SMT Systems in the University of Tokyo for NTCIR-9 PatentMT

Xianchao Wu*
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku,
Tokyo 113-0033, Japan
wu.xianchao@lab.ntt.co.jp

Takuya Matsuzaki
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku,
Tokyo 113-0033, Japan
matuzaki@is.s.u-tokyo.ac.jp

Jun'ichi Tsujii†
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku,
Tokyo 113-0033, Japan
jtsujii@microsoft.com

ABSTRACT

In this paper, we present two Statistical Machine Translation (SMT) systems and the evaluation results of Tsujii Laboratory in the University of Tokyo (UOTTS) for the NTCIR-9 patent machine translation tasks (PatentMT). This year, we participated in all the three subtasks: bidirectional English-Japanese translations and Chinese-to-English translation. Our first system is a forest-to-string system making use of HPSG forests of source English sentences. We used this system to translate English forests into Japanese. The second system is a re-implementation of a hierarchical phrase based system. We applied this system to all the three subtasks. We describe the training and decoding processes of the two systems and report the translation accuracies of our systems on the official development/test sets.

Categories and Subject Descriptors

I.2.7 [**Natural Language Processing**]: Machine translation

General Terms

Algorithms, Performance

Keywords

Statistical Machine Translation, HPSG forests, hierarchical phrase

Team Name

[UOTTS]

Subtasks/Languages

[English-to-Japanese][Japanese-to-English][Chinese-to-English]

External Resources Used

[GIZA++][SRILM][Mecab][Chasen][Cabocha][Enju][Moses]

*Wu has moved to NTT Communication Science Laboratories. His new address is: 2-4, Hikaridai, Seika-cho, "Keihanna Science City" Kyoto 619-0237 Japan.

†Tsujii has moved to Microsoft Research Asia. His new address is: Room 12165, Building 2, No. 5 Dan Ling Street, Haidian District, Beijing, P.R.China, 100080

1. INTRODUCTION

This paper reports the statistical machine translation (SMT) systems and the evaluation results of Tsujii Laboratory in The University of Tokyo (group name = UOTTS) for the NTCIR-9 patent machine translation tasks (PatentMT). We participated in all the three subtasks this year: the bidirectional English-Japanese translation subtasks and the Chinese-to-English translation subtask. Two SMT systems were constructed and employed for these three subtasks, a forest-to-string system [9, 8] making use of head-driven phrase structure grammar (HPSG) forests, named Akamon [16, 17], for English-to-Japanese translation and an hierarchical phrase-based system [4, 5], named Helios [19], for translating all the three subtasks.

2. AKAMON: AN HPSG FOREST-TO-STRING TRANSLATION SYSTEM

Our forest-to-string system includes the decoding algorithm and the rule extraction algorithm described in [9, 8]. In our forest-to-string system, we used *deep syntactic information* for obtaining fine-grained translation rules. We used Enju[1], a state-of-the-art HPSG parser for English, to generate packed parse forests for English sentences. Deep syntactic information are included in the forests, which includes a fine-grained description of the syntactic property and a semantic representation of the sentence. We extract fine-grained rules from aligned HPSG forest-string pairs and use them in our forest-to-string decoder [16, 17].

2.1 Deep syntactic information for fine-grained translation rule extraction

Head-driven phrase structure grammar (HPSG) is a lexicalist grammar framework. In HPSG, linguistic entities such as words and phrases are represented by a data structure called a *sign*. A sign gives a factored representation of the syntactic features of a word/phrase, as well as a representation of their semantic content. Phrases and words represented by signs are composed into larger phrases by applications of *schemata*. The semantic representation of the new phrase is calculated at the same time. As such, an HPSG parse tree/forest can be considered as a tree/forest of signs (c.f. the HPSG forest in Figure 2 in [16]).

An HPSG parse tree/forest has two attractive properties as a representation of an English sentence in syntax-based

[1]http://www-tsujii.is.s.u-tokyo.ac.jp/enju/index.html

Feature	Description
CAT	phrasal category
XCAT	fine-grained phrasal category
SCHEMA	name of the schema applied in the node
HEAD	*pointer* to the head daughter
SEM_HEAD	*pointer* to the semantic head daughter
CAT	syntactic category
POS	Penn Treebank-style part-of-speech tag
BASE	base form
TENSE	tense of a verb (past, present, untensed)
ASPECT	aspect of a verb (none, perfect, progressive, perfect-progressive)
VOICE	voice of a verb (passive, active)
AUX	auxiliary verb or not (minus, modal, have, be, do, to, copular)
LEXENTRY	lexical entry, with supertags embedded
PRED	type of a predicate
ARG$\langle x \rangle$	*pointer* to semantic arguments, $x = 1..4$

Table 1: Syntactic/semantic features extracted from HPSG signs that are included in the output of Enju. Features in phrasal nodes (top) and lexical nodes (bottom) are listed separately.

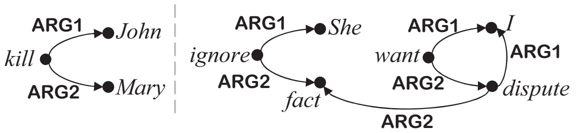

Figure 1: Predicate argument structures for the sentences of "*John killed Mary*" and "*She ignored the fact that I wanted to dispute*".

SMT. First, we can carefully control the condition of the application of a translation rule by exploiting the fine-grained syntactic description in the English parse tree/forest, as well as those in the translation rules. Second, we can identify sub-trees in a parse tree/forest that correspond to basic units of the semantics, namely sub-trees covering a predicate and its arguments, by using the semantic representation given in the signs. We expect that extraction of translation rules based on such *semantically-connected* sub-trees will give a compact and effective set of translation rules.

A sign in the HPSG tree/forest is represented by a typed feature structure (TFS) [1]. A TFS is a directed-acyclic graph (DAG) wherein the edges are labeled with feature names and the nodes (feature values) are typed. In the original HPSG formalism, the types are defined in a hierarchy and the DAG can have arbitrary shape (e.g., it can be of any depth). We however use a simplified form of TFS, for simplicity of the algorithms. In the simplified form, a TFS is converted to a (flat) set of pairs of feature names and their values. Table 1 lists the features used in our system, which are a subset of those in the original output from Enju.

2.2 Realigning Japanese function words

Furthermore, we effectively used Japanese function words to generate generalized translation rules for forest-based translation. Given aligned forest-string pairs, we extract composed tree-to-string translation rules that account for multiple interpretations of both aligned and unaligned target function words. In order to constrain the exhaustive attachments of function words, we limit to bind them to the nearby syntactic chunks yielded by a target dependency parser, Cabocha[2] [7]. Therefore, the proposed approach can not only capture source-tree-to-target-chunk correspondences but can also use forest structures that compactly encode an exponential number of parse trees to properly generate target function words during decoding. The detail of this idea is described in our previous work [17].

3. HELIOS: A HIERARCHICAL PHRASE-BASED SYSTEM

Our hierarchical phrase-based system is a re-implementation of the Hiero system [4, 5]. The system utilizes synchronous context free grammar (SCFG) rules for decoding. In Hiero and our Helios system, there are three types of rules:

1. flat phrasal rules which are flat bilingual phrasal pairs consisting of consecutive source/target words (e.g., X $\rightarrow$ $\langle$the fluid, ryutai$\rangle$);

2. hierarchical phrase rules in which words and variables (place-holders) are used (e.g., $X \rightarrow \langle X_1$ of $X_2, X_2\, X_1\rangle$); and,

3. glue rules which are used to heuristically and serially merge two phrasal outputs together (i.e., $S \rightarrow \langle S_1X_2, S_1X_2\rangle$ and $S \rightarrow \langle X_1, X_1\rangle$).

Thus, flat phrasal rules are more like phrasal translation dictionaries and can capture short distance reordering within phrases, while hierarchical rules can capture relatively long distance reordering between phrases. External parsers are not necessary for Helios since these translation rules can be extracted from word-aligned sentence pairs. This makes our system easily applicable to any language pairs only if the parallel training corpora are given beforehand. Please refer to [19] for the detailed description of this system.

4. EXPERIMENTS

4.1 Setup

Similar to the baseline configurations[3] supported by NTCIR-9, we use the following tools for data preparation:

- GIZA++ [11]: giza-pp-v1.0.3[4] to generate source-to-target word-alignments and *grow-diag-final* [6] strategy for symmetrizing the bidirectional alignments,

- SRILM [13]: version 1.5.12[5] with modified Kneser-Ney smoothing [3] to train and manage 5-gram target language models,

- Additional Scripts[6] for tokenize, detokenize, lowercase, recase, etc.,

- Stanford Chinese Segmenter[7] [2]: version 2008-05-21 using Chinese Penn Treebank (CTB) standard to segment Chinese sentences into word sequences,

[2]http://chasen.org/~taku/software/cabocha/
[3]http://ntcir.nii.ac.jp/PatentMT/baselineSystems
[4]http://giza-pp.googlecode.com/files/giza-pp-v1.0.3.tar.gz
[5]http://www.speech.sri.com/projects/srilm/
[6]http://homepages.inf.ed.ac.uk/jschroe1/how-to/scripts.tgz
[7]http://www-nlp.stanford.edu/downloads/segmenter.shtml

Table 2: Statistics of bi-directional Japanese-English experiment sets for Helios. Here, Mecab was used for Japanese word segmentation. Also, En2Jp is short for English-to-Japanese translation and Jp2En is short for Japanese-to-English translation.

	# sent.	# Ja words	# En words
train	2,963,963	98,923,854	86,048,310
Deva	1,000	37,066	31,890
Devb	1,000	35,921	31,935
Test-En2Jp	2,000	78,587	70,624
Test-Jp2En	2,000	74,070	69,521

- Mecab: version 0.98[8] with dictionary mecab-ipadic-2.7.0-20070801.tar.gz[9] to segment Japanese sentences into word sequences,
- Chasen: version 2.4.4[10] was also used to segment Japanese sentences into word sequences for training our forest-to-string decoder,
- Enju: version 2.3.1[11] was used to generate English HPSG parse forests for English-to-Japanese translation.

In addition, we used Moses[12] with revision = "3717" to train an English recase model using the English sentences in the parallel training sentences.

4.2 Statistics of Data

Table 2 shows the statistics of the parallel Japanese-English data for training, tuning, and testing of our Helios system. Since the reference sentences for the official test sets was not available before the result submission time, we split the original development set averagely into two parts. The first part, named deva, is used as our experimental development set. The second part, named devb, is used as our experimental test set. Through this setting, we can tune the hyper parameters (e.g., maximum reordering length, maximum number of words to scan during rule matching, etc.) in our system by minimum-error rate training (MERT) [10] on deva and comparing the BLEU [12] scores on devb. Then, we use all the 2,000 sentences in the development set for MERT and report the translation accuracies on the final test set(s) with 2,000 sentences.

We filtered out parallel sentences that are too long (# of words > 64) to be used for training GIZA++. In the training set, there are averagely 33.4 and 29.0 words in the Japanese and the English sentences, respectively. A Japanese-to-English decoder and an English-to-Japanese decoder are trained based on the filtered parallel corpus. The monolingual sentences in the original training corpus are used to train a 5-gram English language model and a 5-gram Japanese language model.

In order to save computing time, we further filtered out parallel sentences that are longer than 40 and use the remaining parallel sentences to train our forest-to-string decoder. The statistics of the filtered data sets are shown in Table 3. Note that, different from Table 2, we used Chasen instead of Mecab for Japanese word segmentation. Through referring to the English sentences, we further combined the split number and English words together in the Chasen outputs. For example, '1 2 3' will be recovered back to '123', 'c o m e b a c k' will be recovered back to 'come back'.

[8]http://sourceforge.net/projects/mecab/files/
[9]http://sourceforge.net/projects/mecab/files/mecab-ipadic/
[10]http://chasen-legacy.sourceforge.jp/
[11]http://www-tsujii.is.s.u-tokyo.ac.jp/enju/
[12]http://www.statmt.org/moses/

Table 3: Statistics of English-to-Japanese experiment sets for Akamon. Chasen was used for Japanese word segmentation.

	Train	Deva	Devb	Test
# sentences	2,018,214	1,000	1,000	2,000
# En words	49,474,332	31,890	31,935	70,624
# of En forests	1,987,526	989	987	1,966
parse succ. rate	98.5%	98.9%	98.7%	98.3%
# Ja words	53,271,286	37,262	36,200	73,984

Table 4: Statistics of Chinese-to-English experiment sets.

	Train	Deva	Devb	Test
# sent.	999,950	1,000	1,000	2,000
# Ch words	37,656,651	36,051	37,267	54,228
# En words	42,347,290	38,674	38,873	58,172

In the training set as shown in Table 3, there are averagely 26.4 and 24.5 words respectively in the Japanese and the English sentences. HPSG forests were successfully created for 98.5% of the two million training sentences. The parse success rates are in the similar level for the development sets and the final test set. We prune the original HPSG forests to save the time for rule extraction. Using the pruning criteria expressed in [8], we continue to prune a parse forest by setting p_e to be 8, 5, and 2, until there are no more than $e^{10} = 22,026$ trees in a forest. After pruning, there are an average of 133.6 trees in a parse forest.

Table 4 shows the statistics of the Chinese-to-English parallel corpus. From the training data, we filtered out 50 parallel sentences in which there is only one strange word in one language side. In the training set, there are averagely 37.7 and 42.3 words respectively in the Chinese and the English sentences.

4.3 Training and Decoding

Figure 2 shows the training and tuning progress of our Helios system. Given original bilingual parallel corpora, we first tokenize and lowercase the source and target sentences (e.g., word segmentation of Chinese and Japanese, punctuation segmentation of English). The pre-processed monolingual sentences will be used by SRILM [13] to train a n-gram language model. In addition, we filter out too long sentences here, i.e., only relatively short sentence pairs will be used to train word alignments. Then, we use GIZA++ [11] and *grow-diag-final* symmetric strategy [6] on the tokenized parallel corpus to obtain a word-aligned parallel corpus. A hierarchical phrase translation rule set is extracted from the word-aligned parallel corpus. Taking the target language model, the rule set, and the preprocessed development set as inputs, we perform minimum-error rate training (MERT) [10] on the decoder to tune the weights of the features.

Figure 3 shows the training progress of our Akamon system. The progress is about the same with that shown in

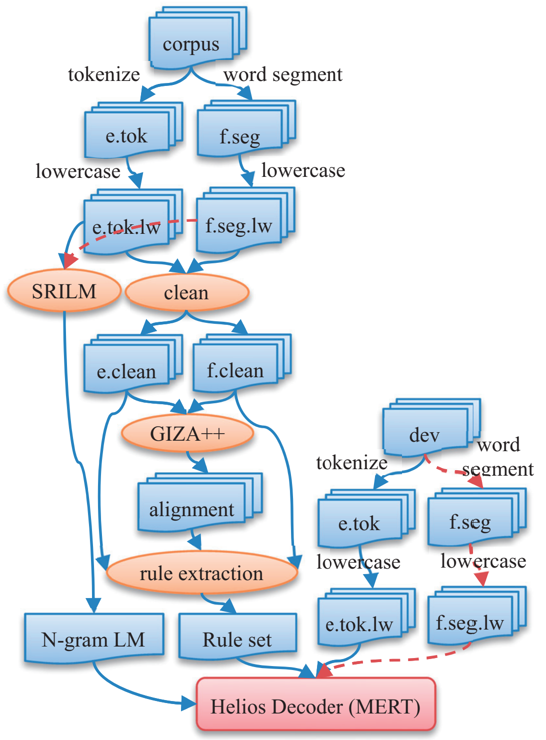

Figure 2: Training and tuning process of our Helios system.

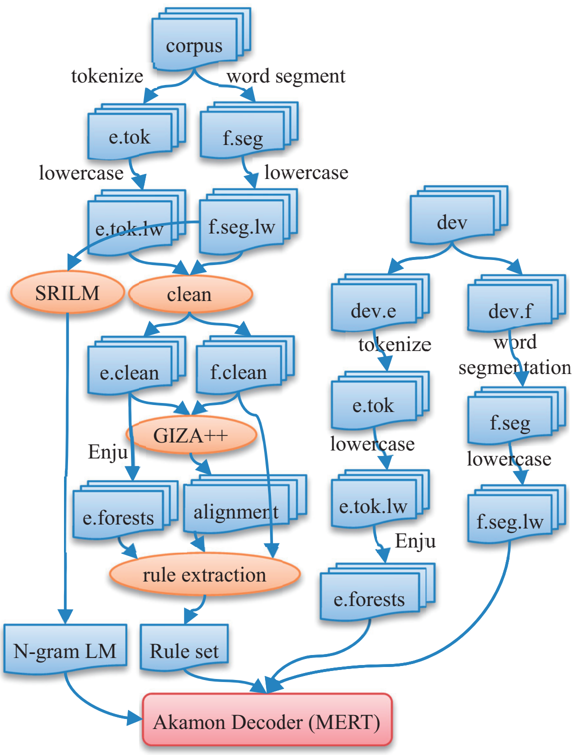

Figure 3: Training and tuning process of our Akamon system.

Table 5: Statistics of translation rules for the Helios system.

	# flat phrase	# H-rules	# total rules	H-rule ratio
dev-C2E	591,030	5,542,141	6,133,171	90.4%
test-C2E	481,627	4,245,317	4,726,944	89.8%
dev-J2E	850,748	5,017,407	5,868,155	85.5%
test-J2E	926,523	7,723,336	8,649,859	89.3%
dev-E2J	532,721	4,506,717	5,039,438	89.4%
test-E2J	622,040	5,311,912	5,933,952	89.5%

Figure 2. The main difference is that we use Enju to generate HPSG forests for the English sentences and extract tree-to-string rules. The inputs for MERT also include the HPSG forests of the English sentences in the development set.

Using the parallel training data shown in Table 4 and Table 2, we extract flat and hierarchical phrase table for Helios, as listed in Table 5. From this table, we can see that there are around 85% to 90% hierarchical rules in the final translation table. There are averagely 2.3K to 4.3K translation rules that are available to each sentence in the development/test sets.

Using the parallel training data shown in Table 3, we extract fine-grained tree-to-string translation rules [16] for Akamon, as listed in Table 6. From this table, we can see that there are 89.2% reordering rules in the final translation table. However, for each tree type, there are only averagely 1.03 tree rules. In addition, the average number of tree nodes is 57.1, which is relatively too large to be used during decoding in an acceptable decoding time. Thus, we prune again the final rule set basing on the fragmental score ($>=$ 0.0001) and the number of words ($<=$ 15) in the target language side of tree-to-string rules. The statistics of the pruned rule set is shown in Table 6 as well. Under this pruning configuration, 90.2% rules in the original rule set were filtered out. Consequently, the average number of tree nodes drops from 57.1 to 24.5 in the pruned rule set, and to 3.6 in the pruned rule subsets that were used in the development set and the final test set. In addition, for each tree type, there are averagely 1.38, 31.86, and 30.67 tree rules. Through these numbers, we argue the pruning process does filter out the less-generalized rules which were too big or whose frequencies were too small. Figure 4 shows the distribution of tree nodes in four rule sets: the original rule set (total), the pruned rule set (pruned), the subset of pruned rule set that was used in the development text (dev) and the test set (tst). Finally, there are averagely 4.8K and 4.9K translation rules that are available to each sentence in the development/test sets.

4.4 Helios and Akamon Results

Table 7 shows the case-insensitive BLEU-4 [12] scores of our Helios system on all the three translation tasks. In this table, DL denotes the maximum distortion length, i.e., the maximum word span for matching hierarchical rules. For Chinese-to-English translation, we achieved the best BLEU scores for the development set by setting DL to be 15. Note that in the test set, we achieved the best BLEU scores under DL=20. However, there are no significant difference by setting DL to be 15 or 20 in the final test set. As DL increases,

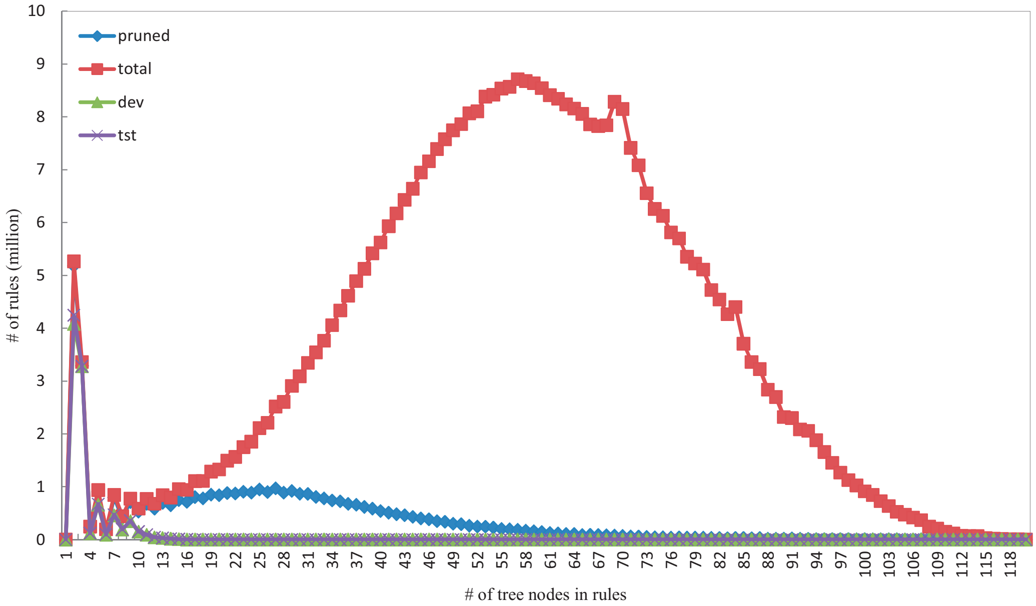

Figure 4: Distribution of the number of tree nodes in the tree-to-string translation rule set.

Table 6: Statistics of the tree-to-string translation rules that were gained from the training data, pruned by fragmental score and the number of Japanese words/variables, and applied to the development/test sets.

	Total	Pruned	Dev	Test
# rules	442,512,218	43,422,988	9,561,065	9,796,070
# reordering rules	394,599,394	27,342,822	3,162,747	3,174,074
reordering ratio	89.2%	63.0%	33.1%	32.4%
# tree types	428,637,187	31,372,733	300,073	319,424
# candidates per tree type (avg.)	1.03	1.38	31.86	30.67
# tree nodes (avg.)	57.1	24.5	3.6	3.6

Table 7: C2E, J2E, and E2J case-insensitive BLEU-4 scores by using Helios.

	DL	mteval-v11b.pl (deva-MERT, devb-TEST)	mteval-v12.pl (deva-MERT, devb-TEST)	Decoding Time (sec./sent.)	mteval-v11b.pl (dev-MERT, test-TEST)	mteval-v12.pl (dev-MERT, test-TEST)
C2E	5	0.3136	0.3209	2.15	0.3074	0.3097
C2E	10	0.3287	0.3362	3.66	0.3247	0.3264
C2E	15	**0.3379**	**0.3448**	6.38	0.3226	0.3242
C2E	20	0.3347	0.3420	9.67	**0.3227**	**0.3243**
J2E	5	0.2490	0.2499	3.29	0.2412	0.2436
J2E	10	0.2728	0.2742	4.81	0.2663	0.2687
J2E	15	**0.2843**	0.2847	8.38	0.2745	0.2770
J2E	20	0.2838	**0.2848**	15.01	**0.2752**	**0.2772**
J2E (bug free)	20	-	-	-	**0.3061**	**0.3089**
E2J	5	0.2552	0.2544	0.94	0.2531	0.2525
E2J	10	0.2809	0.2801	2.14	0.2815	0.2808
E2J	15	0.2855	0.2847	3.96	0.2853	0.2842
E2J	20	**0.2887**	**0.2878**	4.91	**0.2888**	**0.2881**
E2J (bug free)	20	-	-	-	**0.3204**	**0.3199**

the time needed for decoding each sentence also (approximately linearly) increases. For English-Japanese translations, we achieved the best BLEU scores by setting DL to be 20.

Unfortunately, we found a serious bug in our pre-processing logic of the English and Japanese sentences. The bug is that, we used halfwidth Japanese characters for training, yet used fullwidth Japanese characters for MERT and testing. After fixing this bug, we gained significant improvements on both Japanese-to-English and English-to-Japanese translation. For Japanese-to-English translation, we gained a optimal BLEU score of 0.3089 on the test set, which is quite close to the best result of 0.3169. For English-to-Japanese translation, we gained a optimal BLEU score of 0.3204, which is

Table 8: Comparison of our system and other baseline systems for Chinese-to-English translation.

dev=MERT, test=TEST	BLEU	NIST	RIBES	adequency	acceptability pairwise comparison score	(tie)
Moses-Hiero-ze-1	0.3072	7.9025	0.7719	3.290	0.476	0.273
Moses-ze-1	0.2932	7.7498	0.7284	2.893	NA	NA
BestGroup-ze-1	0.3944	8.9112	0.8327	4.033	0.744	0.168
UOTTS-ze-1 (DL=15)	0.3074	7.8917	0.7662	3.293	0.441	0.278
UOTTS-ze-2 (DL=20)	0.3067	7.874	0.7678	NA	NA	NA

Table 9: Comparison of our system and other baseline systems for Japanese-to-English translation.

dev=MERT, test=TEST	BLEU	NIST	RIBES	adequency	acceptability pairwise comparison score	(tie)
Moses-Hiero	0.2895	7.7696	0.70644	2.617	0.474	0.331
Moses	0.2861	7.7562	0.675831	2.427	0.447	0.343
BestGroup	0.3169	7.8161	0.740397	3.430	0.638	0.249
UOTTS-1 (DL=10)	0.2605	7.5903	0.6732	2.377	0.425	0.345
UOTTS-2 (DL=15)	0.2697	7.6936	0.6976	NA	NA	NA

Table 10: Comparison of our systems and other baseline systems (where NTT-UT stands for the joint group of NTT Communication Science Laboratories and The University of Tokyo) for English-to-Japanese translation.

dev=MERT, test=TEST	BLEU	NIST	RIBES	adequency	acceptability pairwise comparison score	(tie)
Moses-Hiero	0.3166	7.7954	0.7200	2.603	0.472	0.304
Moses	0.3190	7.8811	0.7068	2.477	0.456	0.308
BestGroup (NTT-UT)	0.3948	8.7134	0.7813	3.670	0.695	0.198
UOTTS (Akamon)	0.2799	7.2575	0.6861	2.193	0.411	0.315
UOTTS (Helios, DL=20)	0.2781	7.2363	0.6899	NA	NA	NA

3.16 (%) points better than our former optimal BLEU score of 0.2888.

Table 8 shows the comparison of our submitted results (with DL=15 and 20, case sensitive BLEU-4) and the baseline systems for Chinese-to-English translation. Our submitted result is slightly better then the two baseline systems (Moses). However, our systems are still far from the top-1 system. We take this as an important chance for communicate with other groups to push the improvement of our systems.

Table 9 shows the comparison of our submitted results (with DL=10 and 15) and the baseline systems for Japanese-to-English translation. Generally, it turned out that even with the same data, Japanese-to-English translation achieved a relatively low BLEU score than English-to-Japanese translation. It will be meaningful to further investigate the detailed reasons for this observation. Furthermore, pre-ordering [18] and post-ordering [14] techniques have been proposed by us for further improving the translation accuracies of Japanese-to-English translation.

The comparison of our submitted results (Akamon and Helios system) and the baseline systems for English-to-Japanese translation is shown in Table 10. In this table, group NTT-UT (NTT Communication Science Laboratories with our group) achieved the best results. Please refer to [15] for the detailed techniques of system combination.

5. CONCLUSIONS

We have described the detailed training and decoding process of a forest-to-string SMT system and a hierarchical-phrase based SMT system in the University of Tokyo. Making use of these two systems, we participated in all the three translation tasks held by NTCIR-9 and achieved good ranks. In addition, we support the n-best lists of our systems to NTT Communication Science Laboratories for system combination. The combination results achieved top-level ranks among all the teams. We will concentrate to update both the theoretical and the practical aspects of our systems and make them open-source to be wildly used by the SMT research communities.

6. REFERENCES

[1] B. Carpenter. *The Logic of Typed Feature Structures.* Cambridge University Press, 1992.

[2] P.-C. Chang, M. Galley, and C. Manning. Optimizing chinese word segmentation for machine translation performance. In *Proceedings of ACL 3rd Workshop on Statistical Machine Translation*, 2008.

[3] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University Center for Research in Computing Technology, 1998.

[4] D. Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL*, pages 263–270, Ann Arbor, MI, 2005.

[5] D. Chiang. Hierarchical phrase-based translation. *Computational Lingustics*, 33(2):201–228, 2007.

[6] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the*

ACL 2007 Demo and Poster Sessions, pages 177–180, 2007.

[7] T. Kudo and Y. Matsumoto. Japanese dependency analysis using cascaded chunking. In *Proceedings of CoNLL-2002*, pages 63–69. Taipei, Taiwan, 2002.

[8] H. Mi and L. Huang. Forest-based translation rule extraction. In *Proceedings of EMNLP*, pages 206–214, October 2008.

[9] H. Mi, L. Huang, and Q. Liu. Forest-based translation. In *Proceedings of ACL-08:HLT*, pages 192–199, Columbus, Ohio, 2008.

[10] F. J. Och. Minimum error rate training in statistical machine translation. In *Proceedings of ACL*, pages 160–167, 2003.

[11] F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

[12] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318, 2002.

[13] A. Stolcke. Srilm-an extensible language modeling toolkit. In *Proceedings of International Conference on Spoken Language Processing*, pages 901–904, 2002.

[14] K. Sudoh, X. Wu, K. Duh, H. Tsukada, and M. Nagata. Post-ordering in statistical machine translation. In *Proceedings of MT Summit*, Xiamen, China, September 2011.

[15] K. Sudoh, X. Wu, K. Duh, H. Tsukada, M. Nagata, T. Matsuzaki, and J. Tsujii. Ntt-ut statistical machine translation in ntcir-9 patentmt. In *Proceedings of NTCIR-9*, Tokyo, Japan, December 2011.

[16] X. Wu, T. Matsuzaki, and J. Tsujii. Fine-grained tree-to-string translation rule extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 325–334, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[17] X. Wu, T. Matsuzaki, and J. Tsujii. Effective use of function words for rule generalization in forest-based translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, Oregon, June 2011. Association for Computational Linguistics.

[18] X. Wu, K. Sudoh, K. Duh, H. Tsukada, and M. Nagata. Extracting pre-ordering rules from chunk-based dependency trees for japanese-to-english translation. In *Proceedings of MT Summit*, Xiamen, China, September 2011.

[19] X. Wu and J. Tsujii. A term translation system using hierarchical phrases and morphemes. In *Proceedings of Japan Natural Language Processing Conference*, Toyohashi, Japan, March 2011.