The NUS Statistical Machine Translation System for IWSLT 2009

Preslav Nakov, Chang Liu, Wei Lu, Hwee Tou Ng

Department of Computer Science
National University of Singapore
13 Computing Drive, Singapore 117417
{nakov,liuchan1,luwei,nght}@comp.nus.edu.sg

Abstract

We describe the system developed by the team of the National University of Singapore for the Chinese-English BTEC task of the IWSLT 2009 evaluation campaign. We adopted a state-of-the-art phrase-based statistical machine translation approach and focused on experiments with different Chinese word segmentation standards. In our official submission, we trained a separate system for each segmenter and we combined the outputs in a subsequent re-ranking step. Given the small size of the training data, we further re-trained the system on the development data after tuning. The evaluation results show that both strategies yield sizeable and consistent improvements in translation quality.

1. Introduction

This is the first year that the National University of Singapore (NUS) participated in the evaluation campaign of the International Workshop on Spoken Language Translation (IWSLT). We submitted a run for the Chinese-English BTEC task[1], where we were ranked second out of twelve participating teams, based on the average of the normalized scores of ten automatic evaluation metrics. We adopted a phrase-based statistical machine translation (SMT) approach, and we investigated the effectiveness of different Chinese word segmentation standards. Using a maximum entropy model [1] and various data sources, we trained six different Chinese word segmenters. Each segmenter was then used to preprocess the Chinese side of the training/development/testing bi-texts, from which a separate phrase-based SMT system was built. Some of the resulting six systems yielded substantial translation performance gains as compared to a system that used the default segmentation provided by the organizers. Finally, we combined the output of all seven systems.

The rest of this paper is organized as follows: Section 2 introduces the phrase-based SMT model, Section 3 presents our pre-processing techniques, Section 4 describes our re-ranking-based system combination approach, Section 5 explains the training methodology, Section 6 gives the evaluation results, which are discussed in Section 7, and Section 8 concludes and suggests possible directions for future work.

[1]Named after the *Basic Travel Expression Corpus* (BTEC).

2. Phrase-Based Machine Translation

We use the phrase-based statistical machine translation model in all our experiments. A brief description follows.

Statistical machine translation is based on the *noisy channel model*. Given a foreign-language (*e.g.,* Chinese) input sentence **f**, it looks for its most likely English translation **e**:

$$\mathbf{e}^* = \operatorname*{argmax}_{\mathbf{e}} \Pr(\mathbf{e}|\mathbf{f}) = \operatorname*{argmax}_{\mathbf{e}} \Big(\Pr(\mathbf{f}|\mathbf{e}) \times \Pr(\mathbf{e}) \Big)$$

In the above equation, $\Pr(\mathbf{e})$ is the target *language model*; it is trained on monolingual text, *e.g.*, the English side of the training bi-text. The term $\Pr(\mathbf{f}|\mathbf{e})$ is the *translation model*. In phrase-based SMT, it expresses a generative process: (1) the input sentence **f** is segmented into a sequence of phrases (all segmentations are considered equally likely), (2) each phrase is translated into the target language in isolation, and (3) some of the target phrases are reordered. The phrase translation pairs and their probabilities are acquired from a parallel sentence-aligned bi-text, and are typically induced from word-level alignments using various heuristics.

In phrase-based SMT, the noisy channel model is typically extended to a more general log-linear model, where several additional terms are introduced. For each pair of phrases used, there are four terms: forward and backward phrase translation probabilities, and forward and backward lexicalized phrase translation probabilities. There is also phrase penalty, which encourages the model to use fewer, and thus longer, phrases. Word penalty on the target language side is also included, which controls the overall length of the English output. Finally, the phrase reordering is controlled by a distance-based distortion model.

Under this log-linear model, the most likely English translation **e** is found as follows:

$$\begin{aligned}
\mathbf{e}^* &= \operatorname*{argmax}_{\mathbf{e}} \Pr(\mathbf{e}|\mathbf{f}) \approx \operatorname*{argmax}_{\mathbf{e,s}} \Pr(\mathbf{e,s}|\mathbf{f}) \\
&= \operatorname*{argmax}_{\mathbf{e,s}} \Big(\Pr(\mathbf{e})^{\lambda_1} \times \prod_{i=1}^{|\mathbf{s}|} \Pr(\bar{f}_i|\bar{e}_i)^{\lambda_2} \times \Pr(\bar{e}_i|\bar{f}_i)^{\lambda_3} \\
&\times \Pr_w(\bar{f}_i|\bar{e}_i)^{\lambda_4} \times \Pr_w(\bar{e}_i|\bar{f}_i)^{\lambda_5} \times d(start_i, end_{i-1})^{\lambda_6} \\
&\times \exp(|\bar{e}_i|)^{\lambda_7} \times \exp(-1)^{\lambda_8} \Big)
\end{aligned}$$

In the above equation, **s** is a segmentation of **f** into phrases. The symbols $\bar{e_i}$ and $\bar{f_i}$ denote an English-foreign translation phrase pair used in the translation, and $|\mathbf{s}|$ is the number of such pairs under the current segmentation. The terms $\Pr(\bar{e_i}|\bar{f_i})$ and $\Pr(\bar{f_i}|\bar{e_i})$ are the phrase-level conditional probabilities, and $\Pr_w(\bar{e_i}|\bar{f_i})$ and $\Pr_w(\bar{f_i}|\bar{e_i})$ are corresponding lexical weights as described in [2]. The distance-based distortion term $d(start_i, end_{i-1})$ gives the cost for relative reordering of the target phrases at position i and $i-1$; more complex distortion models are possible, *e.g.*, lexicalized. The remaining two terms $\exp(|\bar{e_i}|)$ and $\exp(-1)$ are the word penalty and the phrase penalty respectively. The parameters λ_i are typically estimated from a tuning set using *minimum error rate training* (MERT) as described in [3].

More detailed description of phrase-based SMT models can be found in [2] and [4]. In our experiments, we used Moses [5], a popular open-source toolkit.

3. Pre-processing

For the training, development, and testing bi-texts, we performed the following five types of pre-processing:

3.1. ASCII-ization

Originally, the English letters on the Chinese side of the bi-text were encoded as full-width characters; we converted them to ASCII, thus ending up with some ASCII words on the Chinese side of the bi-texts, *e.g.*, *ABC*, *Diaz*, *Opera*, *Watanabe*, *Yamada*. Note that most of these words are part of named entities, which are likely to be preserved during translation. Thus, in order to improve word alignments, we added each such ASCII word as an individual sentence to both the English and the Chinese side of the training bi-text. Multiple copies could also be added in order to place more confidence on this heuristic in the EM word alignment step. In our experiments, exactly two copies were added.

3.2. Sentence Breaking

We observed that the training and the development bi-texts often contained two or more sentences on the same line. In many cases, this happened simultaneously on both the Chinese and the English side of the bi-text, as in the following example:

- 当然 。 它 是 九 十 九 美元 不是 吗 ？
- *Of course . It was ninety-nine dollars , wasn't it ?*

In such cases, each sentence can be translated individually and the output concatenated afterwards. We found no sentence-level reordering in the training corpus.

The potential advantage of breaking up these sentences is that the decoder will only need to deal with shorter inputs, which are much easier to translate. We thus split the sentences in the training bi-text whenever possible, *i.e.*, when splitting yielded the same number of sentences on the Chinese and English side of the bi-text, thus increasing the number of lines from 19,972 to 23,110, or by 16%. We further had to split the sentences from the development bi-text (and ultimately, from the Chinese test data); otherwise, the performance of our system was negatively affected.

3.3. Capitalization

We removed all capitalization from both the English and the Chinese side of the training and the development bi-texts, as well as from the test data. In order to add proper casing to the final output, we used the re-caser included in Moses, which we trained on the English side of the training bi-text.

3.4. English Re-tokenization

Although English words are typically written naturally in a segmented form, some ambiguities regarding tokenization still remain. In particular, the English side of the bi-text contained many tokens with internal apostrophes, which could cause data sparsity issues, *e.g.*, *it's*, *I'll*, *weren't*. We thus re-tokenized the English side of the training and the development bi-texts by inserting a space before the apostrophe where appropriate.

3.5. Number Translation

Because of the spoken nature of the data and because of the domain, numbers were abundant in both the training and the development bi-texts. Hence, translating them correctly was a must for a good system. Unfortunately, number translation is not so amenable to SMT methods, and we had to design a specialized system that combines a maximum entropy classifier with hand-crafted rules.

First, we manually inspected part of the English side of the training bi-text, and we identified the following five common ways to express a number:

1. Integer, *e.g.*, *size twenty-two*.
2. Digits, *e.g.*, *flight number one one three*.
3. Series, *e.g.*, *March nineteen ninety-nine*.
4. Ordinal, *e.g.*, *July* twenty-seventh.
5. Others: all other cases, *e.g.*, 一 ('*one*') translated as *a/an* in English.

We trained a log-linear (maximum entropy) classifier in order to determine which of the above five categories should apply for each number instance in the Chinese text. The model used the following seven features: (1) number of digits in the numerical form; (2) the numerical value of the number; (3) the preceding word; (4) the preceding character; (5) the following word; (6) the following two words; and (7) conjunction of features (3) and (5).

We selected these features in a greedy process from a pool containing many others, by repeatedly adding the feature contributing the biggest improvement. The process continued until no new feature was able to enhance the performance any further. The weights of these features were then optimized on the supplied training bi-text.

For the Chinese side of the bi-text, including the training, the development, and the testing sections, we translated each Chinese-side number into English as follows: we first used the classifier to choose one of the above five number translation categories, and then we applied the corresponding handcrafted translation rules; no translation was performed if the category was *Others*.

Finally, we added all English words that were, or could be, part of a translated English number, *e.g.*, *ten*, *twenty-three*, *and*, *etc.*, to both sides of the training bi-text twice, as we did for ASCII-ization.

4. Word Segmentation and Re-ranking

In this section, we describe our experiments with different Chinese word segmentations and how we combine them into a single system using re-ranking.

4.1. Chinese Word Segmentation

Chinese word segmentation (CWS) has been shown conclusively as an essential step in machine translation, at least as far as current phrase-based SMT methods are concerned [6]. However, CWS is complicated by the fact that a *word* is not a well-defined concept in Chinese, where characters, words, and phrases form a blurry continuum. As a consequence, multiple standards exist for the CWS task. For example, two of the SIGHAN CWS bakeoffs offered data according to five different standards: Academia Sinica (AS), UPenn Chinese Treebank (CTB), City University of Hong Kong (CITYU), Peking University (PKU), and Microsoft Research (MSR).

It has been hypothesized that different standards may be best suited for different tasks, and the effect of CWS on machine translation has been studied in several recent works, including lattice-based methods [7, 8], segmentation granularity tuning [9], and CWS standards interpolation [10].

In our experiments, we adopted a very pragmatic approach: we prepared seven candidate systems, each using a different CWS. The final translation output was then selected in a system combination step. Five of the seven segmentations (AS, CTB, CITYU, PKU, and MSR) were generated by an in-house segmenter described in [1], which was ranked first in the AS, CITYU, and PKU open tasks and second in the MSR open task in the SIGHAN 2005 bakeoff (CTB was absent in that year). The two remaining systems were the default segmentation provided by the IWSLT organizers, and ICTCLAS-generated [11] segmentation respectively. Although ICTCLAS was also based on the PKU standard, the output seemed different enough from our PKU segmenter to be included as a separate candidate.

4.2. System Combination by Re-ranking

At different stages of our system development, different segmenters had an edge for different parameter settings, including ICTCLAS, AS, MSR, and PKU. However, in our final configuration, the best overall performance came from ICTCLAS and PKU, but not for all three testing datasets that we used – those for IWSLT05, IWSLT07, and IWSLT08. Surprisingly, AS performed worst overall.

Given the instability of the performance of the Chinese word segmenters for different parameter settings, we chose not to rely on a single segmenter, but to train a separate system for each of the above-mentioned seven segmenters and to combine their output in a subsequent system combination step. Unlike in typical system combination setup, our task was greatly simplified for two reasons: (1) due to time constraints, we did not attempt to synthesize new translations from the existing candidates, but rather aimed to select one of those candidates, and (2) all seven systems used the same parameter settings and the same SMT toolkit, and consequently had comparable scores and probabilities. Thus, we used the scores reported by the SMT system, rather than having to rely on system-independent features as is typically done. Our system combination module was trained and used as follows:

1. We ran all seven candidate systems on the development data. The output included the English translation and thirteen associated scores from the SMT toolkit, which we used as features:
   (a) five (5) from the distortion model;
   (b) two (2) from the phrase translation model;
   (c) two (2) from the lexical translation model;
   (d) one (1) for the language model;
   (e) one (1) for the phrase penalty;
   (f) one (1) for the word penalty; and
   (g) one (1) for the final overall translation score (as calculated by Moses from all individual scores above and the MERT-tuned parameters).
2. A global fourteenth feature *repetition count* was added, which gives the number of systems that generated the target translation.
3. The oracle BLEU score for each translation candidate was calculated. Unfortunately, since BLEU was designed as a document-level score, it was often zero at the sentence-level in our case. Using bi-gram BLEU scores proved to be a good work-around.
4. A supervised classifier was trained to select the candidate with the highest BLEU score given the above-described fourteen features. A number of methods to train the classifier were evaluated, including MERT, SVM$^{\text{rank}}$, and maximum entropy (MaxEnt); the last was found to perform the best.

An inspection of the MaxEnt model revealed that the *repetition count* had the dominant weight, *i.e.*, our system combination worked mostly as a majority vote.

We also tried to add various forms of length-ratios, but all of them turned out to be ineffective, contrary to the findings of many teams in 2008. This could be due to our candidate systems being very similar, and thus not that different in the length of their outputs, or to the fourteen features already capturing the pertinent information adequately.

5. Training Methodology

Below we explain how we tuned various parameters of the phrase-based SMT system. We further describe a novel re-training technique yielding sizeable improvements in BLEU.

5.1. Parameter Tuning

The Moses phrase-based SMT toolkit has a large number of options. While it comes with very sensible defaults, we found experimentally that varying some of them had a significant impact on the translation quality.

Table 1 shows some non-standard settings used in our submission. Note that, for word alignments, we used the Berkeley Aligner[2] [12] in unsupervised mode, which we found to outperform GIZA++ significantly. We used the default parameters of the aligner, except that we increased the number of iterations to 40.

5.2. Re-training on the Development Dataset

In a typical machine learning setup, the data is split into training, development, and testing datasets, which are then used as follows:

1. The system is trained on the training dataset.
2. The development dataset is used for tuning, *e.g.*, for meta-parameter estimation, for setting configuration options, and for feature selection.
3. The testing dataset is used to assess the performance of the system on unseen data.

Better utilization of the data can be achieved with an extended procedure that uses re-training:

4. The system is re-trained on a concatenation of the training and the testing datasets, using the parameters from step 2 above.
5. The system is then re-tuned on the development dataset.
6. Finally, the system is re-trained on a concatenation of all available data (training, development, and testing), using the parameters from the previous step.

[2] http://code.google.com/p/berkeleyaligner/

Since the last three steps use more data, we can generally expect improvements in the performance of the overall system.

Given the small size of the training data, which consisted of 19,972 BTEC sentence pairs only, re-training proved to be very helpful in our case. For the development data, we used the CSTAR03 and IWSLT04 datasets, which had a total of 1,006 Chinese and 16,096 English sentences (there were 16 English reference translations for each Chinese sentence). For step 3, we used the IWSLT05, IWSLT07, and IWSLT08 datasets for testing; they had a total of 1,502 Chinese sentences and 19,142 English sentences. If we consider the number of sentence pairs or the size of the English-side, the above-described re-training allowed us to use nearly three times as much training data.

Our setup was further complicated due to the extra system combination step. The full procedure is explained below:

1. We used the training bi-text to build a phrase table and to train an English language model.
2. We used the development dataset to tune the weights of the log-linear model of the phrase-based SMT system using MERT.
3. We concatenated the training and the development datasets; we then re-built the phrase table and re-trained the language model on this new dataset.
4. We repeated the above three steps for each of the seven Chinese word segmenters, thus obtaining seven candidate systems.
5. We performed feature selection for the combination of the seven systems, using three-fold cross-validation on the testing bi-texts: we trained the MaxEnt re-ranking model on IWSLT05+IWSLT07 and tested it on IWSLT08; we then trained on IWSLT05+IWSLT08 and tested on IWSLT07; and, finally, we trained on IWSLT07+IWSLT08 and tested on IWSLT05. We selected features that optimized the average of the BLEU scores for the three folds.
6. We re-trained the MaxEnt model for system combination on a concatenation of all three testing datasets: IWSLT05, IWSLT07, and IWSLT08.
7. We re-built the phrase table and we re-trained the language model on a concatenation of the training and the testing datasets.
8. We used MERT to tune the system feature weights on the development dataset.
9. We re-built the phrase table and we re-trained the English language model on all data combined (training, development, and testing), using the feature weights from step 8.

(Meta-)Parameter	Standard Setting	Our Setting
Language model order	3	5
Language modeling toolkit	SRILM	IRSTLM
Word aligner	GIZA++	Berkeley aligner
Alignment combination heuristic	grow-diag-final	intersection
Phrase reordering model	distance	monotonicity-bidirectional-f
Maximum phrase length	7	8
BLEU reference length used in MERT	shortest	closest
Miscellaneous	–	drop unknown words

Table 1: Some standard settings in Moses and their non-standard counter-parts used in our system.

For the 2009 testing dataset, we used the seven Moses systems obtained in step 9 with the feature weights from step 8. Our final submission was generated by combining these systems using the MaxEnt re-ranking model trained in step 6.

6. Evaluation

As we have seen above, our system is complex and has many settings. While some configuration choices can be explained theoretically, the optimal values of most parameters are to be determined empirically. Below we will try to justify some of the non-standard settings used in our system.

Due to their non-linear nature, it is difficult to isolate the effect of the individual choices. While it is usually more convenient to gradually raise the baseline and to report the improvements following every change, we will adopt a different approach. We will use the *final best* system as the "topline" and we will report the performance drop as each change is reverted. This is more sound since the eventual decision to include or exclude a configuration change is to be made on the sole basis of how it affects the *current best configuration*; the relative improvement with respect to the *baseline* at the time of the introduction of the configuration change is irrelevant, even though the two may be correlated.

All scores reported are obtained using the official evaluation guidelines and the NIST `mteval`[3] script (version 13).

6.1. Pre-processing and Configuration Options

Table 2 shows the individual effects of each of the pre-processing steps described in Section 3 (except for capitalization). The last row contains the BLEU scores for our final topline system evaluated on the three testing datasets as well as an averaged score; each of the previous four rows illustrates the effect of excluding a single step from the pre-processing pipeline. All reported results are for the default segmentation. We can see that the largest improvement is contributed by the number translation module (1.6 BLEU points on average), while sentence breaking has the least average effect (about 0.2 BLEU points). Note, however, that the impact of the individual pre-processing steps varies across the different segmenters.

[3] ftp://jaguar.ncsl.nist.gov/mt/resources/mteval-v13.pl

Table 3 shows the effect of some of the non-standard parameter settings we used with Moses. We can see that replacing GIZA++ with the Berkeley aligner improves the average BLEU score by 0.9 points, while using a lexicalized reordering model yields 1.4 BLEU points of average improvement. However, the highest average positive impact of 1.7 BLEU points is achieved when the unknown words are dropped during decoding.

6.2. Re-training

Table 4 shows the impact of re-training on the seven systems corresponding to the seven different Chinese word segmentations. The *before* re-training rows show the performance of the system on the three testing datasets (and their average) at the end of step 2 as described in the previous section, while the *after* re-training rows show their performance at the end of step 4.

Note that steps 7, 8, and 9 from the previous section add one further re-training step for each of the seven systems, and thus the performance of the individual systems, and ultimately of their combination, is expected to be even higher on the actual testing dataset for 2009. Unfortunately, we could not assess the size of that improvement since we have no access to the reference translations for that dataset at the time of writing.

6.3. System Combination

Table 5 shows the BLEU score for each cross-validation iteration of our combined system as well as the score for the best individual system. In all three iterations, the combined system outperforms the best individual system (which is different for each iteration), showing that our re-ranking-based system combination model is effective in predicting the best candidate out of the seven.

Again, note that in the final submission, we further retrain the system combination model using all data from the IWSLT05, IWSLT07, and IWSLT08 datasets, according to step 6 of our training methodology, as described in the previous section. It is thus reasonable to expect even higher performance on the actual testing dataset for 2009. However, we have no unbiased way of quantifying it since we have no access to the reference translations for that dataset at the time

Excluded Pre-processing Step	IWSLT05	IWSLT07	IWSLT08	Average
ASCII-ization	0.5286	0.3104	0.4438	0.4276
Sentence breaking	0.5260	0.3100	0.4535	0.4298
Number translation	0.5272	0.2941	0.4262	0.4158
English re-tokenization	0.5244	0.3102	0.4439	0.4262
Keep all (i.e., exclude none)	0.5189	0.3264	0.4503	0.4319

Table 2: The effect of excluding each of the four pre-processing steps. Shown are the BLEU v13 scores for three testing datasets as well as their average. All reported results are for the default segmentation.

(Meta-)Parameter	Our Setting	Reverted to	IWSLT05	IWSLT07	IWSLT08	Average
Aligner	Berkeley aligner	GIZA++	0.5246	0.3101	0.4342	0.4230
Reordering model	monotonicity-bidirectional-f	distance	0.5230	0.2983	0.4333	0.4182
Miscellaneous	drop unknown words	–	0.5157	0.3023	0.4278	0.4153
Keep all (i.e., revert nothing)			0.5189	0.3264	0.4503	0.4319

Table 3: The effect of excluding some of the non-standard parameter settings we used with Moses. Shown are the BLEU v13 scores for three testing datasets as well as their average. All reported results are for the default segmentation.

Segmentation	Re-training	IWSLT05	IWSLT07	IWSLT08	Average
Default	before	0.5161	0.2887	0.4241	0.4096
	after	0.5189	0.3264	0.4503	0.4319
ICTCLAS	before	0.5296	0.2923	0.4149	0.4123
	after	0.5394	0.3129	0.4539	0.4354
AS	before	0.5321	0.2901	0.4053	0.4092
	after	0.5272	0.3074	0.4458	0.4268
CITYU	before	0.5390	0.2899	0.4002	0.4097
	after	0.5304	0.3208	0.4301	0.4271
CTB	before	0.5279	0.3012	0.4138	0.4143
	after	0.5319	0.3053	0.4550	0.4307
MSR	before	0.5337	0.2921	0.4214	0.4157
	after	0.5338	0.3217	0.4406	0.4320
PKU	before	0.5317	0.2977	0.4070	0.4120
	after	0.5367	0.3164	0.4501	0.4344

Table 4: Effect of re-training on the seven systems corresponding to the seven different Chinese word segmentations. Shown are the BLEU v13 scores for three testing datasets as well as their average.

Trained on	Tested on	Combination BLEU	Best Individual	BLEU
IWSLT07 + IWSLT08	IWSLT05	0.5457	ICTCLAS	0.5394
IWSLT05 + IWSLT08	IWSLT07	0.3268	Default	0.3264
IWSLT05 + IWSLT07	IWSLT08	0.4656	CTB	0.4550
Average		0.4460	–	0.4403

Table 5: Results for system combination. Shown are the BLEU v13 scores for evaluating on each of the three testing datasets as well as their average. Further shown are the corresponding best individual systems and their scores.

of writing.

7. Discussion

The previous section has shown that the improvements come from several sources, the most notable being the following:

- **Pre-processing:** 1.6 BLEU points from number translation; 0.6 BLEU points from English re-tokenization.
- **Moses tuning:** 1.7 BLEU points from dropping the unknown words; 1.4 BLEU points from the reordering model; 0.9 BLEU points from the Berkeley aligner.
- **Re-training:** approximately 2 BLEU points. This was before the system was re-trained again on all datasets, including the testing ones, which can be expected to boost the performance even further.
- **Segmentation and system combination:** improvement of 1.4 BLEU points over the default segmentation, 1 BLEU point over the single best segmentation, and 0.6 BLEU points over an oracle that picks the best segmentation for any individual test sentence.

We further experimented with hierarchical phrase-based SMT, which was a popular component of many IWSLT system combinations in previous years. However, in our experiments, hierarchical SMT systems performed significantly worse than phrase-based ones. Moreover, combining hierarchical and phrase-based systems greatly complicated our experimental setup, while the performance of the combination did not improve. We thus eventually chose the current setup because of its simplicity, which allowed us to explore other parameter optimizations.

Finally, we tried using word sense disambiguation (WSD) to improve SMT. Using the method described in [13], we were able to achieve approximately 0.5-1.0 BLEU point of absolute improvement. Unfortunately, we could not include the module in our final submission due to logistic issues.

8. Conclusion and Future Work

We have described the NUS system for the Chinese-English BTEC task of the IWSLT 2009 evaluation campaign. In a series of experiments with a state-of-the-art phrase-based SMT model, we have observed that different Chinese word segmentation standards had an edge for different parameter settings. We thus chose not to rely on a single segmenter, but to train a separate system for each of seven segmenters and to combine their outputs in a subsequent system combination step using re-ranking. Given the small size of the training dataset, we further experimented with re-training the system on the development and on the testing datasets. The evaluation results have shown that both strategies yielded sizeable and consistent improvements in translation quality.

In future work, we plan to experiment with lattice-based system combination. Finding a more principled way to combine different word segmentations is another promising research direction that we plan to pursue. Finally, we intend to incorporate WSD in our system combination.

9. Acknowledgments

This research was partially supported by research grants CSIDM-200804 and POD0713875.

10. References

[1] J. K. Low, H. T. Ng, and W. Guo, "A maximum entropy approach to Chinese word segmentation," in *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, 2005.

[2] P. Koehn, F. J. Och, and D. Marcu, "Statistical phrase-based translation," in *Proceedings of NAACL-HLT*, 2003.

[3] F. J. Och, "Minimum error rate training in statistical machine translation," in *Proceedings of ACL*, 2003.

[4] R. Zens, F. Och, and H. Ney, "Phrase-based statistical machine translation," *Annual German Conference on AI (KI 2002), LNAI 2479*, pp. 18–32, 2002.

[5] 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 ACL*, 2007.

[6] J. Xu, R. Zens, and H. Ney, "Do we need Chinese word segmentation for statistical machine translation?" in *Proceedings of the Third SIGHAN Workshop on Chinese Language Processing*, 2004.

[7] J. Xu, E. Matusov, R. Zens, and H. Ney, "Integrated Chinese word segmentation in statistical machine translation," in *Proceedings of IWSLT*, 2005.

[8] C. Dyer, S. Muresan, and P. Resnik, "Generalizing word lattice translation," in *Proceedings of ACL-HLT*, 2008.

[9] P.-C. Chang, M. Galley, and C. D. Manning, "Optimizing Chinese word segmentation for machine translation performance," in *Proceedings of the Third Workshop on Statistical Machine Translation*, 2008.

[10] R. Zhang, K. Yasuda, and E. Sumita, "Chinese word segmentation and statistical machine translation," *ACM Transactions on Speech and Language Processing*, 2008.

[11] H.-P. Zhang, H.-K. Yu, D.-Y. Xiong, and Q. Liu, "HHMM-based Chinese lexical analyzer ICTCLAS," in *Proceedings of the Second SIGHAN Workshop on Chinese Language Processing*, 2003.

[12] A. Haghighi, J. Blitzer, J. DeNero, and D. Klein, "Better word alignments with supervised ITG models," in *Proceedings of ACL-IJCNLP*, 2009.

[13] Y. S. Chan, H. T. Ng, and D. Chiang, "Word sense disambiguation improves statistical machine translation," in *Proceedings of ACL*, 2007.