Translation systems and experimental results of the EHR group for WAT2016 tasks

Terumasa EHARA
Ehara NLP Research Laboratory
Seijo, Setagaya, Tokyo, JAPAN

eharate @ gmail. com

Abstract

System architecture, experimental settings and experimental results of the EHR group for the WAT2016 tasks are described. We participate in six tasks: en-ja, zh-ja, JPCzh-ja, JPCko-ja, HINDENen-hi and HINDENhi-ja. Although the basic architecture of our systems is PBSMT with reordering, several techniques are conducted. Especially, the system for the HINDENhi-ja task with pivoting by English uses the reordering technique. Because Hindi and Japanese are both OV type languages and English is a VO type language, we can use reordering technique to the pivot language. We can improve BLEU score from 7.47 to 7.66 by the reordering technique for the sentence level pivoting of this task.

1 Introduction

Reasonably sized bilingual corpus is needed for the training of a SMT system. Unfortunately, between most of the Asian language pairs, we do not have such corpora, except a few language pairs like between Japanese, Chinese and Korean. On the other hand, between English and most of the Asian languages, we can obtain a large-sized bilingual corpus. So, we can use pivoting techniques (Utiyama and Isahara, 2007) for the SMT between Asian languages with English as the pivot language. HINDENhi-ja task of WAT2016 is an example. English is the VO type language and several Asian languages are OV type. Hindi and Japanese are both OV type languages. For the SMT between VO type and OV type languages, reordering technique is effective (Xu et al., 2009; Isozaki et al., 2010). We can use this reordering technique for SMT between OV type languages with a VO type pivot language. We apply this method to HINDENhi-ja task.

We participate in en-ja, zh-ja, JPCzh-ja, JPCko-ja, HINDENen-hi and HINDENhi-ja tasks in WAT2016. We will describe overall architecture of our systems and experimental settings in section 2. We will focus on the pivoting technique with reordering in section 3. In section 4, we will show the experimental results. We will conclude our discussion in section 5.

2 System architecture and experimental settings

2.1 Overall system architecture

Our basic system architecture is PBSMT (simply SMT) with reordering. We use Moses v. 3 (Koehn et al., 2003) for SMT tool and MGIZA++ v. 0.7.0 (Och and Ney, 2003; Gao and Vogel, 2008) for the alignment tool. Additional techniques used in our systems are listed in Table 1.

Character-based segmentation (Hyoung-Gyu Lee et al., 2015) for Chinese, Korean and Japanese is used in addition to word-based segmentation for the SMT. RBMT plus SPE (statistical post-editing) technique (Ehara, 2007) is applied to the JPCzh-ja task. Reordering is used for all tasks except for the

JPCko-ja task, because of the word order similarity of Korean and Japanese. Pivoting is used for the HINDENhi-ja task.

Task	Word-based PBSMT	Character-based PBSMT	RBMT+SPE	Reordering	Pivoting
en-ja	✔			✔	
zh-ja	✔	✔		✔	
JPCzh-ja	✔	✔	✔	✔	
JPCko-ja	✔	✔			
HINDENen-hi	✔			✔	
HINDENhi-ja	✔			✔	✔

Table 1: Used techniques for the tasks

2.2 en-ja task and HINDENen-hi task

Our English-Hindi or English-Japanese translation system is ordinary word-based SMT with reordering from a VO type language to an OV type language. Our reordering method is rule-based and basically same as (Ehara, 2015) in both tasks. Here, we briefly explain it. English sentences in training corpus are parsed in n-best (n=100) by the Berkeley parser v.1.7 (Petrov et al., 2006)[1] and they are reordered in k-best ($k \leqq n$)[2] by our rule-based reordering tool. Next, we rerank k-best reordered English sentences by alignment score between English and Hindi or English and Japanese. In the last year's work (Ehara, 2015), we used WER to measure alignment score, however, this year, we use Kendall's tau. Another new thing is that reordering process adopts iterative loop consisting of alignment and reranking. Change of Kendall's tau by this iteration will be shown in section 4.1.

For reordering of dev, devtest and test sentences, we use the LM score calculated by "query" command in Moses to rerank k-best reordered sentences. This LM is trained by the reordered training sentences of the TM training corpus. This reranking method is common to Chinese reordering.

For English sentence reordering, deleting articles ("a", "an" and "the") and adding two case markers (subject and object) (Isozaki et al., 2010) are applied, both in en-ja and HINDENen-hi tasks.

We use Moses' tokenizer for English tokenization, Indic NLP normalizer and tokenizer (Kunchukuttan et al., 2014) for Hindi tokenization and JUMAN (Kurohashi et al., 1994) for Japanese segmentation.

SMT training, tuning, and testing are done by Moses with the default settings. Alignment process in the reordering is done by Moses (MGIZA++) and self-built post processor for GIZA++ outputs. This post processor makes word alignment by heuristics using two GIZA++ outputs (A3.final) and two lexical translation tables (lex.f2e and lex.e2f) obtained by Moses.

Experimental setting is as follows. Training data for SMT and reordering are provided from the organizer. The number of TM training sentence pairs in en-ja task are 1,502,767 shown in Table 2. We extract sentence pairs which have high sentence alignment score ($\geqq 0.08$) from the three training corpora. We also filter out sentence pairs which are too long (> 100 words) or have unbalanced sentence length (ratio of word numbers is > 4 or < 0.25). The latter filtering is common with all tasks.

Task	TM training	LM training
en-ja	1,502,767	3,824,408
zh-ja	667,922	3,680,815
JPCzh-ja	995,385	4,186,284
JPCko-ja	996,339	5,186,284
HINDENen-hi	1,450,896	1,599,708
HINDENhi-ja	149,743	406,766

Table 2: Training corpus size (sentences)
(In HINDENhi-ja task, the corpus size is the case of direct translation without pivoting.)

[1] Sentences unparsed by the Berkeley parser are n-best parsed by the Stanford lexicalized parser v.2.0.5 (Klein and Manning, 2003; Levy and Manning, 2003).

[2] Several different parse trees may make same reordered word order. Then $k \leqq n$.

The number of TM training sentence pairs in HINDENen-hi task are 1,450,896. We filter out 21,637 strange sentence pairs such as hi: “à¤¸à¤ à¤ à¥ à¤°à¤¹ à¤¸à¥” and en: “Damping :” from the original training corpus.

LM are trained by the tool lmplz in Moses with order 6. This LM training method is common with all tasks. The number of LM training sentences in the en-ja task are 3,824,408. They are extracted from all en-ja training corpora and HINDENhi-ja training corpus. The number of LM training sentences in the HINDENen-hi task are 1,599,708. They are extracted from HINDENen-hi training corpus and HINDENhi-ja training corpus.

The distortion limit for tuning and testing is 6. This DL value is same in all tasks except for JPCko-ja task.

2.3 zh-ja task and JPCzh-ja task

Our Chinese-Japanese translation system is SMT with reordering. Additionally, JPCzh-ja task uses RBMT plus SPE system. In both tasks, our reordering method is basically same as the method described in section 2.2 except for the parsing rules and the reordering rules. For Chinese sentence reordering, deleting case markers of Japanese (“が”, “を” ,“は” and “は 、”) is done to improve Chinese-Japanese alignment accuracy.

We use Stanford Chinese segmenter (Chang et al., 2008) to segment Chinese. We apply self-built post processor to the output of the Stanford segmenter. This post processor segments alphabetical, numerical and symbolic expressions from Han expressions. It also adjusts several segmentations for Han expressions, for example, “示于表” is segmented to “示 于 表”. We use JUMAN for Japanese segmentation. We also add self-built post processor after JUMAN’s segmentation that connects forms that have the POS “名詞 サ変名詞” to the forms that have the POS “動詞 * サ変動詞”. It also connects forms that have the POS “動詞” to the forms that have the POS “接尾辞 動詞性接尾辞”. As the result, “備えて いる” consisting of two morphemes comes to one morpheme. By these post processors, we can balance segmentation granularity between Chinese and Japanese.

For character based SMT, we segment all Han characters only for the Chinese side.

SMT training, tuning, and testing using reordered Chinese and Japanese corpus are done by Moses.

RBMT system and SPE system for JPCzh-ja task are the same as in the previous work (Ehara, 2015).

Two outputs from word based SMT and character based SMT are packed to lattice form using DP matching and the path which has the best LM score is selected as the system output. This method is similar but simpler than the Matusov’s method (Matusov et al., 2006). In the case of three outputs from word based SMT, character based SMT and RBMT plus SPE, this packing and selecting procedures are made twice.

Experimental setting is as follows. Training data for SMT and reordering are provided from the organizer and NTCIR-10’s PatentMT task EJ subtask site (Goto et al. 2013). The number of TM training sentence pairs in zh-ja task are 667,922. They are extracted from zh-ja task data. The number of TM training sentence pairs in JPCzh-ja task are 995,385. They are extracted from JPCzh-ja task data.

The number of LM training sentences in zh-ja task are 3,680,815. They are extracted from en-ja and zh-ja task data. The number of LM training sentences in JPCzh-ja task are 4,186,284. They are extracted from JPCzh-ja task data and NTCIR-10’s PatentMT task EJ subtask’s data.

2.4 JPCko-ja task

Korean and Japanese are both OV type language and have a very similar word order structure. So we don’t use reordering in this task. We use Mecab-ko[3] for tokenization of Korean and JUMAN for segmentation of Japanese. No Japanese post processor is used for the segmentation for word based SMT in this task. For character based SMT, we segment all characters both for Korean and Japanese.

SMT training, tuning and test using Korean and Japanese corpus is done by Moses.

One problem in JPCko-ja task is an unbalanced usage of parentheses between two languages. As described in (Ehara, 2015), there are mostly parentheses surrounding a number in Korean. On the other hand, there are mostly no parentheses surrounding a number in Japanese. We conduct three methods to

[3] https://bitbucket.org/eunjeon/mecab-ko/

address the problem. (1) All parentheses surrounding a number in the Korean side are deleted. (2) Parentheses are added to a number in the Japanese side not surrounded by parentheses and aligned to the number of the Korean side that is surrounded by parentheses. (3) Same as (2) to SMT phase and after decoding parentheses surrounding a number in the translated Japanese are deleted.

Experimental setting is as follows. Training data for SMT and reordering are provided from the organizer and NTCIR-10's PatentMT task EJ subtask site. The number of TM training sentence pairs in JPCko-ja task are 996,339. They are extracted from JPCko-ja task data.

The number of LM training sentence pairs in JPCko-ja task are 5,186,284. They are extracted from JPCko-ja task data, JPCzh-ja task data and NTCIR-10's PatentMT task EJ subtask's data.

The distortion limit for tuning and testing is set to 0, because the word order of Korean is similar to that of Japanese.

2.5 HINDENhi-ja task

Four methods of translation is executed for HINDENhi-ja task. (1) Ordinary PBSMT trained by Hindi-Japanese bilingual corpus. (2) Using Hindi-English and English-Japanese bilingual corpora, pivoting by sentence level without reordering. (3) Same as (2) except for using reordering. (4) Pivoting by table level with reordering.

Pivoting will be described in section 3. Here, we describe other techniques used in this task.

Words in dev and test set not included in the training corpus (train-parallel.hi) is translated by an online free translator and from this translation list, we make a Hindi-Japanese user dictionary, which has 931 entries. This user dictionary is used in the TM training in addition to the training data.

In the TM training of the case (1), we use filtered training data of HINDENhi-ja task. The filtering method is like the method described in section 2.2. For example, the datum hi: "इंटरफ़ेस विकल्प" and ja: "ã ªã ã ·ã §ã ³>>" is discarded. As the result, we get 148,812 Hindi and Japanese sentence pairs. Adding user dictionary described above, we get 149,743 sentence pairs to train the TM.

Japanese LM is trained not only by the task data but by the TED corpus (Cettolo, 2012)[4]. We extract 260,501 sentences from the <content> part of ted_ja-20160408.xml. Sentences which include sentences in dev or test set are filtered out. As the result, we get 256,891 sentences. Adding Japanese side of the original training data of HINDENhi-ja task, we get 406,766 sentences to train the LM.

SMT training, tuning, testing and pivoting are done by Moses.

3 Pivoting

For the sentence level pivoting, we use the English-Japanese SMT system described in section 2.2. The English side of the TM training data of HINDENen-hi task (1,450,896 sentences) is translated into Japanese by this SMT system, without reordering (case (2)) or with reordering (case (3)). As the result, we get 1,450,896 Hindi and (machine translated) Japanese sentence pairs. Adding TM training corpus of HINDENhi-ja task, we get 1,600,639 sentence pairs to train the TM.

For the table level pivoting, we merge two phrase tables and two reordering tables. They are Hindi-English phrase table and Hindi-English reordering table used in the HINDENen-hi task and English-Japanese phrase table and English-Japanese reordering table used in the en-ja task. Merging two phrase tables, we get Hindi-Japanese pivoted phrase table. Merging two reordering tables, we get Hindi-Japanese pivoted reordering table.

The merging method of the phrase tables is like the "triangulation" (Cohn and Lapata, 2007; Wu and Wang, 2007). Explicitly, four conditional probabilities in the Moses' phrase table are computed as:

$$\emptyset(f|e) = \sum_p \emptyset(f|p)\,\emptyset(p|e)$$

$$lex(f|e) = \sum_p lex(f|p)\,lex(p|e)$$

$$\emptyset(e|f) = \sum_p \emptyset(e|p)\,\emptyset(p|f)$$

$$lex(e|f) = \sum_p lex(e|p)\,lex(p|f)$$

[4] https://wit3.fbk.eu/ (accessed on 17th August 2016).

where, f, p and e means a source, pivot and target phrases, respectively. We discard data which have a low probability from the pivoted phrase table that is $\emptyset(f|e)\emptyset(e|f) < 0.000001$. We merge this pivoted phrase table and the direct phrase table trained by the Hindi-Japanese parallel corpus (see section 2.5). In the merging, we use token frequencies of f and e obtained from two phrase tables. Considering $\emptyset_p(f|e)$ be the conditional probability in the pivoted phrase table, $\emptyset_d(f|e)$ be the conditional probability in the direct phrase table, $\mathrm{F}_p(f)$ be the frequency of f in the pivoted phrase table and $\mathrm{F}_d(f)$ be the frequency of f in the direct phrase table, we get the merged conditional probability:

$$\emptyset(f|e) = \frac{\emptyset_p(f|e)\,\mathrm{F}_p(f) + \emptyset_d(f|e)\,\mathrm{F}_d(f)}{\mathrm{F}_p(f) + \mathrm{F}_d(f)}.$$

Other probabilities are similarly calculated.

The merging method of the reordering tables is as follows. We assume that the source to target reordering orientations are determined as in Table 3. Here we use the combination of three reordering orientations: monotone (m), swap (s) and discontinuous (d) in the source to pivot reordering table (fp) and pivot to target reordering table (pe).

fp＼pe	m	s	d
m	m	s	d
s	s	m	s
d	d	s	m

Table 3: Source to target reordering orientations

This table is simpler than the orientation table of (Patil et al., 2015), because of the word order similarity of Hindi (source) and Japanese (target). From this table, we can calculate orientation probabilities of source to target reordering table (fe) as:

$$\mathrm{m}\,(\mathrm{f} \to \mathrm{e}) = \sum_p \{\mathrm{m}(\mathrm{f} \to \mathrm{p})\mathrm{m}(\mathrm{p} \to \mathrm{e}) + \mathrm{s}(\mathrm{f} \to \mathrm{p})\mathrm{s}(\mathrm{p} \to \mathrm{e}) + \mathrm{d}(\mathrm{f} \to \mathrm{p})\mathrm{d}(\mathrm{p} \to \mathrm{e})\}/D$$

$$\mathrm{s}(\mathrm{f} \to \mathrm{e}) = \sum_p \{\mathrm{m}(\mathrm{f} \to \mathrm{p})\mathrm{s}(\mathrm{p} \to \mathrm{e}) + \mathrm{s}(\mathrm{f} \to \mathrm{p})\mathrm{m}(\mathrm{p} \to \mathrm{e}) + \mathrm{d}(\mathrm{f} \to \mathrm{p})\mathrm{s}(\mathrm{p} \to \mathrm{e}) + \mathrm{s}(\mathrm{f} \to \mathrm{p})\mathrm{d}(\mathrm{p} \to \mathrm{e})\}/D$$

$$\mathrm{d}(\mathrm{f} \to \mathrm{e}) = \sum_p \{\mathrm{m}(\mathrm{f} \to \mathrm{p})\mathrm{d}(\mathrm{p} \to \mathrm{e}) + \mathrm{d}(\mathrm{f} \to \mathrm{p})\mathrm{m}(\mathrm{p} \to \mathrm{e})\}/\mathrm{D}$$

where $\mathrm{m}(\mathrm{f} \to \mathrm{e}), \mathrm{s}(\mathrm{f} \to \mathrm{e})$ and $\mathrm{d}(\mathrm{f} \to \mathrm{e})$ are monotone, swap and discontinuous probabilities from source (f) to target (e) and *D* is a normalizing parameter such that $\mathrm{m}(\mathrm{f} \to \mathrm{e}) + \mathrm{s}(\mathrm{f} \to \mathrm{e}) + \mathrm{d}(\mathrm{f} \to \mathrm{e}) = 1$. Inverse probabilities: $\mathrm{m}(\mathrm{e} \to \mathrm{f}), \mathrm{s}(\mathrm{e} \to \mathrm{f})$ and $\mathrm{d}(\mathrm{e} \to \mathrm{f})$ are similarly calculated. We merge this pivoted reordering table and the direct reordering table trained by the Hindi-Japanese parallel corpus (see section 2.5). The merging method is similar to the merging method of the phrase tables described above.

4 Experimental results

4.1 Results of iterative reordering

We conduct iterative reordering consisting of reranking of k-best reordered sentences and alignment loop described in section 2.2 for en-ja, HINDENen-hi, .zh-ja and JPCzh-ja tasks.

Changes of the average of Kendall's tau of the alignment for JPCzh-ja task are shown in Figure 1. Kendall's tau increases by the iteration but an amount of increase is small.

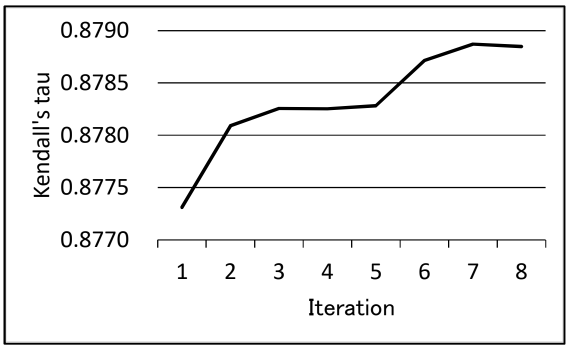

Figure 1: Change of average of Kendall's tau for JPCzh-ja task

For all tasks using the reordering, the number of iterations and average values of Kendall's tau which are obtained by the last iteration are listed in Table 4. The Kendall's tau of en-ja and HINDENeh-hi tasks are rather low than that of zh-ja and JPCzh-ja tasks.

Task	Iteration	Kendall's tau
en-ja	4	0.7655
zh-ja	4	0.9083
JPCzh-ja	8	0.8788
HINDENen-hi	4	0.8398

Table 4: Number of iterations and average values of Kendall's tau for the tasks

4.2 Results of system combination

For JPCzh-ja task, we combine three systems: word based SMT, character based SMT and RBMT plus SPE. We compare translation accuracy using devtest set by BLEU and RIBES. Table 5 shows these scores of several combinations of the systems. We can see our system combination is effective.

No.	System	BLEU	RIBES
1	word based SMT	42.07	82.91
2	char based SMT	41.82	83.03
3	RBMT + SPE	41.61	82.42
4	to combine 1 and 2	42.13	83.13
5	to combine 1, 2 and 3	42.42	83.16

Table 5: Effect of system combination in JPCzh-ja task

4.3 Translation evaluation results

We submitted 12 systems' outputs to the evaluation site. The system descriptions are summarized in Table 6. Official evaluation results of our systems by the organizer (Nakazawa et al., 2016) are listed in Table 7. Evaluation results of the top ranked system are also listed in the table for the comparison.

For JPCzh-ja task, adding RBMT plus SPE increase BLEU score but decrease RIBES, AMFM and HUMAN scores.

For JPCko-ja task, deleting parentheses surrounding a number in Korean side increase three automatic scores. Oppositely, adding parentheses surrounding a number in Japanese side increase HUMAN score.

For HINDENhi-ja task, table level pivoting has higher BLEU and HUMAN score and lower RIBES and AMFM scores than sentence level pivoting. Comparing system 2 and 3 of HINDENhi-ja task, reordering of pivot language (English) is effective with three automatic scores. Pivoting method (system 1, 2 and 3) substantially increases automatic scores compared with the direct method (system 4). However, evaluation scores of this task are largely low compared with the scores of other tasks.

Task	System No.	Word-based PBSMT	Character-based PBSMT	RBMT+SPE	Reordering	Sentence level pivoting	Table level pivoting	Parenthes handling
en-ja	1	✔			✔			
zh-ja	1	✔	✔		✔			
JPCzh-ja	1	✔	✔	✔	✔			
	2	✔	✔		✔			
JPCko-ja	1	✔	✔					del
	2	✔	✔					add & del
	3	✔	✔					add
HINDENen-hi	1	✔			✔			
HINDENhi-ja	1	✔			✔		✔	
	2	✔			✔	✔		
	3	✔				✔		
	4	✔						

Table 6: System descriptions of the tasks

Three systems are evaluated by the JPO adequacy. The system for JPCko-ja task has high JPO adequacy score and systems for HINDENen-hi and HINDENhi-ja tasks have low JPO adequacy score. The former system can translate more than 90% of sentences with 4 or 5 JPO adequacy score, however, the latter systems can only translate 10 or 20% of sentences with 4 or 5 JPO adequacy score (see Figure 2).

Task	System No.	BLEU	RIBES	AMFM	HUMAN	HUMAN (top rank)	JPO adq.	JPO adq. (top rank)
en-ja	1	31.32	0.7599	0.7467	39.000	55.250	---	4.02
zh-ja	1	39.75	0.8437	0.7695	32.500	63.750	---	3.94
JPCzh-ja	1	41.05	0.8270	0.7350	35.500	46.500	---	3.44
	2	40.95	0.8280	0.7451	39.000		---	
JPCko-ja	1	71.51	0.9447	0.8664	-3.000	21.750	---	4.62
	2	68.78	0.9411	0.8517	---		---	
	3	62.33	0.9271	0.8180	21.750		4.56	
HINDENen-hi	1	11.75	0.6719	0.6508	0.000	57.250	2.48	2.55
HINDENhi-ja	1	7.81	0.5793	0.4681	13.750	39.750	2.00	2.13
	2	7.66	0.5860	0.4731	10.000		---	
	3	7.47	0.5823	0.4549	---		---	
	4	2.36	0.4402	0.3628	---		---	

Table 7: Evaluation results (Segmenter for ja is JUMAN)

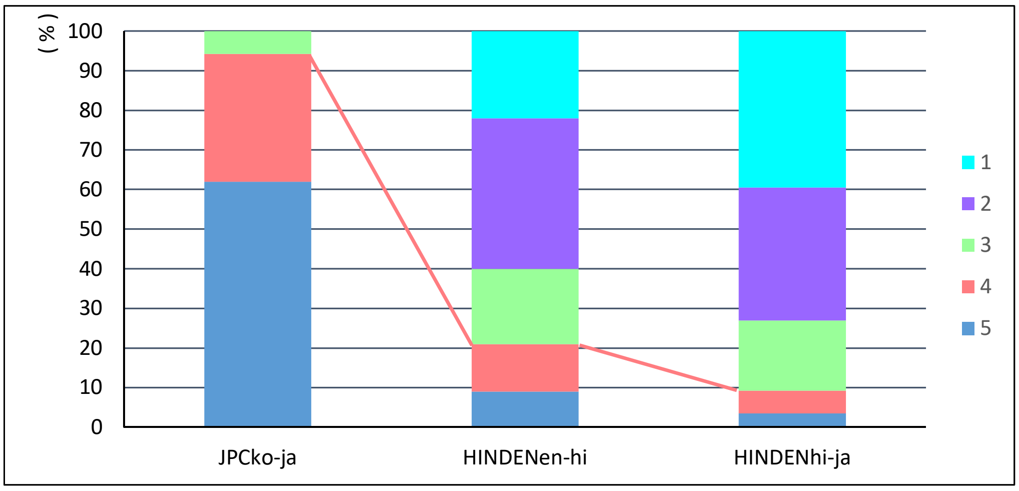

Figure 2: Evaluation results by the JPO adequacy

5 Conclusion

System descriptions, experimental settings and experimental results of EHR group were described. We participate in the 6 tasks and submitted 12 systems' outputs. We can recognize our translation techniques are effective. They are iterative reordering, system combination and pivoting with reordering.

Several remaining issues are as follows. To improve parsing accuracy such that reordering accuracy be higher. To improve English-Hindi and Hindi-Japanese translation accuracy that are largely lower than that of other language pairs. To challenge machine translations for other Asian languages such as Indonesian, Thai, Vietnamese, Mongolian and so on.

Reference

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. WIT^3: Web Inventory of Transcribed and Translated Talks. *Proceedings of the 16th EAMT Conference*, pages 261-168.

Pi-Chuan Chang, Michel Galley and Chris Manning. 2008. Optimizing Chinese Word Segmentation for Machine Translation Performance. *Proceedings of the Third Workshop on Statistical Machine Translation,* pages 224-232.

Trevor Cohn and Mirella Lapata. 2007. Machine Translation by Triangulation: Making Effective Use of Multi-Parallel Corpora. *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 728–735.

Terumasa Ehara. 2007. Rule Based Machine Translation Combined with Statistical Post Editor for Japanese to English Patent Translation. *Proceedings of Machine Translation Summit XI, Workshop on Patent Translation*, pages 13-18.

Terumasa Ehara. 2015. System Combination of RBMT plus SPE and Preordering plus SMT. *Proceedings of the 2nd Workshop on Asian Translation (WAT2015)*, pages 29–34.

Qin Gao and Stephan Vogel. 2008. Parallel Implementations of Word Alignment Tool. *Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, pages 49–57.

Isao Goto, Ka Po Chow, Bin Lu, Eiichiro Sumita and Benjamin K. Tsou. 2013. Overview of the Patent Machine Translation Task at the NTCIR-10 Workshop. *Proceedings of the 10th NTCIR Conference*, pages 260-286.

Hideki Isozaki, Katsuhito Sudoh, Hajime Tsukada, and Kevin Duh. 2010. Head Finalization: A Simple Reordering Rule for SOV Languages. *Proceedings of the Joint 5th Workshop on Statistical Machine Translation and MetricsMATR*, pages 244–251.

Dan Klein and Christopher D. Manning. 2003. Accurate Unlexicalized Parsing. *Proceedings of the 41st Meeting of the Association for Computational Linguistics,* pages 423-430.

Philipp Koehn, Franz J. Och and Daniel Marcu. 2003. Statistical Phrase-Based Translation. *Proceedings of HLT-NAACL 2003*, pages 48-54.

Anoop Kunchukuttan, Abhijit Mishra, Rajen Chatterjee,Ritesh Shah, and Pushpak Bhattacharyya. 2014. Sata-Anuvadak: Tackling Multiway Translation of Indian Languages. *Proceedings of the Ninth International Conference on Language Resources and Evaluation Conference.*

Sadao Kurohashi, Toshihisa Nakamura, Yuji Matsumoto and Makoto Nagao. 1994. Improvements of Japanese morphological analyzer JUMAN. *Proceedings of The International Workshop on Sharable Natural Language Resources,* pages 22-28.

Hyoung-Gyu Lee, Jaesong Lee, Jun-Seok Kim, and Chang-Ki Lee. 2015. NAVER Machine Translation System for WAT 2015. *Proceedings of the 2nd Workshop on Asian Translation (WAT2015)*, pages 69–73.

Roger Levy and Christopher D. Manning. 2003. Is it harder to parse Chinese, or the Chinese Treebank? *ACL 2003,* pages 439-446.

Evgeny Matusov, Nicola Ueffing and Hermann Ney. 2006. Computing Consensus Translation from Multiple Machine Translation Systems Using Enhanced Hypotheses Alignment. *Proceedings of EACL*, pages 33-40.

Toshiaki Nakazawa, Hideya Mino, Chenchen Ding, Isao Goto, Graham Neubig, Sadao Kurohashi and Eiichiro Sumita. 2016. Overview of the 3rd Workshop on Asian Translation. *Proceedings of the 3rd Workshop on Asian Translation (WAT2016).*

Franz Josef Och, Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics,* volume 29, number 1, pages 19-51.

Deepak Patil, Harshad Chavan and Pushpak Bhattacharyya. 2015. Triangulation of Reordering Tables: An Advancement Over Phrase Table Triangulation in Pivot-Based SMT. *Proceedings of the Twelfth International Conference on Natural Language Processing.*

Slav Petrov, Leon Barrett, Romain Thibaux and Dan Klein. 2006. Learning Accurate, Compact, and Interpretable Tree Annotation. *Proceedings of COLING-ACL 2006,* pages 433-440.

Masao Utiyama and Hitoshi Isahara. 2007. A Comparison of Pivot Methods for Phrase-based Statistical Machine Translation. *Proceedings of NAACL HLT 2007*, pages 484–491.

Hua Wu and Haifeng Wang. 2007. Pivot Language Approach for Phrase-Based Statistical Machine Translation. *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 856–863.

Peng Xu, Jaeho Kang, Michael Ringgaard, and Franz Och. 2009. Using a Dependency Parser to Improve SMT for Subject-Object-Verb Languages. *Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL,* pages 245–253.