The POSTECH's Statistical Machine Translation Systems for the NTCIR-8 Patent Translation Task

Hwidong Na
Pohang University of Science and Technology (POSTECH),
leona@postech.ac.kr

Hankyoung Kim
Pohang University of Science and Technology (POSTECH),
arch@postech.ac.kr

Jong-Hyeok Lee
Pohang University of Science and Technology (POSTECH),
jhlee@postech.ac.kr

ABSTRACT

This paper proposes the POSTECH's statistical machine translation (SMT) systems for the NTCIR-8 Patent Translation Task. We entered the translation subtasks and submitted a formal run for a Japanese-to-English (KLE-je) and a English-to-Japanese (KLE-ej) translation. The baseline system is derived from a common phrase-based SMT framework. For KLE-je, we adopted a cluster-based model using syntactic information as well as the word similarity of Japanese sentences. For KLE-ej, we adopted a reordering method to improve the fluency of translation result. We did not submit a formal run for the Patent Mining Task.

Categories and Subject Descriptors

I.2.7 [**Computing Methodologies**]: ARTIFICIAL INTELLIGENCE—*Natural Language Processing*

General Terms

Experimentation, Languages

Keywords

phrase-based SMT, reordering, cluster-based SMT

1. INTRODUCTION

It is well known that domain specific models perform well in natural language processing tasks[3, 16, 4, 13, 17, 5]. In statistical machine translation (SMT), topic dependent translation methods are based on syntactic information or genres. However, there has not been any previous study done to date on adapting domain specific models using syntactic information and genres simultaneously.

In this paper, we present an integrated method using domain specific models in SMT. We assume syntactic structure similarity to group sentence types, and word similarity information to classify the training corpus on the topic. We extract these models from clusters obtained by computing the syntactic structure and word similarity. This is done with a machine learning algorithm using the syntactic structure and word similarity as distance functions that cluster the training corpus to achieve the models. The domain specific language and translation models are then used for a formal run in Japanese-to-English translation (KLE-je).

When anlyzing the quality of translation, people usually consider the following two aspects: the first is adequacy, the second is fluency. Adequacy is strongly influenced by lexical transfer, i.e. the correct choice of target words for given source words. Fluency is affected by the reordering of sentence parts to ensure grammarical consistency in translation, i.e. getting the target words in the right position. A high quality of translation should satisfy both the adequacy and fluency.

Phrase-based SMT (PBSMT) does not sufficiently satisfy both aspects. PBSMT is good at lexical transfer with local reordering for a phrase pair. When the translation deals with a multiple choice of target words for given source words, the best choice is the highest probablity. We expect the best choice is to be an accurate translation. However, PBSMT is poor at global reordering because each phrase moves to another position within a predefined window size. When the source sentence has a very different structure from the target one, the correct position of a phrase is often far beyond the limitation of movement. Therefore, we cannot expect a fluent translation from PBSMT alone. We adopt a reordering method to improve the fluency of translation [12] in English-to-Japanese translation (KLE-ej).

2. CLUSTER-BASED SMT

Experiments with language and translation models have shown that adopting domain specific models is a possible solution to improve translation quality. Some previous research classified a sentence type, such as interrogatives, imperatives and enumerations, with syntactic information [8, 16, 4]. However, their method uses corpus dependent heuristics which cannot be generalized [8] and fails to encode syntactic structure into the domain classification [16] or requires linguistic analysis on the target corpus [4]. Alternatively, other researches classified training corpus with their topics [13, 17, 5]. They made some improvements to previous research results using the same IPC as the input [13] and tried several measures such as tf/idf, LSA, perplexity and EM algorithm for model adaption [5]. Unforunately, these topic dependent translation methods have a weak point because they lack syntactic structural information.

In this paper, we present an integrated method for the domain classification problem which considers syntactic structure and word similarity. In the clustering methods for SMT, there are two kinds of approach which improve the model quality by adapting domain specific models using general models. One is the syntactic structural approach. This approach acquires domain specific models using clustering by the syntactic characteristics such as sentence type. It gives better language models for each domain because n-gram word order is influenced by the syntactic structure of a sentence. The other approach is clustering by topic or genre, which improves the translation models by adjusting the translation probabilities to fit into their topic. Language model quality can also be improved since the literary style of a genre affects the n-gram word order.

Because these two approaches each have their own advantages, we suggest a method that employs both of their advantages. We cluster the training corpus using the syntactic structure and word similarity. Based on our assumptions, similarity of syntactic structure indicates n-gram word order similarity, and word similarity between sentences means that their topics are close to each other. Accordingly, our method obtain clusters that have consistency in syntactic structure and topic. Therefore, domain specific models based on obtained clusters have both the virtues of syntactic structure and genre approach. Moreover, this method utilizes a machine learning algorithm without human supervision. Our algorithm requires no other linguistic knowledge on the corpus except for the dependency tree of the source language side. Because this algorithm automatically groups training corpus with syntactic structure similarity, it is less sensitive to corpus characteristics.

After clustering, we build PBSMT system acquired domain specific models based on clusters and then interpolate each model with the general model. This system adapts the models by linear interpolation. In our experimental results for Japanese-English, adopting a language and translation model works well. Adopting both models simultaneously shows the best result and we only submitted the best result for the formal run.

2.1 Scoring similarity of syntatic structures

To get the similarity of syntactic structure between two sentences, we use the source language dependency tree obtained by CaboCha[14]. The tree structure uses a *bunsetsu* as a unit, which is the semantic unit of Japanese that consists of morphemes. Some morphemes in a bunsetsu are more important than others to indicate the syntactic role of it, i.e. functional words. We call them *SYN* of bunsetsu and CaboCha identifies SYN by default. In the dependency tree structure, each bunsetsu has SYN information and every morpheme has its part of speeches (POS) tag. We define *POS of a bunsetsu* as follows.

POS of busetsu = POS of the first morpheme of SYN + the rest morpheme of SYN

In this paper we used a convolution kernel for computing the dependency tree similarity[3]. This measures the similarity between two dependency trees by counting the number of common subtrees. With this kernel function and information from the dependency tree, we calculate the syntactic similarity between two sentences. We make some modifications to match the kernel function to our experiment, i.e. the POS of bunsetsu that we already defined. Then, our modified version of the kernel function to calculate the score of syntactic structure similarity of two dependency tree can be described as follows.

Let a dependency tree of a sentence S_i be d_i, and N_i be the set of nodes that has elements as n_{ij} in tree d_i. Then, we define the set of common dependencies as $sim(n_1, n_2)$ where $n_1 \in N_1$ and $n_2 \in N_2$, satisfying the following conditions:

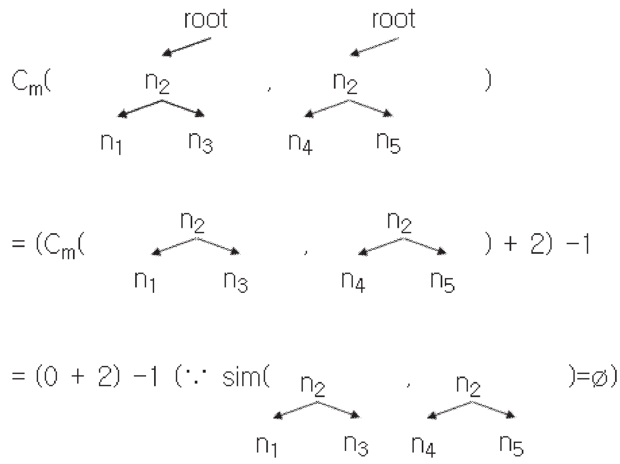

Figure 1: An example of C_m at exceptional case.

$$sim(n_1, n_2) = \left\{ (x, y) \middle| \begin{array}{c} x \in children(n_1) \\ y \in children(n_2) \\ POS(x) = POS(y) \end{array} \right\}$$

$C_m(n_1, n_2)$ is the number of common subgraphs rooted at n_1 and n_2. The original paper [3] defined the function C_m as follows:

If word(n_1) = word(n_1)
or children$(n_1) = \emptyset$ or children$(n_2) = \emptyset$
Then $C_m(n_1, n_2) = 0$
Else $C_m(n_1, n_2) = \prod_{(x,y) \in sim(n_1, n_2}(C_m + 2) - 1$

But in this formula, C_m yields a wrong value of -1 for a case where $sim(n_1, n_2)$ is an empty set (see example in Figure 1). The value of C_m should not be negative because C_m is a function that counts the number of common subgraph. So we redefine C_m as follows:

If POS(n_1) = POS(n_1) or sim$(n_1, n_2) = \emptyset$
or children$(n_1) = \emptyset$ or children$(n_2) = \emptyset$
Then $C_m(n_1, n_2) = 0$
Else $C_m(n_1, n_2) = \prod_{(x,y) \in sim(n_1, n_2}(C_m + 2) - 1$

Then, the modified kernel function between the two trees can be calculated as in [3].

$$K_(d_1, d_2) = \sum_{n_1 \in N_1, n_2 \in N_2} C_m(n_1, n_2)$$

The kernel value of two trees K is the number of common dependent of nodes. By normalizing K with the number of nodes at the two trees, we have the syntactic structure similarity $S(d_1, d_2)$ of the dependency trees d_1, d_2 of the two sentences.

$$S(d_1, d_2) = \frac{K(d_1, d_2)}{|N_1| + |N_2|} \quad (1)$$

2.2 Scoring word similarity

To calculate the word similarity score, we used a well-known measure, called the cosine similarity. It is used to calculate the similarity between vectors. Considering a sentence S_i as a bag-of-words w_{ij}, S_i is represented as vector V_i using their term frequency in the training corpus. Then, the cosine similarity between two vectors V_1 and V_2 of the two sentences is:

$$W(V_1, V_2) = \frac{V_1^T V_2}{|V_1||V_2|} \tag{2}$$

It ranges from 0 to 1, since the term frequencies cannot be negative.

2.3 Clustering algorigthm

In our clustering algorithm, we use (1) and (2) to get the syntactic structure and word similarity between the two sentences. We define a distance function $D(S_1, S_2)$ that deals with both similarities between the two sentences simultaneously.

$$D(S_1, S_2) = \alpha D(d_1, d_2) + (1 - \alpha) W(V_1, V_2)$$

2.3.1 *Default clustering algorithm*

The default clustering algorithm takes source language sentences as its input and makes groups from them. This algorithm runs according to the following process.

Step 1. Calculate the syntactic structure and word similarity of every sentence pair (S_i, S_j) in the training corpus according to the distance function $D(S_i, S_j)$.

Step 2. Randomly assign sentences in the training corpus to clusters. The number of clusters K is given by the user.

Step 3. For each sentence S_i in the training corpus, get the distances to clusters by calculating the average distance. An average distance $AVGD(S_i, C_k)$ from a sentence to a cluster C_k is:

$$AVGD(S_i, C_k) = \frac{\sum_{S_j \in C_k} D(S_i, S_i)}{|C|}$$

Step 4. For each sentence in the training corpus, assign it to the closest cluster. The distance function of this algorithm is a measure of similarity, so a larger distance function value means a closer relation. In other words, find the most similar cluster and assign the target sentence to the cluster.

$$C^* = \arg\max_{C_k} AVGD(S_i, C_k)$$

Step 5. Repeat the steps 3 and 4 until the number of reassigned sentences is smaller than the given thresholds.

Based on this algorithm, we divide the training corpus into clusters without any supervision. It takes the numbers of cluster and the interpolation ratio between the syntactic structure and word similarity.

2.3.2 *Classifier*

After the clustering algorithm is finished, we assign a sentence to the closest cluster by calculating the average distances to the clustered the training corpus. This classifier takes a linear time to assign the sentences.

Step 1. Similar to Step 1 of the clustering algorithm, calculate the distances between the sentence pairs in the target sentence and the training corpus.

Step 2. Calculate the average distance from the target sentence to each clusters.

Step 3. Select the most similar cluster according to the average distance and assign the target sentence to that cluster.

To decode the source language sentence, we must find the proper domain. It can be considered as a domain prediction problem from the input sentence. We used the proposed classifier not only to solve the problem in the decoding process but also to boost the calculation speed and minimize the storage space. The advanced algorithm is described in next section.

2.3.3 *Advanced clustering algorithm*

If there are an sufficient amount of sentences in each cluster during the clustering process, additional sentences may not be benificial to the clustering performance. If this is true, clustering accuracy depends more on the number of iterations than the size of the cluster. For this reason, we make the clustering algorithm efficient in terms of time and space. If there are plenty of sentences in the training corpus, select some portion of the corpus as the seeds and run the default clustering algorithm with them. Others are then clustered by the classifier to gather the cluster. With this algorithm, the iterative part (Step 3,4) is a quadratic, and the others have a linear complexity in time and space. The advanced algorithm runs according to the following process.

Step 1. Select a sufficient portion of the training corpus as the seeds and run the default clustering algorithm with them.

Step 2. Calculate the distances between the sentence pairs between the in-seed to the out-of-seed.

Step 3. After the default clustering algorithm is finished, assign the non-clustered sentences in the training corpus. For this step, we used the classifier mentioned above with the distance matrix calculated in step 2.

Step 4. Merge the clustered portion of the training corpus with the assigned sentences.

This algorithm reduces the quadratic complexity $O(x^2) + O(x(n - x)) = O(n)$ where x is the number of sentences in the seed corpus and $x \ll n$.

2.4 Adapting domain specific models

After the clustering steps are finished, we move on to the decoding steps. To translate an input sentence, we build models for each cluster and adapt the domain specific models interploated with general model. Linear interpolation is applied to both the language and translation models.

$$\begin{aligned} P(e) &= \lambda_{cl} P(e)_{cluster} + (1 - \lambda_{cl}) P(e)_{general} \\ P(f|e) &= \lambda_{ct} P(f|e)_{cluster} + (1 - \lambda_{ct}) P(f|e)_{general} \end{aligned}$$

We manually tune the rates λ_{cl} and λ_{ct} between the domain specific models and the general model. The overall architecture of the decoding system is shown in Figure 2.

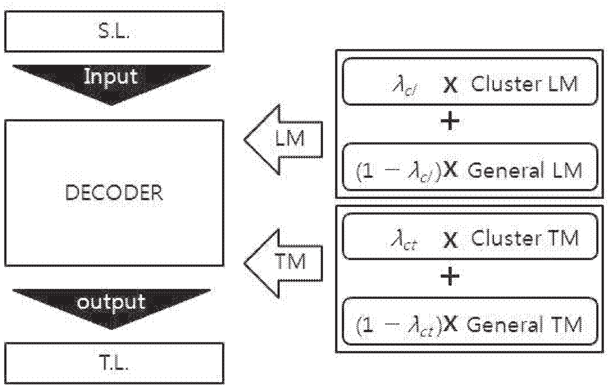

Figure 2: Genenral architecture of cluster-based SMT system.

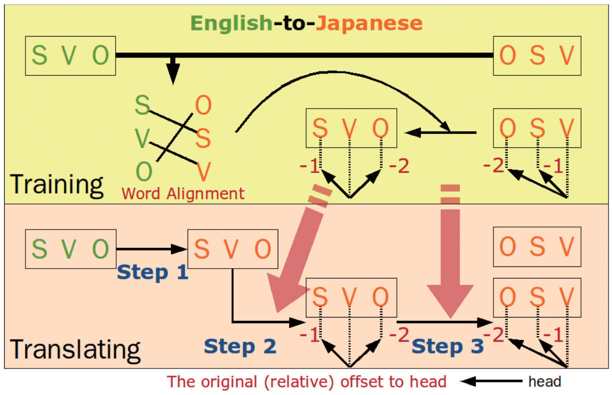

Figure 3: General Architecture of reordering method for improving fluency

3. REORDERING METHOD TO IMPROVE FLUENCY

If PBSMT gives the correct lexical choice, then a global reordering method improves the fluency and eventually the overall quality of the translation. Preprocessing as global reordering has been studied previously. The research in [15] investigated reordering between subject-verb-object (SVO) and SOV languages. They predefined "precedence reordering" based on a dependency parse tree, thereby specifying the order of dependents for a given head. The use of features for representing global reordering within a PBSMT model has also been studied [2, 7]. They modeled the "orientation" of a phrase as global reordering.

However, different sentence pairs requires different global reordering strateges. Because Japanese has a relatively free order except for the main predicate (verb), an English sentence with an SVO structure translated into a Japanese sentence can have either an SOV or an OSV structure. A Japanese sentence can be translated into a English sentence with an active or passive voice, containing different position for the predicate. In this case, the above two reordering approaches, i.e. reordering the source sentences without considering the corresponding sentences in the target language, fail to capture the sentence-specific difference.

[12] has considered a global reordering method at the sentence-level using word alignment. They reordered the target sentences according to word alignment in the training corpus, and recovered the order of reordered the target sentences which are the monotone translation results of the source sentences in the test corpus. We adapt their method to submit a formal run in English-to-Japanese translation. The general architecture of the method is shown in Figure 3.

3.1 Training phase

If we know the correct word alignments of a sentence pair, they guide the global reordering at the sentence-level. We assume the intersection of bidirectional word alignment is correct. Let a sentence S consist of a sequence of unit u_i and every unit has its alignment a_i. Using a bunsetsu as an unit u_i, each target sentence in the training corpus is reordered such that follows the order of the source sentence. In other words, a reordered sentence S' is the sorted sequence of u_i by a_i. Then we train PBSMT using the corpus with the reordered target sentences.

After we reorder the target sentence, we train a non-projective dependency parser as well as the PBSMT. The maximum spanning tree (MST) parser obtains the dependency of a sentence by finding the MST of the graph[11]. We regards each bunsetsu as a node in a directed graph. Dependency parse trees from the original target language parser replaces the role of human annotation in this paper. [1]

The original parse tree and the word alignments are used to train local tree order model (LTOM), suggested in [1]. For a given head word, we identify the permutation of dependents. Then we count the number of the permutations in the whole corpus in terms of the relative frequency.

3.2 Translating phase

In the translating phase, the improved fluency of the translations result in English-to-Japanese translation is performed as follows.

Step 1. Translate a source sentence using PBSMT:
As we assume that PBSMT gives an adequate translation, we use a PBSMT system for the lexical transfer. We do not use the distance-based reordering method (distortion limit = 0).

Step 2. Build the structure (parsing) of the translation:
We parse the translation result from PBSMT. The translated sentence is locally reordered within a phrase, but globally follows the order of the source language. Then the MST parser builds the dependency of the reordered sentence.

Step 3. Reconstruct the structure:
We reconstruct the parse tree in order to follow the order of the target language. For a given head word, the permutation of dependents with the highest frequency is used to fix each offset of the dependent. Finally, reading off the parse tree according to the offsets, we have a fluent translation result.

More details are included in [12]. [2]

[1]If we have a human annotated corpus with parallel translation, more accurate training is possible.
[2]The overall running scripts will be available on our website soon.

Table 1: Automatic evaluation results using n-gram based metrics in Japanese-to-English translation.

System	Moses	Cluster	Moses
Corpus	NTCIR-8		+NTCIR-7
Distortion limit	default (6)		unlimited (-1)
DevBLEU	27.70	26.61	28.24
DevNIST	7.1103		7.5114
BLEU	27.42	27.03	30.31
NIST	7.3021		7.5676

4. EXPERIMENTS

The NTCIR-8 Patent Translation Task provides Japanese-English parallel corpora for training and development corpora[6]. The corpora consists of the *official data released after NTCIR-7* and the *new data for NTCIR-8*. When we submitted formal runs, we only used the later part for the training and development corpus, and the formal run of the NTCIR-7 Patent Translation Task for the test corpus. After the result of the automatic evaluation on formal run was announced, we noticed that using both part gives much better results and re-investigated our submitted runs. The formal run of the NTCIR-8 Patent Translation Task was used for this purpose.

As a preprocessing, we tokenized English sentences using `tokenize.perl` and lowercase it. We tokenized Japanese sentences using Mecab after converting the two-byte alphanumeric characters into one-byte and merged them for the formal run submission. We utilized Chasen as the tokenizer with some modification of `.chasenrc` to concatenate a sequence of numbers or alphabetic characters for re-investigating our submission as the organinzers did. Finally we filtered out the too long sentences which has more than 40 words in the training corpus using `clean-corpus-n.perl` [3].

We used Moses and built-in scripts in it to run GIZA++ with parameter tuning based on the minimum error rate training (MERT). In the experiment we only used the NTCIR-8 corpus, we set the distortion limit to default (6). In the experiment using both corpus, we set the distortion limit to unlimited (-1) which is same as the setting of organizers. The alignment heuristic for baseline system is grow-diag-final-and (GDFA). For the language model, we use SRILM to build the 5-gram. To train and parse the non-projective dependency of reordered sentences, the MST Parser was used which is freely availiable[4]. CaboCha analyzes the denpendency structure of the Japanese sentences. We regrads the result of CaboCha as the gold-standard training corpus for the MST Parser. Due to memory restriction, we only used 10,000 sentences to train the MST Parser.

For an intrincsic formal run, we applied the cluster-based method (section 2) to the Japanese-to-English translation and the reordering method (section 3) to the English-to-Japanese translation. The number of cluster we used is 4. For the extrinsic formal run, we splitted the claim in search topics separated by manually defined delimiters and applied the reordering method.

4.1 Automatic Evaluation

The two most popular metrics BLEU and NIST by `mt-evalv13.pl` were used to report on the automatic evaluation of the translation quality[5].

In addition, the head word chain metric (HWCM) is used in order to compare the fluency of the translation[10]. Intuitively, a grammatically correct sentence has higher parsing accuracy than an incorrect one. By comparing the parse tree between the translation from system and human, we measure the fluency of translation.

Table 2: Automatic evaluation results using n-gram based metrics and tree-based metrics in English-to-Japanses translation.

System	Moses	Reordering	Moses	Reordering
Corpus	NTCIR-8		+NTCIR-7	
Distortion limit	6	0	-1	0
DevBLEU	26.94	31.23	30.08	29.27
DevNIST	6.9110	7.1459	7.5493	7.3705
BLEU	27.84	29.03	34.40	29.70
NIST	7.2980	7.1683	7.9172	7.3611
HWCM	11.56	16.87	21.04	11.58
ROOT	11.51	27.30	22.07	21.18

5. DISCUSSION

5.1 Cluster-based SMT

The proposed method achieved a lower score than the PB-SMT system using the NTCIR-8 corpus. Compared with Moses, the cluster-based method degrades the 0.39 BLEU point. Mainly due to the errorneous implementation, we cannot re-investigate the proposed method using both NTCIR-8 and NTCIR-7 corpus.

The provided corpus has less variation in sentence type than the other domain. If we adapt this method to domains which have a variety of sentence types such as conversation or news content, we expect that our method would perform well.

5.2 Reordering method to improve fluency

The proposed method improves the quality of translation significantly in aspects of fluency and also improves adequacy on NTCIR-8 only. [6]Further improvement is not achieved compared with Moses using unlimited distortion when we use whole training corpus.

There are some possible reasons for the difficulty in Japanese-to-English translation using the reordering method.

- Poor alignment
  Incorrect alignment is the root of the errors since our reordering method is based on word alignment. Some English words never match Japanese, for instance, articles such as the, a, an. Some are almost incorrect, for instance, auxiliary verbs such as is, are, does, will. They are zero-fertility words. Filtering alignments on zero-fertility words improves the accuracy of reordering.

- Reordering unit
  Word-level reordering is more sensitive to failure than bunsetsu-level reordering. Because of the larger number of possibilities, the last step suffers due to the large space that needs to be searched.

[3]These scripts is available on the web site for the ACL workshop on SMT. http://www.statmt.org/wmt10/scripts.tgz
[4]http://sourceforge.net/projects/mstparser/

[5]http://www.nist.gov/speech/tools/
[6]We found an implementation error for LTOM [12] and fixed it.

- Absence of case marker
  Japanese has case markers to represent the syntactic role of a bunsetsu. The absence of functional words in English makes the parsing of reordered sentence more difficult than Japanese parsing. [9] proposed a method for fill out this gap by inserting a functional word in the English-to-Korean SMT.

For this reason, we did not carry out a formal run for Japanese-to-English using this method.

6. CONCLUSION

We investigated two different methods for different directions in translation. For Japanese-to-English translations, a cluster-based SMT is used. With an advanced (faster) clustering algorithm, proposed method acheives a slightly lower BLEU score than the PBSMT system. There is possible improvement if we use a separate similarity for clustering TM and LM independently because interpolating both of the similarities may mass up the ability to distinguish a sentence as the correct cluster. Therefore it would be better if we utilize indenpendent clusters for TM and LM othogonally.

On the other hand, we investigated a reordering method to improve fluency of translation. In the trainining phase, reordering utilize word alignment information considering Japanese bunsetsu as unit. In the translating phase, the monotone decoding result is parsed and reordered according to the LTOM of dependency structure. In this way, the reordering method improves the fluency as well as the adequacy.

Acknowledgments

This work was supported in part by the Korea Science and Engineering Foundation (KOSEF) grant funded by the Korean government (MEST No. 2010-0016489) and in part by the BK 21 Project in 2010.

7. REFERENCES

[1] P.-C. Chang and K. Toutanova. A discriminative syntactic word order model for machine translation. In *Proc. of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 9–16, Prague, Czech Republic, June 2007.

[2] P.-C. Chang, H. Tseng, D. Jurafsky, and C. D. Manning. Discriminative reordering with Chinese grammatical relations features. In *Proc. of the Third Workshop on Syntax and Structure in Statistical Translation (SSST-3) at NAACL HLT 2009*, pages 51–59, Boulder, Colorado, June 2009.

[3] M. Collins and N. Duffy. Parsing with a single neuron: Convolution kernels for natural language problems. Technical report, Citeseer, 2001.

[4] A. Finch and E. Sumita. Dynamic model interpolation for statistical machine translation. In *Proc. of the Third Workshop on Statistical Machine Translation*, pages 208–215. Association for Computational Linguistics, 2008.

[5] G. Foster and R. Kuhn. Mixture-model adaptation for SMT. In *Proc. of the Second Workshop on Statistical Machine Translation*, pages 128–135. Association for Computational Linguistics, 2007.

[6] A. Fujii, M. Utiyama, M. Yamamoto, T. Utsuro, T. Ehara, H. Echizen-ya, and S. Shimohata. Overview of the patent translation task at the ntcir-8 workshop. In *Proc. of the 8th NTCIR Workshop Meeting on Evaluation of Information Access Technologies: Information Retrieval, Question Answering and Cross-lingual Information Access*, June 2010.

[7] M. Galley and C. D. Manning. A simple and effective hierarchical phrase reordering model. In *Proc. of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 848–856, Honolulu, Hawaii, October 2008.

[8] S. Hasan and H. Ney. Clustered language models based on regular expressions for SMT. In *Proc. of the 10th Annual Conf. of the European Association for Machine Translation (EAMT)*. Citeseer, 2005.

[9] G. Hong, S.-W. Lee, and H.-C. Rim. Bridging morpho-syntactic gap between source and target sentences for english-korean statistical machine translation. In *Proc. of the ACL-IJCNLP 2009 Conference Short Papers*, pages 233–236, Suntec, Singapore, August 2009.

[10] D. Liu and D. Gildea. Syntactic features for evaluation of machine translation. In *Proc. of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 25–32, Ann Arbor, Michigan, June 2005.

[11] R. McDonald, F. Pereira, K. Ribarov, and J. Hajič. Non-projective dependency parsing using spanning tree algorithms. In *Proc. of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 523–530, Morristown, NJ, USA, 2005.

[12] H. Na, J.-J. Li, J. Kim, and J.-H. Lee. Improving fluency by reordering target constituents using mst parser in English-to-Japanese phrase-based smt. In *Proc. of the twelfth Machine Translation Summit*, pages 276–283, Ottawa, Ontario, Canada, August 2009.

[13] I. Takeshi, T. AKIBA, and I. Katunobu. Effect of the Topic Dependent Translation Models for Patent Translation-Experiment at NTCIR-7.

[14] Y. M. Taku Kudo. Japanese dependency analysis using cascaded chunking. In *Proc. of the 6th Conference on Natural Language Learning 2002 (COLING 2002 Post-Conference Workshops)*, pages 63–69, 2002.

[15] P. Xu, J. Kang, M. Ringgaard, and F. Och. Using a dependency parser to improve smt for subject-object-verb languages. In *Proc. of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 245–253, Boulder, Colorado, June 2009.

[16] H. Yamamoto and E. Sumita. Bilingual cluster based models for statistical machine translation. *IEICE TRANSACTIONS ON INFORMATION AND SYSTEMS E SERIES D*, 91(3):588, 2008.

[17] K. Yasuda, A. Finch, H. Okuma, M. Utiyama, H. Yamamoto, and E. Sumita. System Description of NiCT-ATR SMT for NTCIR-7.