Forest-to-String SMT for Asian Language Translation: NAIST at WAT 2014

Graham Neubig
Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama-cho, Ikoma-shi, Nara, Japan
neubig@is.naist.jp

Abstract

This paper describes the Nara Institute of Science and Technology's (NAIST) submission to the 2014 Workshop on Asian Translation's four translation tasks. All systems are based on forest-to-string (F2S) translation, in which the input sentence is first parsed using a syntactic parser, then a forest of possible syntactic analyses is translated into the target language. In addition to the baseline F2S system, we add rescoring using a recurrent neural network language model (RNNLM), which allows for more fluent output. The resulting system achieved the highest results in both automatic and manual evaluation for all four of the language pairs targeted by the workshop.

1 Introduction

The Workshop on Asian Translation (WAT) 2014 (Nakazawa et al., 2014) included a translation task over four language pairs, all involving translating Japanese (ja), a language with SOV word order, to/from English (en) or Chinese (zh), languages with SVO word order. Because of this, it can be expected that one of the major challenges facing translation systems in this task is the proper reordering of the words between the source and target languages.

One promising way to tackle the reordering problem is through the use of tree-to-string (T2S) translation, a translation formalism where the source sentence is first parsed using a syntactic parser, then sub-structures of the parse tree are translated into target-side strings (Liu et al., 2006). Mi et al. (2008) have also demonstrated that forest-to-string (F2S) translation allows for more robust use of source-side syntax by not considering a 1-best parse tree, but a myriad of parse candidates stored efficiently in a packed-forest data structure. In our previous work (Neubig and Duh, 2014), we have shown that F2S translation is effective for en-ja and ja-en translation, and can outperform alternative methods such as pre- or post-ordering. Thus, in our WAT submission, we choose this formalism, and specifically it's implementation in the open-source Travatar decoder[1] (Neubig, 2013) as the base of our system.

Another promising development over the past couple years is the use of continuous-space representations of language combined with neural-network-based probabilistic models. These have been incorporated into translation as either language models (LMs) (Vaswani et al., 2013) or translation models (TMs) (Le et al., 2012), allowing for large increases in translation accuracy. In our submission, we incorporate this continuous-space representation by training a recurrent neural network language model (RNNLM; Mikolov et al. (2010)) and using its scores as a feature in *n*-best hypothesis rescoring.

We also made a few small improvements to our ja-en system, mainly in an attempt to reduce the number of unknown words. Specifically, we perform compound splitting (Koehn and Knight, 2003) of unknown words to help reduce the effects of under-segmentation, perform one small word substitution to regularize for the peculiarities of the development/test data, and add large external dictionaries.

As a result of the incorporation of F2S translation and RNNLMs, we see a large gain in accuracy over a baseline phrase-based machine translation model. Specifically, we see a gain in BLEU of 8.21 for en-ja, 5.44 for ja-en, 4.71 for zh-ja, and 2.47 for ja-zh. In addition, according to the official automatic evaluation, our system outperformed all other submitted systems in all tracks. Scripts to

[1] http://phontron.com/travatar

Proceedings of the 1st Workshop on Asian Translation (WAT2014), pages 20–25,
Tokyo, Japan, 4th October 2014.

largely reproduce our experiments will be released open source.[2]

2 Data and Data Processing

2.1 Data Used

For the majority of our systems, we simply used the ASPEC corpus provided by the WAT task. For the zh-ja and ja-zh systems, we used all of the data, amounting to 672k sentences. For the en-ja and ja-en systems, we used all of the data for training the language models, but because the ja-en translation data was automatically aligned and low-confidence sentences were often noisy, we only used the first 2 million sentences of the training data, discarding the rest.

In addition to this official data, for the ja-en pair we submitted one system that used additional dictionaries to reduce the number of unknown words. Specifically, we used the EDICT[3], and Eijiro[4] dictionaries, as well as the Japanese-English links between Wikipedia pages. There are a number of ways to incorporate these dictionaries, but in the submitted system, we simply added a rule to the translation table for all unknown words that existed in the dictionary.

2.2 Tokenization and Preprocessing

For English, Japanese, and Chinese, tokenization was performed using the Stanford Parser (Klein and Manning, 2003), the KyTea toolkit (Neubig et al., 2011), and the Stanford Segmenter (Tseng et al., 2005) respectively. We also performed case normalization for English, by changing the first word in English sentences to its most common capitalization before training models and translation, and capitalizing the first letter of the sentence after translation.[5] For zh-ja translation, in order to make unknown words more comprehensible, we converted simplified Chinese characters to their Japanese equivalents, and vice-versa for ja-zh translation (using the `Kanconvit.pm` Perl script).

In addition to our standard tokenization, for Japanese, while KyTea is on average more robust to unknown words than other standard alternative word segmenters for Japanese, it also has a greater tendency to under-segment words, which can be detrimental for machine translation. As a quick fix to this problem, we re-segmented all words that appear in the dev or test set but not the training set using the compound segmentation method of (Koehn and Knight, 2003), which splits words into two, resolving ambiguities such that the newly split words have the highest unigram probability.

Finally, in a preliminary analysis of our ja-en system using the error analysis method of Akabe et al. (2014),[6] we discovered a peculiarity in the development data: prolific use of the word “標題” (which can be translated into “the mentioned,” or “the XX in the title”). This word appeared prolifically in the dev set (as well as devtest and test), but not once in the training corpus. In order to solve this problem, we normalized “標題” into the lexically different but semantically largely equivalent “表題,” which appeared many times in the training corpus.

2.3 Syntactic Parsing

As we are performing translation using syntactic parsing, it is essential that we have an accurate syntactic parser. Based on the experiments presented in Neubig and Duh (2014) we opt to use the Egret parser,[7] which implements the latent variable parsing model of (Petrov et al., 2006).

For the parsing models in English and Chinese, we use models trained on the English and Chinese Penn Treebanks respectively (Marcus et al., 1993; Xue et al., 2005). For the Japanese model, we train our own model on the Japanese Word Dependency Treebank (Mori et al., 2014). As this is a dependency treebank, we use head rules contained the Travatar toolkit to transform the dependency trees into phrase structure trees.[8]

For training, we simply use 1-best parses, but at test time we use a forest of parse trees, specifically using forests with all tree edges that exist in at least one of the 100-best parses.

3 Model Training

3.1 Alignment

For T2S translation, it is necessary to have an accurate word alignment model, which allows for the extraction of more rules that match the parse tree, and the estimation of more accurate reordering probabilities (Neubig and Duh, 2014).

[2] `http://phontron.com/project/wat2014`

[3] `http://www.edrdg.org/jmdict/edict.html`

[4] `http://www.eijiro.jp`

[5] This is often referred to as “truecasing.”

[6] In fact we slightly modified the method to use the translation reference and a smoothed naive Bayes classifier.

[7] `https://github.com/neubig/egret`

[8] `ja-depadjust.pl` and `ja-dep2cfg.pl`

Thus, for en-ja and ja-en translation, we use Nile[9] (Riesa and Marcu, 2010), a supervised syntax-based aligner that can improve alignment accuracy by incorporating information about parse trees and learn from manually created alignments. For our manual alignments, we use the alignments provided by the Kyoto Free Translation Task (Neubig, 2011). Unfortunately, for zh-ja and ja-zh translation, we do not have any hand-aligned data available, so we use the GIZA++ unsupervised aligner (Och and Ney, 2003).

3.2 Translation Model Training

For training our translation model, we extract a synchronous tree substitution grammar (STSG) according to the method of Galley et al. (2006). We used composed rules including up to 5 minimal rules, and attached null-aligned words to the highest possible point in the parse tree. For the translation model features, we used a standard set of 5 features including forward and backward translation probabilities, forward and backward lexical probabilities, and the phrase penalty. When calculating the translation probabilities, we first applied Kneser-Ney smoothing to the phrases counts (Kneser and Ney, 1995).

3.3 Language Model Training

For all systems, we trained a 6-gram language model smoothed with modified Kneser-Ney smoothing using KenLM (Heafield et al., 2013). In addition, because we had two different data sets containing Japanese data (en-ja and zh-ja), we trained two separate language models and interpolated them together. We chose different interpolation coefficients for the en-ja and zh-ja tasks by choosing the interpolation coefficients that maximize the likelihood on the development data on each of the respective tasks. This gave us a small but significant improvement in perplexity on the development set.

In addition to the *n*-gram language model, we incorporated a recurrent neural network language model (RNNLM) (Mikolov et al., 2010). This, as mentioned in the introduction, will allow us to incorporate recent advances in continuous-space language modeling, improving robustness to unknown or low-frequency linguistic phenomena. We used the RNNLM toolkit,[10] with 500 hidden layers and 300 classes. Because training RNNLM on large data sets is prohibitively expensive, we used only the first 500,000 sentences from the parallel data to train models for each respective task. As RNNLMs cannot be trivially incorporated into decoding due to their continuous-space state representation, we instead use the RNNLM score as an additional feature in 10,000-best rescoring of the output of the baseline model.

3.4 Parameter Optimization

In order to optimize the parameters of the log-linear model, we use standard minimum error rate training (MERT; Och (2003)). As the two official evaluation measures of the contest are BLEU (Papineni et al., 2002) and RIBES (Isozaki et al., 2010), we submitted two systems, one optimized for BLEU, and one optimized for BLEU+RIBES. We also attempted optimizing for RIBES only, which did result in higher RIBES scores, but also extremely short translations and extremely low scores, so we decided against submitting this system.

4 Issues for Context-aware Machine Translation

We did not make any particular attempt to consider super-sentential context in our system. Sub-sentential context is considered to a lesser extent by the *n*-gram LM, and to a greater extent by the RNNLM, the theoretically infinite history of which could potentially capture syntactic or semantic agreement of words that are beyond the scope of traditional *n*-grams.

5 Experimental Results

In Table 1 we show the results for our systems with and without the RNNLM, and tuning with BLEU or BLEU+RIBES. In addition, we show the results for a PBMT system trained using the Moses toolkit (Koehn et al., 2007), with the same data as the F2S system and the default settings except for a reordering limit of 18, which gave better results on all language pairs than the default of 6.

From this table we can first see that the F2S translation greatly outperforms PBMT. The trend is more prominent in the translation to or from English, a result of the fact that the amount of reordering is greater between English and Japanese than between Chinese and Japanese. In addition, we can see that the gain over PBMT is smaller

[9] `https://code.google.com/p/nile/`

[10] `http://rnnlm.org`

			en-ja		ja-en		zh-ja		ja-zh	
System	RNN	Tune	B	R	B	R	B	R	B	R
PBMT	No	B	29.00	68.99	19.41	64.71	35.57	76.78	27.07	79.80
F2S	No	B	36.50	79.69	23.76	71.79	39.82	82.42	29.27	81.26
		B+R	36.24	79.95	24.02	71.77	39.22	83.10	28.83	82.58
	Yes	B	**37.21**	80.21	24.72	72.39	**40.61**	83.18	**29.78**	81.66
		B+R	**36.94**	**80.78**	**25.12**	**72.64**	40.01	**83.52**	29.03	**82.82**
F2S+Dict	Yes	B	-		**25.27**	**72.56**	-		-	

Table 1: Overall BLEU and RIBES results for a baseline Moses, and five of our systems without and with the RNNLM rescoring, tuning for BLEU or BLEU+RIBES, and with/without dictionaries. Bold indicates systems not statistically different from the best system according to bootstrap resampling (Koehn, 2004).

in pairs where the source is Japanese. We account this to the fact that the syntactic parsing accuracy is lower, partly due to the fact that the training data for the parser is smaller (approximately 6,000 sentences), and partly due to the fact that we have done very little grammar engineering for Japanese, in contrast to the more carefully thought-out phrase structure of the English and Chinese treebanks.

Next, taking a look at the results with RNNLM, we can see that adding RNNLM helps across all language pairs on the order of 0.7-1.0 BLEU points, with slightly smaller gains for RIBES. These consistent gains are in concert with previous results, adding further evidence to the observation that continuous space language models are beneficial for translation.

We can also see that adding RIBES to the evaluation function used in parameter optimization leads to a significant increase in RIBES across all data sets. On the other hand, it also leads to a decrease in BLEU for the ja-zh and zh-ja data sets, although no significant decrease is observed in the ja-en and en-ja data sets.

It should also be noted that the systems tuned for BLEU+RIBES tend to result in significantly shorter translation outputs than other systems. Table 2 shows the average length of sentences for each of the systems (using RNNLM in all cases). From this, we can see that for all language pairs except ja-en, BLEU-tuned systems tend to largely match the length of the reference hypotheses, while the BLEU+RIBES tuned systems are significantly shorter.

Tune	en-ja	ja-en	zh-ja	ja-zh
B	29.86	25.10	37.26	27.55
B+R	28.40	25.05	35.61	25.64
Ref.	29.73	24.39	37.51	27.78

Table 2: The average number of words per sentence according to different tuning objectives, as well as for the reference.

6 Official Results

In this section, we discuss the official results of the evaluation focusing on two aspects: the relationship with human evaluation, and a comparison with other systems. In Table 3 we show the official results for each system, including human evaluation. When reports for multiple segmenters are reported, we display the ones calculated using KyTea. Human evaluation is calculated according to pairwise comparison with the baseline according to the official task description (Nakazawa et al., 2014), where 100 indicates that the proposed system exceeds the baseline in all judgements, and 0 indicates that the system is equivalent with the baseline.

First, we note that the NAIST submissions achieved the highest score in all three evaluation measures across all four language pairs. The competing teams include PBMT, Hiero, tree-to-string, string-to-tree systems prepared by the organizers, as well as systems prepared by the participants (often based on preordering, or statistical post-editing of rule-based MT). This provides further evidence to support our previous observation that F2S translation provides strong results for the language pairs under consideration for the WAT task (Neubig and Duh, 2014).

		en-ja			ja-en			zh-ja			ja-zh		
Dict	Tune	B	R	H	B	R	H	B	R	H	B	R	H
No	B	**37.2**	80.2	**56.3**	23.3	**72.4**	**37.5**	**41.3**	83.5	**50.8**	**30.5**	81.8	**17.8**
	B+R	**37.2**	**80.7**	**51.5**	23.5	**72.4**	-	40.8	**83.8**	38.0	29.8	**83.0**	1.3
Yes	B		-		**23.8**	**72.3**	**40.5**		-			-	
Best B		35.0	79.0	36.0	21.1	69.9	25.0	37.7	82.6	22.5	28.7	81.0	**14.0**
Best R		35.0	79.0	36.0	20.6	70.8	23.3	37.7	82.6	22.5	27.7	81.0	3.8
Best H		34.9	78.6	43.3	20.4	67.8	25.5	37.7	82.6	22.5	28.7	81.0	**14.0**

Table 3: BLEU, RIBES, and HUMAN evaluation according to the official evaluation results. We also show the best competing systems other than ours according to each evaluation metric. Bold indicates systems within 0.3 of the best system for BLEU and RIBES, and systems that do not show a significant decrease from the best system according to Student's t-test for HUMAN ($p < 0.05$).

System	Translation
Source	由于气候变化和能源保障问题，引入了环境污染税和碳税，这对电力产业尤为重要。
Tuned B	気候の変化とエネルギー保障問題，環境汚染税と炭素税を導入し，電力産業に対して極めて重要である。
Tuned B+R	気候変化やエネルギー保障問題，環境汚染税と炭素税を導入し，電力産業が重要である。

Table 4: Examples of zh-ja translations for systems tuned with different objectives.

Next, we take a look at the correlation between automatic evaluation measures and human evaluation scores. Just looking at our systems, we can see that in general BLEU scores are a good indicator of human evaluation, while in many instances systems with higher RIBES scores achieve lower human evaluation. This is in contrary to previous evaluation campaigns including the Japanese language (Goto et al., 2011), and thus a somewhat noteworthy result.

We hypothesize that this is due to the fact, as mentioned in the previous section, that systems tuned for BLEU+RIBES achieve higher RIBES scores, but also produce short hypotheses. Because the hypotheses are short, they have a larger chance of dropping words and missing important information. We show one example of this in Table 4, where the BLEU system received a higher manual evaluation score than the BLEU+RIBES system. In this example, the system tuned with only BLEU produces a longer hypothesis than that of the system tuned with BLEU+RIBES. In particular, focusing on the word "に対して" ("for" in English), the BLEU system includes this word and is able to achieve a translation corresponding to the true meaning of "is important for the electric industry," while the BLEU+RIBES system drops the word, causing a mistaken translation of "the electric industry is important."

7 Conclusion

In this paper we described the NAIST submission to the WAT 2014 translation task. The system was based on forest-to-string statistical machine translation, and achieved the highest translation accuracy on all four language pairs.

While the accuracy was relatively high, there is still significant amounts of work to be done. First, our subjective assessment of the translation results indicated that the parsing accuracy for Japanese is still likely lower than it is for other languages. Examining the use of other, more accurate parsers for Japanese is high on the list of priorities. In addition, the statistical model used in our translation system is a standard one based on standard maximum likelihood estimation and minimum error rate training of a small number of dense features. In future work we hope to improve both the model estimation and parameter tuning processes using more sophisticated models, features, and methods.

Acknowledgments:

This work was supported by JSPS KAKENHI Grant Number 25730136.

References

Koichi Akabe, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2014. Discriminative language models as a tool for machine translation error analysis. In *Proc. COLING*, pages 1124–1132.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proc. ACL*, pages 961–968.

Isao Goto, Bin Lu, Ka Po Chow, Eiichiro Sumita, and Benjamin K. Tsou. 2011. Overview of the patent machine translation task at the NTCIR-9 workshop. In *Proceedings of NTCIR*, volume 9, pages 559–578.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable modified Kneser-Ney language model estimation. In *Proc. ACL*, pages 690–696.

Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010. Automatic evaluation of translation quality for distant language pairs. In *Proc. EMNLP*, pages 944–952.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proc. ACL*, pages 423–430.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for M-gram language modeling. *Proc. ICASSP*.

Philipp Koehn and Kevin Knight. 2003. Empirical methods for compound splitting. In *Proc. EACL*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. ACL*, pages 177–180.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. EMNLP*.

Hai-Son Le, Alexandre Allauzen, and François Yvon. 2012. Continuous space translation models with neural networks. In *Proc. NAACL*, pages 39–48.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proc. ACL*.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn treebank. *Computational linguistics*, 19(2):313–330.

Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *Proc. ACL*, pages 192–199.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proc. 11th InterSpeech*, pages 1045–1048.

Shinsuke Mori, Hideki Ogura, and Tetsuro Sasada. 2014. A Japanese word dependency corpus. In *Proc. LREC*.

Toshiaki Nakazawa, Hideki Mino, Isao Goto, Sadao Kurohashi, and Eiichiro Sumita. 2014. Overview of the 1st Workshop on Asian Translation. In *Proc. WAT*.

Graham Neubig and Kevin Duh. 2014. On the elements of an accurate tree-to-string machine translation system. In *Proc. ACL*.

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. 2011. Pointwise prediction for robust, adaptable Japanese morphological analysis. In *Proc. ACL*, pages 529–533.

Graham Neubig. 2011. The Kyoto free translation task. `http://www.phontron.com/kftt`.

Graham Neubig. 2013. Travatar: A forest-to-string machine translation engine based on tree transducers. In *Proc. ACL Demo Track*, pages 91–96.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. ACL*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. ACL*, pages 311–318.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proc. ACL*, pages 433–440.

Jason Riesa and Daniel Marcu. 2010. Hierarchical search for word alignment. In *Proc. ACL*, pages 157–166.

Huihsin Tseng, Pichuan Chang, Galen Andrew, Daniel Jurafsky, and Christopher Manning. 2005. A conditional random field word segmenter for sighan bakeoff 2005. In *Proc. SIGHAN*.

Ashish Vaswani, Yinggong Zhao, Victoria Fossum, and David Chiang. 2013. Decoding with large-scale neural language models improves translation. In *Proc. EMNLP*, pages 1387–1392.

Naiwen Xue, Fei Xia, Fu-Dong Chiou, and Martha Palmer. 2005. The Penn Chinese treebank: Phrase structure annotation of a large corpus. *Natural language engineering*, 11(02):207–238.