bound-length N-gram generates more translation units than Chunk-bound N-gram, it extracts fewer correct translation pairs (and results in low coverage). A possible explanation for this phenomenon is that Bound-length N-gram tends to generate too many unnecessary translation units which increase the noise for the extraction algorithm.

Dependency-linked N-gram follows a similar transition of accuracy and coverage as Chunk-bound N-gram. Figure 5 illustrates the Venn diagram of the number of correct translation pairs extracted in each model. As many as 3439 translation pairs from Dependency-linked N-gram and Chunk-bound N-gram are found in common. Based on these observation, we could say that dependency links do not contribute significantly. However, as dependency parsers are still prone to some errors, we will need further investigation with improved dependency parsers.

Table 8 lists the sample correct translation pairs that are unique to each model. Most translation pairs unique to Chunk-bound N-gram are

[3]We did not evaluate results when f_{curr} = 1.0, since it means threshold 0 , i.e. random pairing.

named entities (NP compounds) and one-to-one correspondence. This matches our expectation, as translation units in Chunk-bound N-gram are limited within chunk boundaries. The reason why the other two failed to obtain these translation pairs is probably due to a large number of overlapped translation units generated. Our extraction algorithm filters out the overlapped entries once the correct pairs are identified, and thus a large number of overlapped translation units sometimes become noise.

Bound-length N-gram and Dependency-linked N-gram include longer pairs, some of which are idiomatic expressions. Theoretically speaking, translation pairs like "look forward" should be extracted by Dependency-linked N-gram. A close examination of the data reveals that in some sentences, "look" and "forward" are not recognized as dependency-linked. These preprocessing failures can be overcome by further improvement of the tools used.

Based on the above analysis, we conclude that chunking boundaries are useful clues in building bilingual seed dictionary as Chunk-bound N-gram has demonstrated high precision and wide coverage. However, for parallel corpora that include a great deal of domain-specific or idiomatic expressions, partial use of dependency links is desirable.

There is still a remaining problem with our method. That is how to determine translation pairs which co-occur only once. One simple approach is to use a machine-readable bilingual dictionary. However, a more fundamental solution may lie in the partial structural matching of parallel sentences (Watanabe et al., 2000). We intend to incorporate these techniques to improve the overall coverage.

6 Conclusion

This paper reports on-going research on extracting bilingual lexicon from English-Japanese parallel corpora. Three models including a previously proposed one in (Kitamura and Matsumoto, 1996) are compared in this paper. Through preliminary experiments with 10000 bilingual sentences, we obtain that our new models (Chunk-bound N-gram and Dependency-linked N-gram) gain approximately 13% improvement in accuracy and 5-9% improvement in coverage from the baseline model (Bound-length N-gram). We present quantitative and qualitative analysis of the results in three models. We conclude that chunk boundaries are useful for building initial bilingual lexicon, and that idiomatic expressions may be partially handled with by dependency links.

References

M. Haruno, S. Ikehara, and T. Yamazaki. 1996. Learning bilingual collocations by word-level sorting. In *COLING-96: The 16th International Conference on Computational Linguistics*, pages 525–530.

M. Kitamura and Y. Matsumoto. 1996. Automatic extraction of word sequence correspondences in parallel corpora. In *Proc. 4th Workshop on Very Large Corpora*, pages 79–87.

J. Kupiec. 1993. An algorithm for finding noun phrase correspondences in bilingual corpora. In *ACL-93: 31st Annual Meeting of the Association for Computational Linguistics*, pages 23–30.

Y. Matsumoto and M. Asahara. 2001. Ipadic users manual. Technical report.

I.D. Melamed. 1995. Automatic evaluation and uniform filter cascades for inducing n-best translation lexicons. In *Proc. of 3rd Workshop on Very Large Corpora*, pages 184–198.

I.D. Melamed. 2000. Models of translational equivalence. In *Computational Linguistics*, volume 26(2), pages 221–249.

B. Santorini. 1991. Part-of-speech tagging guidelines for the penn treebank project. Technical report.

F. Smadja, K.R. McKeown, and V. Hatzuvassiloglou. 1996. Translating collocations for billingual lexicons: A statistical approach. In *Computational Linguistics*, volume 22(1), pages 1–38.

K. Takubo and M. Hashimoto. 1995. *A Dictionary of English Bussiness Letter Expressions*. Nihon Keizai Shimbun, Inc.

H. Watanabe, S. Kurohashi, and E. Aramaki. 2000. Finding structual correspondences from bilingual parsed corpus for corpus-based translation. In *COLING-2000: The 18th International Conference on Computational Linguistics*, pages 906–912.

K. Yamamoto and Y. Matsumoto. 2000. Acquisition of phrase-level bilingual correspondence using dependency structure. In *COLING-2000: The 18th International Conference on Computational Linguistics*, pages 933–939.