The ICT System Description for IWSLT 2008

Yang Liu, Zhongjun He, Haitao Mi, Yun Huang, Yang Feng, Wenbin Jiang, Yajuan Lu, Qun Liu

Key Laboratory of Intelligent Information Processing
Institute of Computing Technology
Chinese Academy of Sciences
P.O. Box 2704, Beijing, China, 100190
{yliu, zjhe, htmi, huangyun, fengyang, jiangwenbin, lvyajuan, liuqun}@ict.ac.cn

Abstract

This paper presents a description for the ICT systems involved in the IWSLT 2008 evaluation campaign. This year, we participated in Chinese-English and English-Chinese translation directions. Four statistical machine translation systems were used: one linguistically syntax-based, two formally syntax-based, and one phrase-based. The outputs of the four SMT systems were fed to a sentence-level system combiner, which was expected to produce better translations than single systems. We will report the results of the four single systems and the combiner on both the development and test sets.

1. Introduction

The ICT system for IWSLT 2008 is a combination of four statistical machine translation systems:

1. Silenus, a linguistically syntax-based system that uses tree-to-string rules learned from packed forests;
2. Bruin, a formally syntax-based system that implements a maximum entropy based reordering model on BTG rules;
3. Mencius, a phrase-based system that enables lexicalized reordering and similarity-based partial matching of bilingual phrases;
4. Change, a formally syntax-based system that employs hierarchical phrases.

They are combined at sentence level using a general linear model.

This year, we participated in three tracks (two translation directions):

1. BTEC task, Chinese-English direction;
2. Challenge task, Chinese-English direction;
3. Challenge task, English-Chinese direction.

This paper is structured as follows. Section 2 describes the four SMT systems and the combiner; Section 3 gives the experimental results, and Section 4 is the conclusion.

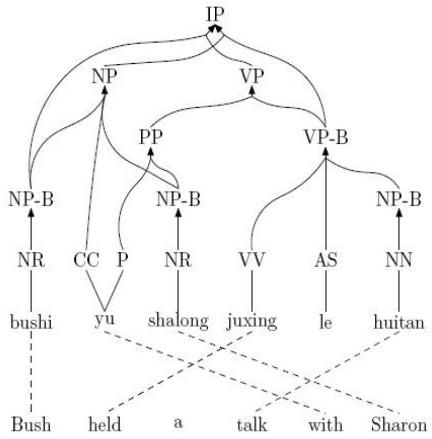

Figure 1: *A pair of linked source forest and target string. The solid lines denote hyperedges and the dashed lines denote word alignments.*

2. SMT Systems

2.1. Silenus

Deriving from the tree-to-string system Lynx [1, 2], Silenus [3, 4] uses packed forests instead of 1-best parse trees in both training and decoding.

A packed parse forest is a compact representation of all derivations (i.e., parse trees) for a given sentence under a context-free grammar [5]. A forest can be formally defined as a tuple $\langle V, E, \bar{v}, \mathbf{R} \rangle$, where V is a finite set of nodes, E is a finite set of hyperedges, $\bar{v} \in V$ is a distinguished node that denotes the goal item in parsing, $\mathbf{R}$ is the set of weights. For a given sentence $w_{1:l} = w_1 \dots w_l$, each node $v \in V$ is in the form if $X_{i,j}$, which denotes the recognition of non-terminal X spanning from position i through j (that is, $w_{i+1} \dots w_j$). Each hyperedge $e \in E$ is a triple $e = \langle T(e), h(e), f(e) \rangle$, where $h(e) \in V$ is its head, $T(e) \in V^*$ is a vector of tail nodes, and $f(e)$ is a weight function from $\mathbf{R}^{|T(e)|}$ to $\mathbf{R}$.

Figure 1 shows a pair of Chinese forest and English string. The solid lines denotes hyperedges and the dashed

Table 1: *A minimal derivation of the example shown in Figure 1.*

(1)	IP(x_1:NP-B,x_2:VP)→ x_1 x_2
(2)	NP-B(x_1:NR)→ x_1
(3)	NR(bushi)→Bush
(4)	VP(x_1:PP,x_2:VP-B)→ x_1 x_2
(5)	PP(x_1:P, x_2:NP-B)→ x_1 x_2
(6)	P(yu))→with
(7)	NP-B(x_1:NR)→ x_1
(8)	NR(shalong)→Sharon
(9)	VP-B(x_1:VV, AS(le), x_2:NP-B)→ x_1 a x_2
(10)	VV(juxing) →held
(11)	NN(huitan)→talk

lines denote word alignments. Each hyperedge is associated with a probability, which we omit in Figure 1 for clarity. In a forest, a node usually has multiple incoming hyperedges. For example, the source node $\text{IP}_{0,6}$ has two incoming hyperedges:

$$e_1 = \langle(\text{NP-B}_{0,1}, \text{VP}_{1,6}), \text{IP}_{0,6}, 0.6\rangle$$
$$e_2 = \langle(\text{NP}_{0,3}, \text{VP-B}_{3,6}), \text{IP}_{0,6}, 0.4\rangle$$

Silenus searches for the best derivation (a sequence of translation rules) $\hat{d}$ that converts a source tree T in the packed forest into a target-language string s:

$$\hat{d} = \underset{d \in D}{\text{argmax}}\, Pr(d|T) \tag{1}$$

Table 1 gives a derivation for the example forest-string pair.

To learn tree-to-string rules from annotated training data, we follow GHKM [6] to first identify minimal rules and then obtain composed rules. Like in tree-based extraction, we extract rules from a packed forest F in two steps: frontier set computation (where to cut) and fragmentation (how to cut). It turns out that the exact formulation developed for frontier set in tree-based case can be applied to a forest without change. The fragmentation step, however, becomes much more complicated since we now face a choice of multiple hyperedges at each node.

We develop a breadth-first search algorithm for extracting tree-to-string rules from packed forests. The basic idea is to visit each frontier node v, and keep a queue $open$ of growing fragments rooted at v. We keep expanding incomplete fragments from $open$, and extract a rule if a complete fragment is found. Some minimal rules learned from the example forest-string pair are listed in Table 1.

In tree-based extraction, for each sentence pair, each rules extracted naturally has a count of 1, which will be used in maximum-likelihood estimation of rule probabilities. However, a forest is an implicit collection of trees. Each tree has its own probability (that is, product of hyperedge probabilities). As a result, a rule extracted from non 1-best parse should be penalized accordingly and should have fractional counts instead of unit count.

We penalize a rule r by the posterior probability of the corresponding tree fragment $t = lhs(r)$, which can be computed as the product of the outside probability of its root, the inside probabilities of its leaf nodes, and the probabilities of hyperedges involved in the fragment:

$$\alpha\beta(t) = \alpha(root(t)) \times \prod_{e \in t} P(e) \times \prod_{v \in leaves(t)} \beta(v) \tag{2}$$

where $\alpha(\cdot)$ and $\beta(\cdot)$ are the outside and inside probabilities of nodes, $root(\cdot)$ returns the root of a tree fragment and $leaves(\cdot)$ returns the leaf nodes of a tree fragment.

Now, the fractional count of a rule r is simply

$$c(r) = \frac{\alpha\beta(lhs(r))}{\alpha\beta(\bar{v})} \tag{3}$$

where $\bar{v}$ denotes the root of the forest.

We extend the simple model in Eq. 1 to a log-linear model [7]:

$$\hat{d} = \underset{d \in D}{\text{argmax}}\, Pr(d|T)^{\lambda_1} \times p_{lm}(s)^{\lambda_2} \times e^{\lambda_3|d|} \times e^{\lambda_4|s|} \tag{4}$$

where $p_{lm}(s)$ is the language model score, $|d|$ is the number of rules in a derivation, and $|s|$ is the number of target words produced. The derivation probability $Pr(d|T)$ is the product of probabilities of translation rules involved in d:

$$Pr(d|T) = \prod_{r \in d} Pr(r) \tag{5}$$

where each $Pr(r)$ can be decomposed into the product of six probabilities:

$$\begin{aligned} Pr(r) \quad = \quad & p(r|lhs(r))^{\lambda_5} \times p(r|rhs(r))^{\lambda_6} \\ & \times p(r|root(lhs(r)))^{\lambda_7} \\ & \times p_{lex}(lhs(r)|rhs(r))^{\lambda_8} \\ & \times p_{lex}(rhs(r)|lhs(r))^{\lambda_9} \\ & \times p(T)^{\lambda_{10}} \end{aligned} \tag{6}$$

where the first three terms are conditional probabilities based on fractional counts, $p_{lex}(\cdot)$ denotes lexical weighting, and $p(T)$ denotes the probability of the matched source tree T.

To search for 1-best derivation, the decoder employs the cube pruning method [8] that approximately intersects the translation forest with language model. Basically, cube pruning works bottom up in a forest, keeping at most k +LM items at each node, and uses the best-first expansion idea from the Algorithm 2 of Liang Huang [9] to speed up the computation. K-best derivations can also be easily obtained by applying Algorithm 3 of Liang Huang [9].

2.2. Bruin

Bruin [10] is a formally syntax-based system that implements a maximum entropy based reordering model on BTG [11]

rules. Bruin employs the following three BTG rules to direct translation:

$$A \xrightarrow{[\,]} (A^1, A^2) \tag{7}$$

$$A \xrightarrow{\langle\,\rangle} (A^1, A^2) \tag{8}$$

$$A \rightarrow (x, y) \tag{9}$$

The first two rules are used to merge two neighboring blocks into one larger block either in a monotonic or an inverted order. A block is a pair of source and target contiguous sequences of words. The last rule translates a source phrase x into a target phrase y and generate a block A.

In the following, we will define the model by separating different features (including the language model) from the rule probabilities and organizing them in a log-linear model. This straight way makes it clear how rules are used and what they depend on.

For the two merging rules, applying them on two consecutive blocks A^1 and A^2 is assigned a probability $Pr^m(A)$:

$$Pr^m(A) = \Omega^{\lambda_\Omega} \cdot \triangle_{p_{LM}}(A^1, A^2)^{\lambda_{LM}} \tag{10}$$

where Ω is the reordering score of blocks A^1 and A^2, λ_Ω is its weight, and $\triangle_{p_{LM}}(A^1, A^2)$ is the increment of the language model score of two blocks according to their final order, and λ_{LM} is its weight.

The application of a lexical rule is assigned a probability $Pr^l(A)$:

$$\begin{aligned} Pr^l(A) \quad = \quad & p(x|y)^{\lambda_1} \cdot p(y|x)^{\lambda_2} \cdot p_{lex}(x|y)^{\lambda_3} \\ & \cdot p_{lex}(y|x)^{\lambda_4} \cdot exp(1)^{\lambda_5} \cdot exp(|x|)^{\lambda_6} \\ & \cdot p_{LM}(x)^{\lambda_{LM}} \end{aligned} \tag{11}$$

where $p(\cdot)$ are the phrase translation probabilities in both directions, $p_{lex}(\cdot)$ are the lexical translation probabilities in both directions, and $exp(1)$ and $exp(|x|)$ are the phrase penalty and word penalty, respectively.

We define the reordering model Ω on three factors: an order o, a block A^1, and a block A^2. Given two neighboring blocks A^1 and A^2, the central problem is how to predict their order $o \in \{monotonic, inverted\}$. This is a typical two-class classification. To estimate the conditional probability $p(o|A^1, A^2)$, a reasonable way is to use features of blocks as reordering evidences under maximum entropy model:

$$p_\theta(o|A^1, A^2) = \frac{exp(\sum_i \theta_i h_i(o, A^1, A^2))}{\sum_{o'} exp(\sum_i \theta_i h_i(o', A^1, A^2))} \tag{12}$$

where $h_i(o, A^1, A^2) \in \{0, 1\}$ is a feature function and θ_i is the corresponding feature weight.

There are three steps to train a maximum entropy based reordering model: (1) first extract reordering examples from word-aligned bilingual corpora, (2) then generate features from these examples, and (3) finally estimate the feature weights of maximum entropy model.

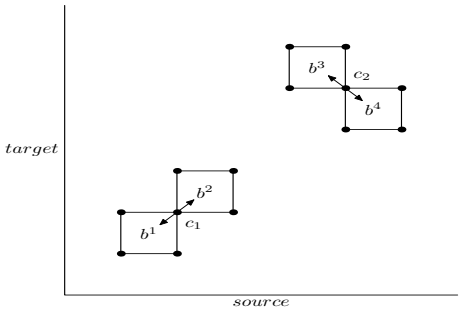

Figure 2: *Blocks and orders. The bold dots are corners and the arrows from the corners are their links.*

We define each vertex of a block as a *corner*. Each corner has four *links* in four directions: *top-left*, *top-right*, *bottom-left*, and *bottom-right*. Each link connects two blocks that share with the same corner. The *top-right* and *bottom-left* links connect blocks in a monotonic order and *top-left* and *bottom-right* links connect blocks in an inverted order. For example, there are four blocks, b^1, b^2, b^3, and b^4, in Figure 2. b^1 and b^2 share with a corner c_1 and b^3 and b^4 share with a corner c_2. According to above definitions, b^1 and b^2 are connected in a monotonic order by the *bottom-left* and *top-right* links of c_1, and b^3 and b^4 are connected in an inverted order by the *top-left* and *bottom-right* links of c_2.

Two kinds of features are designed for our MaxEnt-based reordering model: lexical features and collocation features. Lexical features are defined on the first words of source and target phrases. Collocation features are defined on the combination of boundary words. Why are we particularly interested in boundary words? We believe that boundary words of blocks capture information of block reordering. To test this assumption, we calculate the information gain ratio(IGR) for boundary words as well as the whole blocks against the order on the extracted reordering orders. IGR measures how precisely a feature f predicts a class c:

$$IGR(f, c) = \frac{E(c) - E(c|f)}{Ef} \tag{13}$$

where $E(\cdot)$ is an entropy and $E(\cdot|\cdot)$ is a conditional entropy. Surprisingly, the IGR for boundary words (0.2637) is very close to that of blocks (0.2655), suggesting that boundary words do provide sufficient information for predicting reordering.

Based on CKY algorithm, the decoder finds the best derivation that produces the input sentence and its translation. To speed up the computation, Bruin also makes use of cube pruning. The lazy Algorithm 3 [9] are used for n-best list generation.

2.3. Mencius

Mencius [12] is a phrase-based system that is very similar to Moses [13]. The major difference is that we introduce

similarity-based partial matching for bilingual phrases to alleviate data sparseness problem.

Given two source phrases $\hat{f}_1^J$ and $\hat{f'}_1^J$, their matching similarity is given by

$$\text{SIM}(\hat{f}_1^J, \hat{f'}_1^J) = \frac{\sum_{j=1}^{J} \delta(f_j, f'_j)}{J} \quad (14)$$

where

$$\delta(f, f') = \begin{cases} 1 & \text{if } f = f' \\ 0 & otherwise \end{cases} \quad (15)$$

Note that we only consider two source phrases that have the same length. To make partially matching more reliable, we further restrict that they share with the same parts-of-speech sequence.

Our hope is that similar bilingual phrases can be used to create translation templates if one source phrase cannot find translations in the phrase table. For example, suppose that we cannot find translations for a source phrase "*yu zuotian dida taiguo*" in a phrase table, in which we find a similar source phrase "*yu zuowan dida bulage*" with its translation "arrived in Prague last evening". According to the alignment information, we obtain a translation template:

$$\langle \textit{yu } X_1 \textit{ dida } X_2\textit{, arrived in } X_2\ X_1 \rangle$$

Then, the unmatched source substrings "*zuotian*" and "*taiguo*" can be translated into "*yesterday*" and "*Thailand*", respectively. As a result, the translation for "*yu zuotian dida taiguo*" is "*arrived in Thailand yesterday*".

Given a source sentence, the decoder firstly search for all possible translation options from the phrase table by exact matching. For source phrases which have no translations, we construct translations by similarity-based partially matching, as shown in above example. Then, the decoder works the same as Moses does.

2.4. Change

Change is an implementation of the state-of-the-art hierarchical phrase-based model. Considered as an extension of standard phrase-based model, hierarchical phrase-based model allows non-contiguous parts of source sentence to be translated into possibly non-contiguous parts of target sentence. The model can be formalized as a synchronous context-free grammar, in which a rule is of the form:

$$X \rightarrow \langle \gamma, \alpha, \sim \rangle \quad (16)$$

where X is a non-terminal, γ and α are strings of terminals and non-terminals, and $\sim$ is a one-to-one correspondence between the non-terminals of γ and α.

Our implementation faithfully follows Chiang's work [8]. The only exception is the condition for terminating cube pruning. Chiang's implementation [8] quits upon considering the next item if its score falls outside the beam by more than ϵ. We find that large number of items will often be enumerated under this condition in our experiments. To tackle this problem, we further limit the number of items taken from the heap.

2.5. System Combination

We combine the outputs of single SMT systems at sentence level, similarly to the work by Macherey and Och [14]. Global linear models are used as a framework for reranking a merged n-best list:

$$\hat{y} = \underset{y \in \text{GEN}(x)}{\text{argmax}}\ \mathbf{f}(x, y) \cdot \mathbf{W} \quad (17)$$

where x is a source sentence, y is a translation, $\mathbf{f}(x, y)$ is a feature vector, $\mathbf{W}$ is a weight vector, and $\text{GEN}(x)$ is the set of possible candidate translations.

There types of features are used: (1) relative BLEU scores against 1-best translations from other candidates, (2) language model scores, and (3) length of the translation. The feature weights are tuned using minimum-error-rate training [15]. In this year's evaluation, each single SMT system generated 200-best list translations, which were merged and served as the input to the combiner.

3. Experimental Results

3.1. Data

Besides the data provided by the organizer, we used the following additional data [1]:

1. Chinese LDC (CLDC-LAC-2003-004);
2. Chinese LDC (CLDC-LAC-2003-006);
3. Chinese LDC (2004-863-008);
4. Chinese LDC (2004-863-009);
5. LDC2002L27 "Chinese-English Translation Lexicon Version 3.0";
6. LDC2005T34 "Chinese-English Named Entity Lists Version 1.0";
7. "Tanaka's Corpus".

The training corpus contains about $8.1M$ Chinese words and $8.6M$ English words. The data were used by all the four single systems to train their model parameters respectively. The English sentences of training corpus were used to train a 5-gram language model using SRILM [16]. Similarly, the Chinese part was used to train a 5-gram language model for English-to-Chinese direction.

[1] They are allowed for usage according to the "resource" page of the official IWSLT 2008 website

Table 3: *BLEU scores of five systems on test sets.*

System	BTEC_CE		CT_CE		CT_EC	
	CRR	ASR.1	CRR	ASR.1	CRR	ASR.1
Silenus	0.4639	0.4061	0.4048	0.3541	0.4988	0.4119
Bruin	0.4796	0.4097	0.4114	0.3280	0.4746	0.3950
Mencius	0.4689	0.3960	0.3850	0.3050	0.4700	0.3864
Change	0.4712	0.3946	0.4066	0.3308	0.4815	0.3961
Combination	0.4943	0.4403	0.4370	0.3703	0.5045	0.4249

Table 2: *BLEU scores of five systems on IWSLT 2007 Chinese-English development set.*

System	provided	provided + additional
Silenus	0.3413	0.3919
Bruin	0.3572	0.4213
Mencius	0.3692	0.4341
Change	0.3792	0.4287
Combination	0.3894	0.4483

3.2. Annotation

We used the Chinese lexical analysis system ICTCLAS for splitting Chinese characters into words and the tokenizer provided by IWSLT for tokenizing English sentences. After that, we convert all alphanumeric characters to their 2-byte representation. Then, we ran GIZA++ and used the "grow-diagfinal" heuristic to get many-to-many word alignments.

We observe that in a sentence some phrases are more likely to appear at the beginning, while other phrases are more likely to be located at the end. Inspired by the literature in language modeling, we mark the beginning and ending of word aligned sentences with two tags, "⟨s⟩" and "⟨/s⟩", to capture such reordering information. The sentences to be translated will also be annotated with the two tags, which will be removed after decoding.

To get packed forests for Silenus, we used the Chinese parser modified [17] by Haitao Mi and the English parser [18] modified by Liang Huang to produce entire parse forests. Then, we ran the Python scripts [19] provided by Liang Huang to output packed forests. To prune the packed forests, Huang [19] uses inside and outside probabilities to compute the distance of the best derivation that traverses a hyperedge away from the globally best derivation. A hyperedge will be pruned if the difference is greater than a threshold. Nodes with all incoming hyperedges pruned are also pruned.

3.3. Results

Table 2 presents the BLEU scores (case-sensitive, with punctuations) of our five systems achieved on the IWSLT 2007 Chinese-English development set. Prior to the evaluation, we used the development sets from 2003 to 2006 as development sets to tune model scaling factors and used 2007 development set as test set. "provided" denotes the training data provided by the organizer that consist of about $30K$ pairs of sentences. "provided+additional" denotes all the training data we have, as listed at the beginning of Section 3.1. We observe that using more data results in substantial improvements of about 5 BLEU points.

Table 3 gives the BLEU scores (case-sensitive, with punctuations) of our five systems achieved on the test sets. "BTEC_CE" denotes Chinese-English direction of BTEC task, "CT_CE" denotes Chinese-English direction of challenge task, and "CT_EC" denotes English-Chinese direction of challenge task. "CRR" denotes correct recognition results and "ASR.1" denotes using 1-best ASR output.

Our sentence-level system combiner outperformed single systems consistently on all tasks. While system combination benefited Chinese-English direction significantly, the improvements on English-Chinese direction were relatively small. One possible reason might be that fewer development sets are available for English-Chinese direction for system combiner to optimize the parameters automatically.

For single SMT systems, Bruin got better results than the others on Chinese-English direction. Interestingly, Silenus surpassed other systems significantly on English-Chinese direction. There are two findings worth noting:

1. Silenus uses packed forests instead of 1-best parses, minimizing the negative effect of parsing errors. As the amount and domain of data used for training parsers are comparatively limited, parsers will inevitably output ill-formed trees when handle real-world text. Guided by such noisy syntactic information, syntax-based models that rely on only 1-best parses are prone to produce degenerate translations. The results suggest that packed forests do help syntax-based systems to achieve comparable performance with phrase-based systems on tourism-related sentences.

2. Parsing accuracy has a substantial effect on syntax-based models. Silenus obtained better results on English-Chinese direction than Chinese-English direction. We believe the major reason is that parsing on English is more accurate than Chinese.

4. Conclusion

In this paper, we give a brief introduction to our four single SMT systems and one system combiner. We report the resources used, annotation techniques, and results achieved on the test sets. We find that our implementation of sentence-level system combination works for all tasks. Another interesting finding is that syntax-based models could produce translations as good as phrase-based systems on tourism-related text if packed forests are used.

5. Acknowledgements

This authors were supported by National Science Foundation of China, Contracts 60736014 and 60573188, and 863 State Key Project No. 2006AA10108.

6. References

[1] Y. Liu, Q. Liu, and S. Lin, "Tree-to-string alignment template for statistical machine translation," in *Proceedings of COLING/ACL 2006*, Sydney, Australia, July 2006, pp. 609–616.

[2] Y. Liu, Y. Huang, Q. Liu, and S. Lin, "Forest-to-string statistical rules," in *Proceedings of ACL 2007*, Prague, Cezch Republic, June 2007, pp. 704–711.

[3] H. Mi, L. Huang, and Q. Liu, "Forest-based translation," in *Proceedings of ACL/HLT 2008*, Columbus, Ohio, USA, June 2008, pp. 192–199.

[4] ——, "Forest-based translation rule extraction," in *Proceedings of EMNLP 2008*, Hawaii, USA, Oct. 2008.

[5] S. Billot and B. Lang, "The structure of shared forests in ambiguous parsing," in *Proceedings of ACL 1989*, 1989, pp. 143–151.

[6] M. Galley, M. Hopkins, K. Knight, and D. Marcu, "What's in a translation rule?" in *Proceedings of HLT/NAACL 2004*, Boston, Massachusetts, USA, May 2004, pp. 273–280.

[7] F. J. Och and H. Ney, "Discriminative training and maximum entropy models for statistical machine translation," in *Proceedings of ACL 2002*, 2002, pp. 295–302.

[8] D. Chiang, "Hierarchical phrase-based translation," *Computational Linguistics*, vol. 33, no. 2, pp. 201–228, 2007.

[9] L. Huang and D. Chiang, "Better k-best parsing," in *Proceedings of IWPT 2005*, Vancouver, Canada, Oct. 2005, pp. 53–64.

[10] D. Xiong, Q. Liu, and S. Lin, "Maximum entropy based phrase reordering model for statistical machine translation," in *Proceedings of COLING/ACL 2006*, Sydney, Australia, July 2006, pp. 521–528.

[11] D. Wu, "stochastic inversion transduction grammars and bilingual parsing of parallel corpora," *Computational Linguistics*, vol. 23, pp. 377–404, 1997.

[12] Z. He, Q. Liu, and S. Lin, "Partial matching strategy for phrase-based statistical machine translation," in *Proceedings of ACL/HLT 2008 (Short Paper)*, Columbus, Ohio, June 2008, pp. 161–164.

[13] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, and O. Bojar, "Moses: Open source toolkit for statistical machine translation," in *Proceedings of ACL 2007 (poster)*, Prague, Czech Republic, June 2007, pp. 177–180.

[14] W. Macherey and F. J. Och, "An empirical study on computing consensus translations from multiple machine translation systems," in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pp. 986–995. [Online]. Available: http://www.aclweb.org/anthology/D/D07/D07-1105

[15] F. J. Och, "Minimum error rate training in statistical machine translation," in *Proceedings of ACL 2003*, 2003, pp. 160–167.

[16] A. Stolcke, "Srilm - an extensible language modeling toolkit," in *Proceedings of International Conference on Spoken Language Processing*, vol. 30, 2002, pp. 901–904.

[17] D. Xiong, S. Li, Q. Liu, and S. Lin, "Parsing the penn chinese treebank with semantic knowledge," in *Proceedings of IJCNLP 2005*, 2005, pp. 70–81.

[18] E. Charniak and M. Johnson, "Coarse-to-fine n-best parsing and maxent discriminative reranking," in *Proceedings of ACL 2005*, Ann Arbor, Michigan, June 2005, pp. 173–180.

[19] L. Huang, "Forest reranking: Discriminative parsing with non-local features," in *Proceedings of ACL 2008*, Columbus, Ohio, June 2008, pp. 586–594.