A Pilot Study Towards End-to-End MT Training

Feifei Zhai ffzhai2012@gmail.com
Queens College, City University of New York, Queens, NY 11367

Liang Huang[*] lianghuang.sh@gmail.com
School of EECS, Oregon State University, Corvallis, OR 97330

Abstract

Typical MT training involves several stages, including word alignment, rule extraction, translation model estimation, and parameter tuning. In this paper, different from the traditional pipeline, we investigate the possibility of end-to-end MT training, and propose a framework which combines rule induction and parameter tuning in one single module. Preliminary experiments show that our learned model achieves comparable translation quality to the traditional MT training pipeline.

1 Introduction

Typically, as shown in Figure 1(a), traditional machine translation (MT) training involves a long pipeline of several stages, including word alignment (by GIZA++), rule extraction, translation model (TM) estimation (by max-likelihood), and parameter tuning by MERT (Och, 2003), PRO (Hopkins and May, 2011), or MIRA (Watanabe et al., 2007; Chiang et al., 2008). This cascaded procedure inevitably propagates errors downstream, while at the same time making MT training overly complicated.

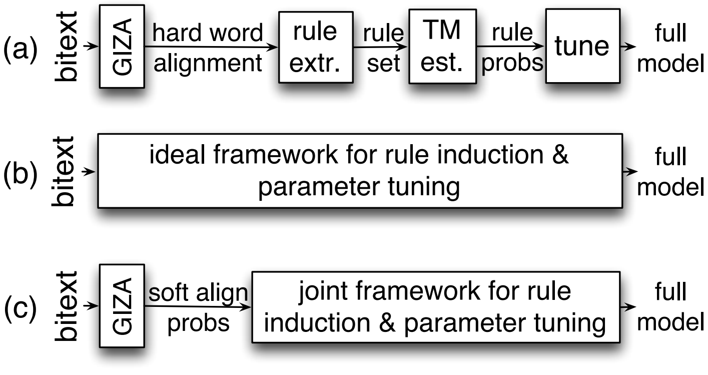

Figure 1: Three approaches to MT training. (a) the standard pipeline; (b) ideal end-to-end MT training; (c) our work, in between (a) and (b), combines rule induction, TM estimation, and parameter tuning in one module.

In this paper, instead of following the traditional training pipeline, we explore the possibility of end-to-end MT training, which would ideally induce a full model with translation

[*] Work done while Prof. Liang Huang was in City University of New York.

rules at the same time (as shown in Figure 1 (b)). Practically, it is a big challenge, because the search space would be too prohibitive if we consider all possible rules from scratch without any constraints from word alignment.

To make this idea computationally affordable, we take a step back and propose a joint learning framework to do rule induction and parameter tuning together, shown in Figure 1 (c). Instead of knowing nothing at the beginning, we use word translation probabilities from GIZA++ to compute the lexical translation probabilities of phrase pairs, which is an effective feature guiding us to select good translation rules. Note that even with these probabilities, this is still a non-trival problem, since the joint learner still needs to deal with the entire space of all possible translation rules and all possible decoding derivations.

For simplicity reasons we use phrase-based translation. Final experiments show that the joint learned model achieves comparable performance to the conventional training pipeline in a small scale. To our best knowledge, this is the first time, although on small data, verifying that it is possible to do effective end-to-end MT training. The most significant contribution of this paper lies in this point. We believe this will be a promising direction to MT research.

2 Joint Framework for Rule Induction and Parameter Tuning

Algorithm 1 gives our joint framework in detail. We start with an empty rule set $R = \emptyset$, and for each sentence pair (x, y) where x is the source input and y the target translation, we try forced decoding finding all derivations that can map x to y (line 5), using all possible phrase pairs from (x, y). This is because without a rule set to start with, theoretically any portion of x could map to any portion of y and the learner needs to figure out which phrase pairs make more sense. Here $f(\cdot)$ and $e(\cdot)$ return the source and target projection of a derivation respectively. This forced decoding, although much more constrained than real decoding, is still intractable, and we have to resort to beam search similar to those employed in real decoding, i.e., the set D in line 5 is the "best-scoring" subset of all possible forced derivations .

Algorithm 1 Joint rule induction and param. tuning.

```
 1: $R \leftarrow \emptyset$                                                  ▷ initial rule set
 2: $\mathbf{w} \leftarrow \mathbf{0}$                                       ▷ initial parameters
 3: repeat
 4:   for each sentence pair $(x, y)$ in bitext do
 5:     $D(x,y) \leftarrow \{d \mid f(d)=x, e(d)=y\}$                     ▷ forced decoding
 6:     $R \leftarrow R \cup \{r \mid r \in d, d \in D(x,y)\}$             ▷ add new rules
 7:   recalculate conditional probs for each $r \in R$
 8:   for each sentence pair $(x, y)$ in bitext do
 9:     $d' \leftarrow \mathop{\mathbf{argmax}}_{d: f(d)=x} \mathbf{w} \cdot \mathbf{\Phi}(x, d)$    ▷ real decoding
10:     if $e(d') \neq y$ then                                             ▷ wrong translation?
11:       $d^* \leftarrow \mathop{\mathbf{argmax}}_{d \in D(x,y)} \mathbf{w} \cdot \mathbf{\Phi}(x, d)$   ▷ best gold deriv.
12:       $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{\Phi}(x, d^*) - \mathbf{\Phi}(x, d')$           ▷ update model
13: until converged
```

After forced decoding we collect translation rules in the forced derivations and add them to standing rule set R (line 6). Then we recalculate the rule conditional probabilities (line 7, TM estimation). After that we perform real decoding, trying to find the best translation (line 9) [1] .

[1] At first, since all the weights are set to 0, all derivations have the same score 0. Both the best-scoring forced derivation and real decoding derivation are selected randomly

If this translation $e(d')$ is different from the reference translation y, then an update is needed to reward the highest-scoring forced (or "gold") derivation d^* and to penalize the highest-scoring non-gold (Viterbi) derivation d' (line 12). We apply a latent-variable max-violation perceptron (Huang et al., 2012), which has been successfully used in MT training (Yu et al., 2013; Zhao et al., 2014), to do update.

Technically, this framework is similar to (Xiao and Xiong, 2013). The main difference is that we try to do end-to-end MT training, and combine rule induction, TM estimation, and parameter tuning together, while they only focus on rule induction. Moreover, to accommodate the vast amount of search errors in decoding, we update weights by max-violation perceptron, performing prefix instead of full-sequence updates, whereas the updates in Xiao and Xiong (2013) are still full-sequence updates which are insensitive to search errors. For simplicity reasons we do not make this difference explicit in line 12.

3 Phrase-based Forced Decoding

Forced decoding generates those derivations that can produce the exact reference translation. For example, given the following sentence pair,

> **Source:** *Bùshí yǔ Shālóng jǔxíng le huìtán*
> **Target:** *Bush held a talk with Sharon*

One possible derivation created by forced decoding is as follows:

$$\dfrac{\dfrac{\dfrac{({}_0\_\_\_\_\_\_) : (0, \text{``''})}{(\bullet_1\_\_\_\_\_) : (s_1, \text{``Bush''})}\ r_1}{(\bullet\_\_\bullet\bullet\bullet_6) : (s_2, \text{``Bush held a talk''})}\ r_2}{(\bullet\bullet\bullet_3\bullet\bullet\bullet) : (s_3, \text{``Bush held a talk with Sharon''})}\ r_3$$

where each hypothesis is in form $(v) : (s, p)$, in which v is the *coverage vector* (a • indicates the source word at this position is already "covered (or translated)"), and (s, p) is the score and partial translation of each state.

Generally, we can employ a traditional phrase-based decoder to do forced decoding. However, the distortion limit in the decoder will prohibit long-distance reorderings, and exclude many sentence pairs from getting forced derivations, especially for language pairs with very different word orders, such as Chinese and English.

Hence, in order to do better forced decoding, we use a more flexible limit to constrain the number of *gaps* during decoding (also known as "*IBM constraint*" in (Zens et al., 2004)), rather than distortion limit. Here, a *gap* refers to a consecutive of positions that are not covered in the coverage vector. For example, consider the third hypothesis of the above derivation, $(\bullet\_\_\bullet\bullet\bullet_6) : (s_2, \text{``Bush held a talk''})$, its coverage vector has one gap, i.e., the two untranslated words. Also, we don't want the decoding process to be too flexible on reordering, so we demand that there are at most two gaps in a specific coverage vector (See Figure 2).

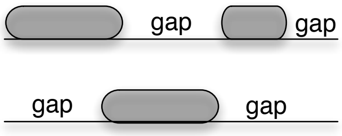

Figure 2: Two possible scenarios in gap-based decoding. The gray boxes denote covered segments.

Obviously, compared to distortion limit, gap limit is more flexible for decoding, especially when there is only one gap in the coverage vector, we can jump to any uncovered position of the source sentence for translation. Hereafter, we use *dl-based decoding* to denote the decoding algorithm using distortion limit, while *gap-based decoding* denotes the algorithm using gap limit.

In addition, same as forced decoding, real decoding can also be dl-based or gap-based. We use the same choice for forced and real decoding.

4 Rule Generation and Collection

As described in § 2, since there is no predefined rule set, we need to consider all possible phrase pairs from the training sentence pair. To reduce the search space of phrase pairs, we set the maximum phrase length to 3 [2]. As an example, considering the above sentence pair, we enumerate all possible source and target phrases from it for forced decoding:

possible source phrases	possible target phrases
Bùshí	*Bush*
Bùshí yǔ	*Bush held*
Bùshí yǔ Shālóng	*Bush held a*
yǔ	*held*
yǔ Shālóng	*held a*
yǔ Shālóng jǔxíng	*held a talk*
...	...

From these phrases, we can pick a source phrase and a target phrase arbitrarily to form a phrase rule, and apply it to forced decoding. For example, as a start, if we select *Bùshí* and *Bush* respectively, we can create hypothesis $(\bullet_1 \_\_\_\_\_) : (s_1, \text{“Bush”})$ in the above derivation. We consider all possible combinations of these phrases as rules for forced decoding.

In addition, since training sentence pairs do not always contain equal information on both sides, we introduce two null rules $\langle f, null\rangle$ and $\langle null, e\rangle$ to capture the redundant information for forced decoding. $\langle f, null\rangle$ deletes a source word, and $\langle null, e\rangle$ inserts a target word to the translation.

After forced decoding, we collect the rules used in forced derivations and add them to the standing rule set R (line 6 in Algorithm 1). [3] Based on the rule counts in R, we recalculate the rule conditional probabilities by the formula from (DeNero et al., 2006):

$$\phi(e|f) = \frac{c(f, e)}{c(f) + k^{l-1}} \tag{1}$$

where f and e are the source and target phrase, $c(\cdot)$ is the count of phrase or phrase pair, l is the length of phrase f, and k is a tuning parameter. The formula boosts the probability of short phrases, and results in better translation quality. After some validation experiments, we set $k = 4.0$ finally.

Similar to our rule generation process, Wuebker and Ney (2013) has proposed a length-based training method to do rule induction by EM algorithm. The major difference between our framework and theirs is that their algorithm only relates to rule induction, and still need MERT to do parameter tuning, while we combine rule induction and parameter tuning together. In this way, the two step in our framework can help each other, but the two step in (Wuebker and Ney,

[2] We have also tried longer length limit, but the performance becomes worse, because with longer limit, the learner greatly prefers longer phrases, which are not good at generalization.

[3] We count all rules in all unpruned forced derivations, which are stored in a lattice. The rules are counted based on its contribution to the derivation, i.e., the score it added to the derivation.

2013) are isolated. We think it is possible to integrate their method into our framework. We leave it as our future work.

5 Parameter Tuning

We apply a max-violation perceptron, which has successfully scaled the MT tuning process from dev set to training data (Yu et al., 2013), to do parameter tuning. We skip the details here. The basic idea is to find the step where the difference (violation) between the best forced decoding derivation and the best real decoding derivation is maximal, and then update parameters at this step so that most information can be learned.

Theoretically, the max-violation perceptron allows arbitrary features. However, in our joint learning scenario, we find it is very easy to get overfitting with sparse features. We conjecture that this is because sparse features have a tight connection to rules. Once some bad rules are introduced, it is difficult for the learner to correct them. We will make more effort on this in future.

Here, we only use dense features, which are the same as the ones for phrase-based translation, including bidirectional translation probabilities (computed by Formula (1) in § 4) [4] , bidirectional lexical translation probabilities (estimated based on (Koehn et al., 2003) by word translation probabilities from Moses (Koehn et al., 2007) based on GIZA++), language model, rule penalty, length penalty, and distortion cost. To simplify the system, we haven't used the lexicalized reordering model here.

6 Related Work

On a high level, this work is a combination of two different research directions. One direction is to induce translation rules directly from bitext, rather than using word alignment (Marcu and Wong, 2002; Cherry and Lin, 2007; Zhang et al., 2008; DeNero et al., 2008; Blunsom et al., 2009; Neubig et al., 2011; Levenberg et al., 2012; Xiao and Xiong, 2013). They can learn better translation rules, but don't care about parameter tuning of SMT. Another direction is discriminative training for MT parameter tuning (Liang et al., 2006; Arun and Koehn, 2007; Blunsom et al., 2008; Flanigan et al., 2013; Green et al., 2013; Yu et al., 2013; Zhao et al., 2014). Both the two directions have achieved promising results. We differ from these works in that we make efforts to combine their spirit together, and try to do rule induction and parameter tuning in one step.

7 Experiments

To evaluate our method, we conduct experiments on Chinese-to-English transaltion. Since we need to do forced decoding and real decoding on training corpus each iteration, to guarantee the training efficiency, here we use a small scale data. It includes about 100K sentence pairs from FBIS, where the length of each sentence is less than 30 words. We use GIZA++ and *grow-diag-final-and* strategy to create symmetric word alignment. We train a trigram language model on 1.5M English sentences.

We base our experiments on **Cubit**, a state-of-the-art phrase-based system in Python Huang and Chiang (2007). For the joint learning method, we set the beam size for forced decoding as 10, real decoding as 30. A maximum phrase length of three was used for both baseline and our joint system. The beam size for final test decoding is set to 50.

We take the newswire portion of NIST MT 2006 data as our dev set, and the NIST MT 03-05 data and the newswire portion of 2008 data as the test set. For baseline, we use MERT Och (2003) to tune weights.

[4] For the newly generated rules which do not have any counts, we assign a very small probability for them.

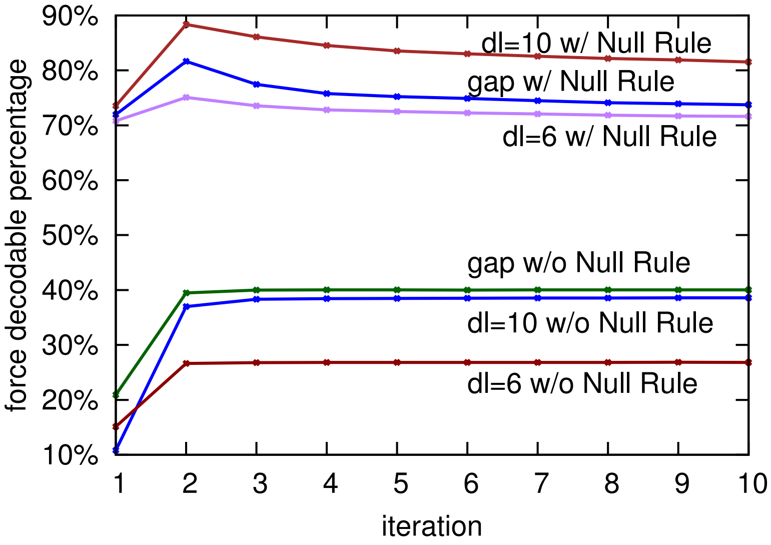

Figure 3: The percentage of reachable sentences in forced decoding at each iteration, with various decoding options.

7.1 DL-based Decoding vs. Gap-based Decoding

We use dl-based decoding and gap-based decoding respectively to promote our joint learning framework. Since we only use dense feature for training, the learning process peak very fast, and we can reach the best BLEU score on dev set in 4 - 5 iterations for both dl-based decoding or gap-based decoding.

Figure 3 compares the forced decoding reachability, and Table 1 shows the corrsponding translation result of different decoding settings. From the Figure and Table, we can first certify the necessity of Null Rule, which improves translation quality significantly. Also, distortion limit has a big influence. Big limit (*dl*=10) allows more flexibility on forced decoding, and thus extract more effective rules, and get better translation quality than smaller limit (*dl*=6). Finally, gap limit gets the best translation quality among all the three, verifying the effectiveness of gap-based decoding.

	decoding	# rule	dev	test
w/o Null Rule	dl=6	0.44M	22.04	21.80
	dl=10	0.55M	23.86	22.56
	gap ≤ 2	0.59M	23.85	22.85
w/ Null Rule	dl=6	0.93M	23.05	22.17
	dl=10	0.94M	24.19	23.47
	gap ≤ 2	1.02M	24.42	23.67

Table 1: Rule set size and BLEU results of our joint learning method with different decoding algorithms.

An interesting phenomenon in Figure 3 is that the gap-based decoding algorithm gets more forced decodable sentences than dl-based decoding (*dl*=10) without Null Rule, but gets fewer ones after adding Null Rule. This is because when we use Null Rule, once the jump exceeds

system		# rule	03	04	05	08	All
baseline	dl=6	1.18M	21.72	25.47	**22.01**	21.59	23.37
	gap	1.18M	21.19	25.18	21.60	21.57	23.05
joint	dl=6	0.93M	20.45	24.37	20.52	20.12	22.17
	gap	1.02M	**21.93**	**25.92**	21.86	**21.85**	**23.67**

Table 2: BLEU scores of different translation systems. The baseline system uses conventional pipeline training. "All" is a combination of the 4 test sets.

distortion limit (bigger than 10), the dl-based forced decoder will not jump, but use Null Rule to generate the corresponding target word, and adopt another Null Rule to delete the source word later. In this way, the dl-based decoder could handle any long-distance jump. Since dl-based decoding is also more flexible than the gap-based one in short-distance jump, it is reasonable that dl-based decoding gets more forced decodable sentences after adding Null Rule. This is also the reason why dl-based decoding collects fewer translation rules (Table 1) with more forced decodable sentences. Since it utilizs many Null Rules on building forced derivations, the number of useful rules decreases in these derivations, resulting in fewer useful translation rules.

7.2 Translation Results

Table 2 shows the final translation results. First, it is interesting that gap-based decoding is better than dl-based in our joint framework, but slightly worse in baseline. This is because with target translation as a constraint in forced decoding, gap limit is more flexible to get better forced derivation (3), and thus more effective rules for better translation. But with a fixed rule set in baseline, its flexibility will introduce more noise into the beam, leading to worse performance.

Moreover, our gap-based joint learned model is better than gap-based baseline by 0.6 BLEU points, and better than dl-based baseline by 0.3 BLEU points.

7.3 Large Data

As previous experiment only trains word alignment on the small 100K corpus, which might create bad alignment and be unfair to baseline, we try another experiment and train word alignment on 2M sentence pairs, but translation model on the original 100K data. The result on the "All" test set is 24.46 for dl-based baseline (dl=6), and 24.57 for our gap-based joint learned model.

7.4 Discussions

The above two experiments have shown that our joint learning framework is comparable to the traditional MT training pipeline.

Based on the experiments, we conclude that three factors influence translation quality. First, in the training corpus, only about 82.9% of all sentence pairs are forced decodable, meaning that 17.1% sentence pairs are excluded from extracting useful rules. Second, due to the overfitting problem, we only use dense features here. Yu et al. (2013) has shown that if only use dense features, max-violation perceptron MT training cannot get good translation quality. At last, we need to handle the space of all possible phrase pairs, and the space of all possible decoding derivations. The search space is too large to optimize effectively.

However, even with these problems, we can still get comparable results with the baseline system, indicating that end-to-end MT training has a great potential to improve, and would be a promising direction for MT research.

8 Conclusion and Future Work

We present a pilot study on end-to-end MT training, and propose a joint framework combining rule induction, TM estimation, and parameter tuning in one module. Preliminary experiments show comparable translation quality to the conventional training pipeline.

Before we can fulfill the ambitious goal of end-to-end MT training, there are still a lot of things to do. In future, we will explore how to effectively use sparse features to improve the translation quality. We also plan to investigate the solution of scaling this framework to large data set. Currently, we need to solve two problems for scaling. The first one is how to deal with long sentences. The number of candidate phrase pairs is exponential to the length of training sentence pairs, making the learning process intractable for long sentence pairs. The practical solution is to segment them into several short ones. We can use IBM model 1 to do that, which has been demonstrated to be effective for SMT (Xu et al., 2005). The second problem is how to use the large data set with millions of sentence pairs. Currently, parallelization seems to be the only solution. After segmentation, we can get millions of short sentence pairs, and then parallelize them to clusters for training.

9 Acknowledgement

We thank the three anonymous reviewers for the valuable comments, and Kai Zhao for discussions. This project was supported in part by DARPA FA8750-13-2-0041 (DEFT), NSF IIS-1449278, and a Google Faculty Research Award.

References

Arun, A. and Koehn, P. (2007). Online learning methods for discriminative training of phrase based statistical machine translation. *Proc. of MT Summit XI*, 2(5):29.

Blunsom, P., Cohn, T., Dyer, C., and Osborne, M. (2009). A gibbs sampler for phrasal synchronous grammar induction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 782–790, Stroudsburg, PA, USA. Association for Computational Linguistics. Gibbs sample initialized by GIZA alignments.

Blunsom, P., Cohn, T., and Osborne, M. (2008). A discriminative latent variable model for statistical machine translation. In *ACL*, pages 200–208.

Cherry, C. and Lin, D. (2007). Inversion transduction grammar for joint phrasal translation modeling. In *Proceedings of SSST, NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 17–24, Rochester, New York. Association for Computational Linguistics.

Chiang, D., Marton, Y., and Resnik, P. (2008). Online large-margin training of syntactic and structural translation features. In *Proceedings of EMNLP 2008.*

DeNero, J., Bouchard-Côté, A., and Klein, D. (2008). Sampling alignment structure under a Bayesian translation model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 314–323, Honolulu, Hawaii. Association for Computational Linguistics.

DeNero, J., Gillick, D., Zhang, J., and Klein, D. (2006). Why generative phrase models underperform surface heuristics. In *Proceedings on the Workshop on Statistical Machine Translation*, pages 31–38, New York City. Association for Computational Linguistics.

Flanigan, J., Dyer, C., and Carbonell, J. (2013). Large-scale discriminative training for statistical machine translation using held-out line search. In *Proceedings of NAACL 2013*.

Green, S., Wang, S., Cer, D., and Manning, C. D. (2013). Fast and adaptive online training of feature-rich translation models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 311–321, Sofia, Bulgaria. Association for Computational Linguistics.

Hopkins, M. and May, J. (2011). Tuning as ranking. In *Proceedings of EMNLP*.

Huang, L. and Chiang, D. (2007). Forest rescoring: Fast decoding with integrated language models. In *Proceedings of ACL*, Prague, Czech Rep.

Huang, L., Fayong, S., and Guo, Y. (2012). Structured perceptron with inexact search. In *Proceedings of NAACL*.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: open source toolkit for statistical machine translation. In *Proceedings of ACL: Demonstrations*.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of NAACL*, pages 127–133.

Levenberg, A., Dyer, C., and Blunsom, P. (2012). A bayesian model for learning scfgs with discontiguous rules. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 223–232, Jeju Island, Korea. Association for Computational Linguistics.

Liang, P., Bouchard-Côté, A., Klein, D., and Taskar, B. (2006). An end-to-end discriminative approach to machine translation. In *Proceedings of COLING-ACL*.

Marcu, D. and Wong, D. (2002). A phrase-based, joint probability model for statistical machine translation. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 133–139. Association for Computational Linguistics.

Neubig, G., Watanabe, T., Sumita, E., Mori, S., and Kawahara, T. (2011). An unsupervised model for joint phrase alignment and extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 632–641, Portland, Oregon, USA. Association for Computational Linguistics.

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

Watanabe, T., Suzuki, J., Tsukada, H., and Isozaki, H. (2007). Online large-margin training for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 764–773, Prague, Czech Republic. Association for Computational Linguistics.

Wuebker, J. and Ney, H. (2013). Length-incremental phrase training for SMT. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 309–319, Sofia, Bulgaria. Association for Computational Linguistics.

Xiao, X. and Xiong, D. (2013). Max-margin synchronous grammar induction for machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 255–264, Seattle, Washington, USA. Association for Computational Linguistics.

Xu, J., Zens, R., and Ney, H. (2005). Sentence segmentation using ibm word alignment model 1. In *Proceedings of EAMT 2005 (10th Annual Conference of the European Association for Machine Translation)*, pages 280–287.

Yu, H., Huang, L., Mi, H., and Zhao, K. (2013). Max-violation perceptron and forced decoding for scalable MT training. In *Proceedings of EMNLP*.

Zens, R., Ney, H., Watanabe, T., and Sumita, E. (2004). Reordering constraints for phrase-based statistical machine translation. In *Proceedings of Coling 2004*, pages 205–211, Geneva, Switzerland. COLING.

Zhang, H., Quirk, C., Moore, R. C., and Gildea, D. (2008). Bayesian learning of non-compositional phrases with synchronous parsing. In *Proceedings of ACL-08: HLT*, pages 97–105, Columbus, Ohio. Association for Computational Linguistics.

Zhao, K., Huang, L., Mi, H., and Ittycheriah, A. (2014). Hierarchical mt training using max-violation perceptron. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 785–790, Baltimore, Maryland. Association for Computational Linguistics.