Syntactic Reordering in Preprocessing for Japanese→English Translation: MIT System Description for NTCIR-7 Patent Translation Task

Jason Katz-Brown
Google Japan, Inc.*
Shibuya, Tokyo 150-8512
jasonkb@google.com

Michael Collins
MIT CSAIL
Stata Center, Cambridge, MA 02139
mcollins@csail.mit.edu

Abstract

We experimented with a well-known technique of training a Japanese→English translation system on a Japanese training corpus that has been reordered into an English-like word order. We achieved surprisingly impressive results by naively reordering each Japanese sentence into reverse order. We also developed a reordering algorithm that transforms a Japanese dependency parse into English word order.
Keywords: *Long-Distance Reordering, Preordering.*

1 Introduction

We experimented with reordering the Japanese training data into an English-like word order before running Moses training (following [11]). When translating an unseen Japanese sentence to English, we first **preorder** it into this English-like word order, then translate preordered Japanese sentence with the specially-trained Moses setup. With this approach, the burden of reordering phrases is pushed to a syntactic preprocessing step, and the Moses translator itself can perform a largely **monotonic** (no reordering) translation, at which it excels.

The challenge is to build an algorithm that reorders a Japanese sentence into a pseudo-Japanese sentence that has the same words but in English-like word order. In this paper I describe two such algorithms. The first is fast and naive, and simply reverses the order of all tokens after splitting the sentence at punctuation and 'は', the topic marker. The second algorithm uses three linguistically-motivated heuristics for flattening a tree formed from a dependency parse.

In our experiments, we found an improvement in translation quality using the naive reverse preprocessor. Surprisingly, we saw a *smaller* improvement using the linguistically-motived smarter preprocessor, which usually produced more accurately English-like pseudo-Japanese.

*Research conducted while a graduate student in MIT CSAIL.

2 Reverse preordering

English is head-initial. Japanese is head-final. So reversing the word order of a Japanese sentence could be a good start towards an English-like order. We factor out the commonality that the topic of English and Japanese sentences both come at the beginning by reversing words before and after the topic marker 'は' separately. Punctuation is kept in the same place.

We begin by tokenizing the sentence with the Mecab [7] morphological analyser, then follow these steps:

1. Split the Japanese sentence at punctuation into a list of "segments".

2. Further split each segment at 'は', the topic marker, to get a pre-topic segment (which ends with 'は') and post-topic segment. The motivation is that the topic comes at the beginning of both Japanese and English sentences, and should not move to the end.

3. Reverse the order of the words in each segment, so each segment reads backwards.

4. Concatenate the segments and punctuation back together in their original order in the sentence.

We call this reordering the **REV preorder**. Let us follow these steps to reorder the example shown in Gloss 1, which has words separated by spaces and segment boundaries marked by ‖. The topic segment is 'プリアンプ 3 は', which is reversed into 'は 3 プリアンプ'. The middle segment is also reversed, and these two segments are concatenated together with the final period to get Gloss 2, the final REV preorder. [htp] This REV preordering could be successfully translated into English monotonically by adding only a few auxiliary words: "The 3 preamp outputs to

(1) プリアンプ 3 は ‖ 入力 さ れ た 再生 信号 を 増幅 し て Ａ ＧＣ アンプ 4 へ 出力 する ‖ 。
Preamp 3-*TOP* ‖ input-*Passive* repr. signal-*Acc* amplify and AGC amp 4-to output ‖ .
"The preamp 3 amplifies an input reproduction signal, and sends out to an AGC amplifier 4."

(2) は 3 プリアンプ する 出力 へ 4 アンプ ＡＧＣ て し 増幅 を 信号 再生 たれさ 入力 。
TOP-3 preamp output to 4 amp AGC and amplify *Acc*-repr. signal *Passive*-input .

4 amp AGC and amplifies the reproduction signal that has been input."

We can analyze this reverse ordering as performing both local and long-distance movement. Long-distance movement can be seen in the verb '出力 する' (output) moving from the end of the sentence to the beginning of the sentence. This long-distance reversal is effective in transforming head-final verb and noun phrases to be head-initial as they are in English. Local movement can be seen in the verb '出力 さ れ た' (whose tokens are literally, output do [passive] [past tense]) reordering to 'た れ さ 入力' ([past tense] [passive] do output). This local reordering is effective for verbs because most English auxiliaries precede the verb they assist, while Japanese auxiliaries and inflections follow the verb their verb.

This naive REV does have two significant problems. First, subjects marked by 'が', the Japanese subject marker, are reordered to follow their verb. We could have chosen to also split segments at 'が', but this would break the word order if the sentence contained a relative clause with 'が' in it. The second problem is that compound nouns are reversed, and English and Japanese compounds already have the same structure. In reversed Gloss 2, '再生 信号' (reproduction signal) has been reordered into '信号 再生' (signal reproduction), which is clearly a worse order than the original.

3 Dependency tree preordering

In this section we present a more sophisticated way to reorder Japanese into English by flattening a dependency tree parse of the Japanese. We start by running the sentence through Mecab, which tokenizes and tags each word with part of speech. We split the sentence into segments at punctuation marks, apply our reordering technique to each segment separately, and in the end concatenate the reordered segments and punctuation (in the same order they appeared in the original sentence) together. We call this reordering the **CABOCHA preorder**.

To reorder a segment, we first parse it with the Cabocha Japanese Dependency Structure Analyzer [8]. The output of Cabocha is a list of **chunks**. A chunk is roughly a content word (usually the head) and affixed function words like case markers or verbal morphology. Each chunk contains the following information:

- ID number
- Start and end position in sentence
- Chunk that this chunk modifies (in other words, parent chunk)
- Position of head

From this list of chunks, we can construct a dependency tree with a node for each chunk and an edge for each dependency. Because of how Cabocha constrains its dependency model, all of a node's children precede it in the sentence. As a result, the root node is always the final chunk of the sentence. Figure 1 shows the dependency tree constructed from the preamp example (once the period at the end has been split away), with each chunk's head underlined and its part of speech listed.

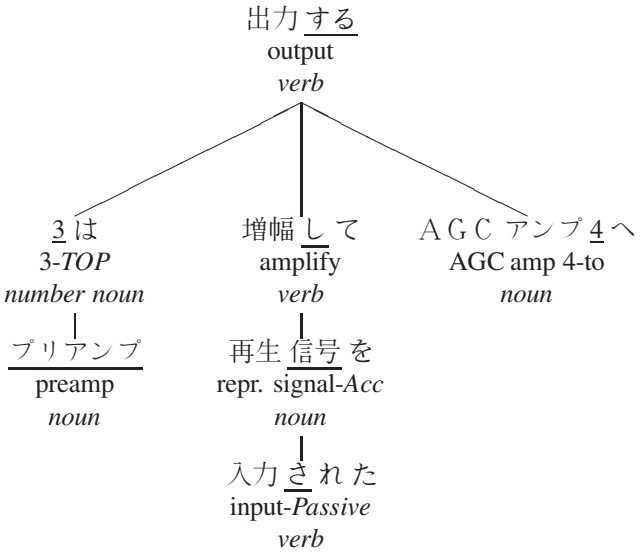

Figure 1. Dependency tree for preamp example.

We reorder a Japanese segment in two steps:

1. Flatten the dependency tree according to four rules.
2. Reverse the word order within each chunk.

To flatten the tree we decide for each node into which position among its children to flatten. The

crux of the algorithm is determining where *chunk* should be placed among its children. All non-verbs are placed before their children, which induces a head-initial word order. The placement of verbs is determined by going down the following list:

1. Immediately after rightmost topic or subject, if it exists.
2. Otherwise, immediately before leftmost object, if it exists.
3. Otherwise, immediately after rightmost verb, if it exists. This is to prevent verbs from leapfrogging verbs that preceded them that share only a coordinative dependency.
4. Otherwise, before all children.

The CABOCHA preorder for our preamp example is shown in Gloss 3. As with the REV preorder, we can add auxiliaries to the gloss of the CABOCHA preorder to form a correct translation: "The 3 preamp outputs the amplified reproduction signals that has been input to 4 amp AGC." The placement of the main verb "output" is questionable; it should probably come after "amplify", with which it coordinates, but our rules put it immediately after its subject, "preamp 3". One fix would be to never place verbs farther left than their leftmost child verb. The verb "amplify" has been placed correctly before its object "reproduction signal". The head-final noun phrase '入力 さ れ た 再生 信号' (input-*Passive* reproduction signal) successfully reordered to be head-initial '信号 再生 た れ さ 入力' (signal reproduction *Passive* input).

Thanks to its systematic head-final to head-initial inversion, we found that the CABOCHA preorder tended to closely match English word order. We demonstrate in Section 4.1 that CABOCHA dominates REV and BASELINE (no reordering) preorders in translation quality when translating monotonically (that is, not allowing reordering other than what has already been reordered in the preorder). We will now take a look at examples of CABOCHA and REV preorders and what it looks like to translate them monotonically.

4 Experiments

We trained our Japanese→English Moses system on only the 53.5 million word Japanese-English Patent Parallel Corpus [10] training set provided for the Patent Translation Task [4]. We trained a 5-gram language model and recasing model on only the English side of this corpus. We use the 915-sentence development (`dev`) and 899-sentence test (`test`) sets, both single-reference, supplied for the Dry Run of the Task. Our training and tuning parameters were the same as the organizer's baseline, except as described below.

4.1 Evaluating preorder efficacy

We experimented with quality of the REV and CABOCHA preorders at various settings of the decoder's maximum distortion distance (which we denote $DistortionLimit$ and beam width. Table 2 shows BLEU scores on the test set supplied for the Dry Run of the Patent Translation Task. Our primary submission in the Formal Run of the Task corresponds to setting $DistortionLimit = 9$ and using REV preorder. Our secondary submission corresponds to setting $DistortionLimit = 9$ and using CABOCHA preorder. Table 1 shows the official results of our systems in the formal run of the Patent Translation Task [4], compared to `Moses`, which was the organizer baseline, and `tsbmt`, a rule-based system that scored highest on average human evaluation.

The systems used to generate the results in Table 2 were trained on other parallel corpora (JENAAD and edict) and used a non-patent recaser. Therefore the scores are not directly comparable to our submitted systems or other systems. Particularly because of the unsuitable recaser, the scores are relatively low. We realized right before the formal run submission deadline that we must submit a system trained only on the patent data. This regretfully caused our late submission.

When translating preorder REV, which has a roughly English word order, quality peaks at about $DistortionLimit = 9$, and drops off for higher values. In contrast, when translating the BASELINE (no reordering) preorder, the higher the setting of $DistortionLimit$, the higher the translation quality. We can interpret this result as follows: Translating between REV and English, most words need to move fewer than 6 places, so allowing them to move farther results in incorrect reordering; translating between BASELINE and English, some words need to move farther than 9 places, so disallowing such long movement rules out many correct translations.

Table 2 also illustrates the impact of preordering algorithm on translation quality. When no reordering is allowed during decoding, CABOCHA achieves the highest BLEU score, validating our observation that its word order is closest to English. However, with a limited amount of reordering, REV is the leader. This is a very surprizing result, but one that was consistent across test corpora or feature function choice.

Equally surprizing is that when unlimited reordering is allowed, the BASELINE preorder, which is the original Japanese word order, performs best. This is shocking, and we can offer no explanation. With unlimited reordering and employing the default Moses feature functions, only the language model can evaluate long-distance reorderings. Because language model scores are in no way conditioned on the source sentence, the language model cannot advise the de-

(3) は 3 プリアンプ する 出力 て し 増幅 を 信号 再生 たれ さ 入力 へ 4 アンプ ＡＧＣ 。
TOP-3 preamp output and amplify *Acc*-repr. signal *Passive*-input to 4 amp AGC .

Formal Run ID	Preorder	BLEU	Adequacy	Fluency	Seconds per sentence
MIT (1)	REV	27.14	3.15	3.66	8.4
MIT (2)	CABOCHA	26.44	(N/A)	(N/A)	8.0
Moses	BASELINE	27.14	2.81	3.55	18.2
tsbmt	(N/A)	26.44	3.81	3.94	0.23

Table 1. Human and BLEU scores on the formal run.

coder on how to reorder words.

The decoder is "driving blind" when positioning words far away from their original spot, but has maximum freedom to assemble them according to the language model into fluent English. This freedom may be the main contributor to the high BLEU score of the unlimited-reordering system. Still, we would expect one of the preordered systems to outperform the baseline. It may be the case that the phrase table of the baseline system is unexpectedly of higher quality than that of the preordered systems, or that the local inversion in the preordered systems degrades BLEU score with unlimited reordering.

4.2 Manual Evaluation

In the Formal Run Manual Evaluation, our primary (REV preorder) system achieved the highest "adequacy" score of any statistical translation system, 314 versus next-best 296. Our "fluency" score was similar to other top statistical systems. In the Automatic Evaluation, our BLEU score was also very similar to the top statistical systems. BLEU score has limited ability to evaluate differences in word order [1], which may explain why our preordered system did relatively better under subjective metrics than BLEU score.

4.3 Beam size

We set the decoder beam size (Moses flag `test-stack`) to 100, half of the organizer baseline beam size setting of 200. According to Table 3, setting beam size 100 is about twice as fast with a small loss in quality compared to beam size 200.

Stack size	BLEU	Seconds per sentence
100	28.46	4.5
200	28.63	8.4
400	28.51	16.1

Table 3. How stack size affects BLEU score and translation time.

4.4 Long-distance reordering features

While experimenting with preordering techniques, we also developed in Moses long-distance reordering feature functions based on a dependency parse of the source sentence. However, we did not finish these features in time for our the Formul Run deadline and are not included in our submissions.

Our method and results are described in detail in [6]. As an example of our method, the dependency parse identifies the input sentence's main verb and object. During translation, we give higher scores to translation hypotheses that put the main verb before its object. Let's look at how this works for Gloss 1. Figure 1 showed its dependency parse. For instance, "amplify" is modified by its child "reproduction signal-*Acc*". Further observing that "reproduction signal-*Acc*" has accusative case, and knowing that the target language English has Subject–Verb–Object order, the translator prefers to translate the verb "amplify" before it translates its object "reproduction-signal". We codified this preference by introducing a feature function in Moses that counts occurrences of a verb being translated before its object.

In addition, we introduced feature functions for a range of grammatical constructs: a feature that counts when relative clauses are translated after the noun they modify, one that counts when genitive modifiers are translated after the noun they modify, and so on. We could have a feature for every part-of-speech and case pairwise combination. Furthermore, we introduced a cohesion constraint in the same vein as [2]

We discriminatively trained the weights of these features to identify the most useful features and maximize translation quality. This discriminative training step is important to tune the system for the grammatical features of the target language. While the verb-before-its-object feature function identifies good English translations, if we were translating into Japanese, we would give a negative weight to the verb-before-its-object feature. This setup would correctly prefer to translate Japanese verbs after their objects.

We achieved the best translation quality when combining approaches and used the reverse preprocessor and an assortment of dependency-motivated feature

$DistortionLimit$	BASELINE	REV	CABOCHA	Seconds per sentence[a]
0	20.86	20.32	21.61	2.2
6	23.76	25.44	24.79	5.0
9	25.24	25.49	25.12	7.8
unlimited	26.07	25.08	24.58	37.2

[a]Time taken on the BASELINE preorder; times for preordered systems tended to be shorter.

Table 2. How $DistortionLimit$ affects BLEU score and translation time for different preorders.

functions at optimal weights. Altogether, when limiting ourselves to $DistortionLimit = 9$, we achieved a BLEU score improvement of 27.96→28.74.

5 Related Work

Collins et al. [3] introduced the technique of preordering for building a phrase-based system with long-distance reordering ability. Working on German→English, they wrote rules to transform a deep parse of the German sentence so that its words read in English word order. They parse the German training data, apply these rules to transform it into English word order in a preprocessing step, then train a phrase-based system on the reordered data. Before translation, they perform the same reordering on the input sentence. This led to a significant improvement in English output word order. Wang et al. [11] followed up with analogous experiments for Chinese→English.

Kanthak et al. [5] further developed the preordering technique. Their system automatically learns how to reorder source sentences into target language word order from monotonization of training data word alignments. However the weakness of their baseline decoder, which failed to translate 37% of their Japanese test corpus, makes it difficult to tell how effective their automatically-trained source-side reorderer is.

Li et al. [9] take the idea of Kanthak et al. one step further. First they trained a statistical source-side reordering model, which predicts whether a node of a tree should keep its children in order or invert them, by using word alignments and deep parses of the source sentences of the training data. To translate a sentence, they generate the 10 best preorders with their reordering model, then translates all of the preorders with a phrase-based decoder (using a maximum distortion limit of 4) and out of the 10 pick the translation with highest combined source-side reordering model score and decoder score. They worked with Chinese→English and achieved an improvement over their no-preordering baseline of the same magnitude as Wang et al. [11]. The advantage of Li et al.'s work is that there is no need for handwrit tree reordering rules.

6 Conclusion

We presented algorithms for reordering Japanese into an English word order before translation, with the surprising result that a naive preprocessor that basically flips the Japanese to read backwards outperforms a dependency-tree flattening method we developed. Our experiments and the NTCIR subjective evaluation showed that reordering during preprocessing improves translation quality and achieves good results at efficient decoder settings.

References

[1] C. Callison-Burch, M. Osborne, and P. Koehn. Re-evaluating the role of BLEU in machine translation research. In *Proceedings of the 11th Conference of the European Chapter of the Association of Computational Linguistics*, pages 249–256, 2006.

[2] C. Cherry. Cohesive phrase-based decoding for statistical machine translation. In *Proceedings of ACL-08: HLT*, pages 72–80, Columbus, Ohio, June 2008. Association for Computational Linguistics.

[3] M. Collins, P. Koehn, and I. Kucerova. Clause restructuring for statistical machine translation. In *ACL*. The Association for Computer Linguistics, 2005.

[4] A. Fujii, T. Utsuro, M. Yamamoto, and M. Utiyama. Overview of the Patent Translation Task at the NTCIR-7 workshop. In *Proceedings of the 7th NTCIR Workshop Meeting on Evaluation of Information Access Technologies: Information Retrieval, Question Answering and Cross-lingual Information Access*, 2008.

[5] S. Kanthak, D. Vilar, E. Matusov, R. Zens, and H. Ney. Novel reordering approaches in phrase-based statistical machine translation. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*, pages 167–174, 2005.

[6] J. Katz-Brown. Dependency reordering features for japanese-english phrase-based translation. Master's thesis, Massachusetts Institute of Technology, August 2008.

[7] T. Kudo. Mecab: Yet another part-of-speech and morphological analyzer, 2007.

[8] T. Kudo and Y. Matsumoto. Japanese dependency analysis using cascaded chunking. In *CoNLL 2002: Proceedings of the 6th Conference on Natural Language Learning 2002 (COLING 2002 Post-Conference Workshops)*, pages 63–69, 2002.

[9] C.-H. Li, M. Li, D. Zhang, M. Li, M. Zhou, and Y. Guan. A probabilistic approach to syntax-based reordering for statistical machine translation. In *ACL*. The Association for Computer Linguistics, 2007.

[10] M. Utiyama, M. Yamamoto, A. Fujii, and T. Utsuro. Description of patent parallel corpus for NTCIR-7 patent translation task. `http://if-lab.slis.tsukuba.ac.jp/fujii/ntc7patmt/ppc.pdf`, 2007.

[11] C. Wang, M. Collins, and P. Koehn. Chinese syntactic reordering for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 737–745, 2007.