# Hierarchical MT Training using Max-Violation Perceptron

**Kai Zhao**[†] **Liang Huang**[†] **Haitao Mi**[‡] **Abe Ittycheriah**[‡]

[†]Graduate Center & Queens College
City University of New York
{kzhao@gc,huang@cs.qc}.cuny.edu

[‡]T.J. Watson Research Center
IBM
{hmi,abei}@us.ibm.com

## Abstract

Large-scale discriminative training has become promising for statistical machine translation by leveraging the huge training corpus; for example the recent effort in phrase-based MT (Yu et al., 2013) significantly outperforms mainstream methods that only train on small tuning sets. However, phrase-based MT suffers from limited reorderings, and thus its training can only utilize a small portion of the bitext due to the distortion limit. To address this problem, we extend Yu et al. (2013) to syntax-based MT by generalizing their latent variable "violation-fixing" perceptron from graphs to hypergraphs. Experiments confirm that our method leads to up to +1.2 BLEU improvement over mainstream methods such as MERT and PRO.

## 1 Introduction

Many natural language processing problems including part-of-speech tagging (Collins, 2002), parsing (McDonald et al., 2005), and event extraction (Li et al., 2013) have enjoyed great success using large-scale discriminative training algorithms. However, a similar success on machine translation has been elusive, where the mainstream methods still tune on small datasets.

What makes large-scale MT training so hard then? After numerous attempts by various researchers (Liang et al., 2006; Watanabe et al., 2007; Arun and Koehn, 2007; Blunsom et al., 2008; Chiang et al., 2008; Flanigan et al., 2013; Green et al., 2013), the recent work of Yu et al. (2013) finally reveals a major reason: it is the vast amount of (inevitable) search errors in MT decoding that astray learning. To alleviate this problem, their work adopts the theoretically-motivated framework of violation-fixing perceptron (Huang et al., 2012) tailed for inexact search, yielding great results on phrase-based MT (outperforming small-scale MERT/PRO by a large margin for the first time). However, the underlying phrase-based model suffers from limited distortion and thus can only employ a small portion (about 1/3 in their Ch-En experiments) of the bitext in training.

$$\begin{tikzcd}
\text{Collins (02)} \arrow[r, "\text{inexact}", "\text{search}"'] & \text{Huang et al. (12)} \arrow[r, "\text{latent}", "\text{variable}"'] \arrow[d, "\text{hypergraph}"] & \text{Yu et al. (13)} \arrow[d] \\
& \text{Zhang et al. (13)} \arrow[r] & \textbf{this work}
\end{tikzcd}$$

Figure 1: Relationship with previous work.

To better utilize the large training set, we propose to generalize from phrase-based MT to syntax-based MT, in particular the hierarchical phrase-based translation model (HIERO) (Chiang, 2005), in order to exploit sentence pairs beyond the expressive capacity of phrase-based MT.

The key challenge here is to extend the latent variable violation-fixing perceptron of Yu et al. (2013) to handle tree-structured derivations and translation hypergraphs. Luckily, Zhang et al. (2013) have recently generalized the underlying violation-fixing perceptron of Huang et al. (2012) from graphs to hypergraphs for bottom-up parsing, which resembles syntax-based decoding. We just need to further extend it to handle latent variables. We make the following contributions:

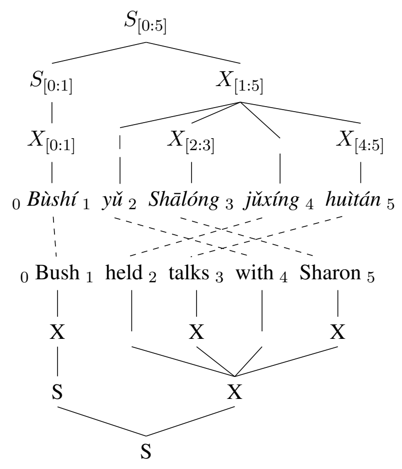
1. We generalize the latent variable violation-fixing perceptron framework to inexact search over hypergraphs, which subsumes previous algorithms for PBMT and bottom-up parsing as special cases (see Fig. 1).

2. We show that syntax-based MT, with its better handling of long-distance reordering, can exploit a larger portion of the training set, which facilitates sparse lexicalized features.

3. Experiments show that our training algorithm outperforms mainstream tuning methods (which optimize on small devsets) by +1.2 BLEU over MERT and PRO on FBIS.

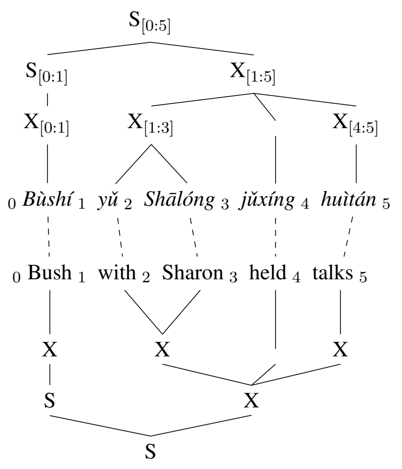
| id | rule |
|---|---|
| $r_0$ | $\mathrm{S} \to \langle \mathrm{X}_{\boxed{1}}, \mathrm{X}_{\boxed{1}} \rangle$ |
| $r_1$ | $\mathrm{S} \to \langle \mathrm{S}_{\boxed{1}}\mathrm{X}_{\boxed{2}}, \mathrm{S}_{\boxed{1}}\mathrm{X}_{\boxed{2}} \rangle$ |
| $r_2$ | $\mathrm{X} \to \langle$*Bùshí*, Bush$\rangle$ |
| $r_3$ | $\mathrm{X} \to \langle$*Shālóng*, Sharon$\rangle$ |
| $r_4$ | $\mathrm{X} \to \langle$*huìtán*, talks$\rangle$ |
| $r_5$ | $\mathrm{X} \to \langle$*yǔ* $\mathrm{X}_{\boxed{1}}$ *jǔxíng* $\mathrm{X}_{\boxed{2}}$, held $\mathrm{X}_{\boxed{2}}$ with $\mathrm{X}_{\boxed{1}}\rangle$ |
| $r_6$ | $\mathrm{X} \to \langle$*yǔ Shālóng*, with Sharon$\rangle$ |
| $r_7$ | $\mathrm{X} \to \langle \mathrm{X}_{\boxed{1}}$ *jǔxíng* $\mathrm{X}_{\boxed{2}}$, $\mathrm{X}_{\boxed{1}}$ held $\mathrm{X}_{\boxed{2}}\rangle$ |

(a) HIERO rules



$S_{[0:5]}$
$S_{[0:1]}$
$X_{[1:5]}$
$X_{[0:1]}$
$X_{[2:3]}$
$X_{[4:5]}$
0 Bùshí 1 yǔ 2 Shālóng 3 jǔxíng 4 huìtán 5
0 Bush 1 held 2 talks 3 with 4 Sharon 5
X
S

(b) gold derivation



$S_{[0:5]}$
$S_{[0:1]}$
$X_{[1:5]}$
$X_{[0:1]}$
$X_{[1:3]}$
$X_{[4:5]}$
0 Bùshí 1 yǔ 2 Shālóng 3 jǔxíng 4 huìtán 5
0 Bush 1 with 2 Sharon 3 held 4 talks 5
X
S

(c) Viterbi derivation

Figure 2: An example of HIERO translation.

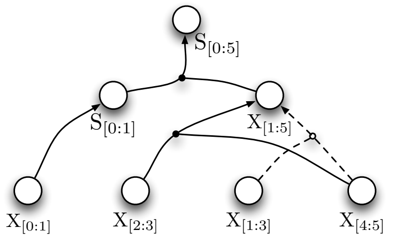


Figure 3: A −LM hypergraph with two derivations: the gold derivation (Fig. 2b) in solid lines, and the Viterbi derivation (Fig. 2c) in dashed lines.

## 2 Review: Syntax-based MT Decoding

For clarity reasons we will describe HIERO decoding as a two-pass process, first without a language model, and then integrating the LM. This section mostly follows Huang and Chiang (2007).

In the first, −LM phase, the decoder parses the source sentence using the source projection of the synchronous grammar (see Fig. 2 (a) for an example), producing a −LM hypergraph where each node has a signature $N_{[i:j]}$, where $N$ is the nonterminal type (either X or S in HIERO) and $[i : j]$ is the span, and each hyperedge $e$ is an application of the translation rule $r(e)$ (see Figure 3).

To incorporate the language model, each node also needs to remember its target side boundary words. Thus a −LM node $N_{[i:j]}$ is split into multiple +LM nodes of signature $N^{a\star b}_{[i:j]}$, where $a$ and $b$ are the boundary words. For example, with a bigram LM, $\mathrm{X}^{\text{held}\star\text{Sharon}}_{[1:5]}$ is a node whose translation starts with "held" and ends with "Sharon".

More formally, the whole decoding process can be cast as a deductive system. Take the partial translation of "held talks with Sharon" in Figure 2 (b) for example, the deduction is

$$\frac{\mathrm{X}^{\text{Sharon}\star\text{Sharon}}_{[2:3]} : s_1 \qquad \mathrm{X}^{\text{talks}\star\text{talks}}_{[4:5]} : s_2}{\mathrm{X}^{\text{held}\star\text{Sharon}}_{[1:5]} : s_1 + s_2 + s(r_5) + \lambda} \quad r_5,$$

where $s(r_5)$ is the score of rule $r_5$, and the LM combo score $\lambda$ is $\log \mathrm{P}_{\mathrm{lm}}(\text{talks} \mid \text{held})\mathrm{P}_{\mathrm{lm}}(\text{with} \mid \text{talks})\mathrm{P}_{\mathrm{lm}}(\text{Sharon} \mid \text{with})$.

## 3 Violation-Fixing Perceptron for HIERO

As mentioned in Section 1, the key to the success of Yu et al. (2013) is the adoption of violation-fixing perceptron of Huang et al. (2012) which is tailored for vastly inexact search. The general idea is to update somewhere in the middle of the search (where search error happens) rather than at the very end (standard update is often invalid). To adapt it to MT where many derivations can output the same translation (i.e., spurious ambiguity), Yu et al. (2013) extends it to handle latent variables which correspond to phrase-based derivations. On the other hand, Zhang et al. (2013) has generalized Huang et al. (2012) from graphs to hypergraphs for bottom-up parsing, which resembles HIERO decoding. So we just need to combine the two generalizing directions (latent variable and hypergraph, see Fig. 1).

### 3.1 Latent Variable Hypergraph Search

The key difference between bottom-up parsing and MT decoding is that in parsing the gold tree for each input sentence is unique, while in MT many derivations can generate the same reference translation. In other words, the gold derivation to update towards is a latent variable.

Here we formally define the latent variable "max-violation" perceptron over a hypergraph for MT training. For a given sentence pair $\langle x, y\rangle$, we denote $H(x)$ as the decoding hypergraph of HIERO without any pruning. We say $D \in H(x)$ if $D$ is a full derivation of decoding $x$, and $D$ can be derived from the hypergraph. Let $good(x, y)$ be the set of $y$-good derivations for $\langle x, y\rangle$:

$$good(x, y) \stackrel{\Delta}{=} \{D \in H(x) \mid e(D) = y\},$$

where $e(D)$ is the translation from derivation $D$. We then define the set of $y$-good partial derivations that cover $x_{[i:j]}$ with root $N_{[i:j]}$ as

$$\begin{aligned} good_{N_{[i:j]}}(x, y) \stackrel{\Delta}{=} \{d \in D \mid D \in good(x, y), \\ root(d) = N_{[i:j]}\} \end{aligned}$$

We further denote the real decoding hypergraph with beam-pruning and cube-pruning as $H'(x)$. The set of $y$-bad derivations is defined as

$$\begin{aligned} bad_{N_{[i:j]}}(x, y) \stackrel{\Delta}{=} \{d \in D \mid D \in H'(x, y), \\ root(d) = N_{[i:j]}, d \notin good_{N_{[i:j]}}(x, y)\}. \end{aligned}$$

Note that the $y$-good derivations are defined over the *unpruned* whole decoding hypergraph, while the $y$-bad derivations are defined over the real decoding hypergraph with pruning.

The max-violation method performs the update where the model score difference between the incorrect Viterbi partial derivation and the best $y$-good partial derivation is maximal, by penalizing the incorrect Viterbi partial derivation and rewarding the $y$-good partial derivation.

More formally, we first find the Viterbi partial derivation $d^-$ and the best $y$-good partial derivation $d^+$ for each $N_{[i:j]}$ group in the pruned +LM hypergraph:

$$d^+_{N_{[i:j]}}(x, y) \stackrel{\Delta}{=} \mathop{\mathbf{argmax}}_{d \in good_{N_{[i:j]}}(x,y)} \mathbf{w} \cdot \mathbf{\Phi}(x, d),$$

$$d^-_{N_{[i:j]}}(x, y) \stackrel{\Delta}{=} \mathop{\mathbf{argmax}}_{d \in bad_{N_{[i:j]}}(x,y)} \mathbf{w} \cdot \mathbf{\Phi}(x, d),$$

where $\mathbf{\Phi}(x, d)$ is the feature vector for derivation $d$. Then it finds the group $N^*_{[i^*:j^*]}$ with the maximal score difference between the Viterbi derivation and the best $y$-good derivation:

$$\begin{aligned} N^*_{[i^*:j^*]} \stackrel{\Delta}{=} &\mathop{\mathbf{argmax}}_{N_{[i:j]}} \\ &\mathbf{w} \cdot \Delta\mathbf{\Phi}(x, d^+_{N_{[i:j]}}(x, y), d^-_{N_{[i:j]}}(x, y)), \end{aligned}$$

and update as follows:

$$\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{\Phi}(x, d^+_{N^*_{[i^*:j^*]}}(x, y), d^-_{N^*_{[i^*:j^*]}}(x, y)),$$

where $\Delta\mathbf{\Phi}(x, d, d') \stackrel{\Delta}{=} \mathbf{\Phi}(x, d) - \mathbf{\Phi}(x, d')$.

## 3.2 Forced Decoding for HIERO

We now describe how to find the gold derivations.[1] Such derivations can be generated in way similar to Yu et al. (2013) by using a language model tailored for forced decoding:

$$\mathrm{P}_{\mathit{forced}}(q \mid p) = \begin{cases} 1 & \text{if } q = p + 1 \\ 0 & \text{otherwise} \end{cases},$$

where $p$ and $q$ are the indices of the boundary words in the reference translation. The +LM node now has signature $N^{p\star q}_{[i:j]}$, where $p$ and $q$ are the indexes of the boundary words. If a boundary word does not occur in the reference, its index is set to $\infty$ so that its language model score will always be $-\infty$; if a boundary word occurs more than once in the reference, its $-$LM node is split into multiple +LM nodes, one for each such index.[2]

We have a similar deductive system for forced decoding. For the previous example, rule $r_5$ in Figure 2 (a) is rewritten as

$$\mathrm{X} \rightarrow \langle \textit{yǔ}\ \mathrm{X}_{\boxed{1}}\ \textit{jǔxíng}\ \mathrm{X}_{\boxed{2}}, 1\ \mathrm{X}_{\boxed{2}}\ 4\ \mathrm{X}_{\boxed{1}}\rangle,$$

where 1 and 4 are the indexes for reference words "held" and "with" respectively. The deduction for $\mathrm{X}_{[1:5]}$ in Figure 2 (b) is

$$\frac{\mathrm{X}^{5\star 5}_{[2:3]} : s_1 \qquad \mathrm{X}^{2\star 3}_{[4:5]} : s_2}{\mathrm{X}^{1\star 5}_{[1:5]} : s(r_5) + \lambda + s_1 + s_2} \quad r_5,$$

where $\lambda = \log \prod_{i \in \{1,3,4\}} \mathrm{P}_{\mathit{forced}}(i + 1 \mid i) = 0$.

[1] We only consider *single* reference in this paper.

[2] Our formulation of index-based language model fixes a bug in the word-based LM of Yu et al. (2013) when a substring appears more than once in the reference (e.g. "the man...the man..."); thanks to Dan Gildea for pointing it out.

## 4 Experiments

Following Yu et al. (2013), we call our max-violation method MAXFORCE. Our implementation is mostly in Python on top of the `cdec` system (Dyer et al., 2010) via the `pycdec` interface (Chahuneau et al., 2012). In addition, we use minibatch parallelization of (Zhao and Huang, 2013) to speedup perceptron training. We evaluate MAXFORCE for HIERO over two CH-EN corpora, IWSLT09 and FBIS, and compare the performance with vanilla $n$-best MERT (Och, 2003) from Moses (Koehn et al., 2007), Hypergraph MERT (Kumar et al., 2009), and PRO (Hopkins and May, 2011) from `cdec`.

### 4.1 Features Design

We use all the 18 *dense* features from `cdec`, including language model, direct translation probability $p(e|f)$, lexical translation probabilities $p_l(e|f)$ and $p_l(f|e)$, length penalty, counts for the source and target sides in the training corpus, and flags for the glue rules and pass-through rules.

For *sparse* features we use Word-Edges features (Charniak and Johnson, 2005; Huang, 2008) which are shown to be extremely effective in both parsing and phrase-based MT (Yu et al., 2013). We find that even simple Word-Edges features boost the performance significantly, and adding complex Word-Edges features from Yu et al. (2013) brings limited improvement and slows down the decoding. So in the following experiments we only use Word-Edges features consisting of combinations of English and Chinese words, and Chinese characters, and do not use word clusters nor word types. For simplicity and efficiency reasons, we also exclude all non-local features.

### 4.2 Datasets and Preprocessing

Our first corpus, IWSLT09, contains ∼30k short sentences collected from spoken language. IWSLT04 is used as development set in MAXFORCE training, and as tuning set for $n$-best MERT, Hypergraph MERT, and PRO. IWSLT05 is used as test set. Both IWSLT04 and IWSLT05 contain 16 references.We mainly use this corpus to investigate the properties of MAXFORCE.

The second corpus, FBIS, contains ∼240k sentences. NIST06 newswire is used as development set for MAXFORCE training, and as tuning set for all other tuning methods. NIST08 newswire is used as test set. Both NIST06 newswire and NIST08 newswire contain 4 references. We mainly use this corpus to demonstrate the performance of MAXFORCE in large-scale training.

For both corpora, we do standard tokenization, alignment and rule extraction using the `cdec` tools. In rule extraction, we remove all 1-count rules but keep the rules mapping from one Chinese word to one English word to help balancing between overfitting and coverage. We use a trigram language model trained from the target sides of the two corpora respectively.

| | sent. | words |
|---|---|---|
| phrase-based MT | 32% | 12% |
| HIERO | 35% | 30% |
| HIERO (all rules) | 65% | 55% |

Table 1: Reachability comparison (on FBIS) between phrase-based MT reported in Yu et al. (2013) (without 1-count rules) and HIERO (with and without 1-count rules).

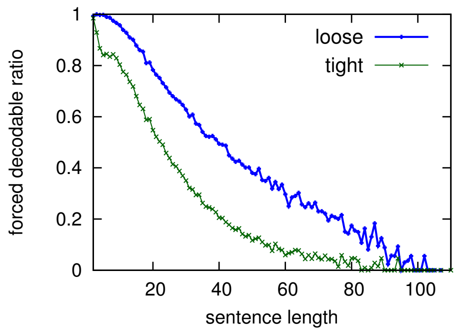


Figure 4: Reachability vs. sent. length for FBIS.

### 4.3 Forced Decoding Reachability

We first report the forced decoding reachability for HIERO on FBIS in Table 1. With the full rule set, 65% sentences and 55% words of the whole corpus are forced decodable in HIERO. After pruning 1-count rules, our forced decoding covers significantly more words than phrase-based MT in Yu et al. (2013). Furthermore, in phrase-based MT, most decodable sentences are very short, while in HIERO the lengths of decodable sentences are more evenly distributed.

However, in the following experiments, due to efficiency considerations, we use the "tight" rule extraction in `cdec` that is more strict than the standard "loose" rule extraction, which generates a reduced rule set and, thus, a reduced reachability. We show the reachability distributions of both tight and loose rule extraction in Figure 4.

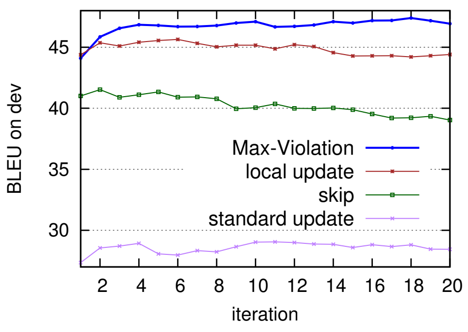


Figure 5: Comparison of various update methods.

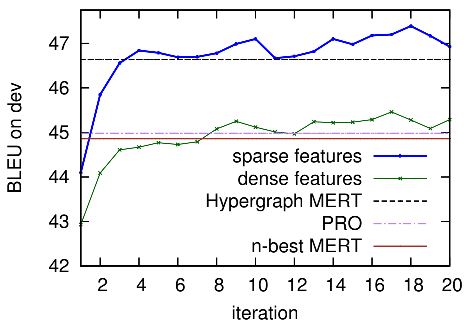


Figure 6: Sparse features (Word-Edges) contribute ∼2 BLEU points, outperforming PRO and MERT.

## 4.4 Evaluation on IWSLT

For IWSLT, we first compare the performance from various update methods in Figure 5.[3] The max-violation method is more than 15 BLEU points better than the standard perceptron (also known as "bold-update" in Liang et al. (2006)) which updates at the root of the derivation tree.[4] This can be explained by the fact that in training ∼58% of the standard updates are invalid (i.e., they do not fix any violation). We also use the "skip" strategy of Zhang et al. (2013) which updates at the root of the derivation only when it fixes a search error, avoiding all invalid updates. This achieves ∼10 BLEU better than the standard update, but is still more than ∼5 BLEU worse than Max-Violation update. Finally we also try the "local-update" method from Liang et al. (2006) which updates towards the derivation with the best Bleu$^{+1}$ in the root group $S_{[0:|x|]}$. This method is about 2 BLEU points worse than max-violation.

We further investigate the contribution of sparse features in Figure 6. On the development set, max-violation update without Word-Edges features achieves BLEU similar to $n$-best MERT and PRO, but lower than Hypergraph MERT. Adding simple Word-Edges features improves BLEU by ∼2 points, outperforming the very strong Hypergraph MERT baseline by ∼1 point. See Table 2 for details. The results of $n$-best MERT, Hypergraph MERT, and PRO are averages from 3 runs.

[3]We find that while MAXFORCE generates translations of length ratio close to 1 during training, the length ratios on dev/test sets are significantly lower, due to OOVs. So we run a binary search for the length penalty weight after each training iteration to tune the length ratio to ∼0.97 on dev set.

[4] We report BLEU with **averaged** reference lengths.

| algorithm | # feats | dev | test |
|---|---|---|---|
| $n$-best MERT | 18 | 44.9 | 47.9 |
| Hypergraph MERT | 18 | 46.6 | 50.7 |
| PRO | 18 | 45.0 | 49.5 |
| local update perc. | 443K | 45.6 | 49.1 |
| MAXFORCE | 529K | **47.4** | **51.5** |

Table 2: BLEU scores (with 16 references) of various training algorithms on IWSLT09.

| algorithm | # feats | dev | test |
|---|---|---|---|
| Hypergraph MERT | 18 | 27.3 | 23.0 |
| PRO | 18 | 26.4 | 22.7 |
| MAXFORCE | 4.5M | **27.7** | **23.9** |

Table 3: BLEU scores (with 4 references) of various training algorithms on FBIS.

## 4.5 Evaluation on FBIS

Table 3 shows BLEU scores of Hypergraph MERT, PRO, and MAXFORCE on FBIS. MAXFORCE actives 4.5M features, and achieves +1.2 BLEU over PRO and +0.9 BLEU over Hypergraph MERT. The training time for Hypergraph MERT and PRO is about 30 min. on the dev set, and is about 5 hours (on 32 cores) for MAXFORCE on the training set.

## 5 Conclusions

We have presented a latent-variable violation-fixing framework for general structured prediction problems with inexact search over hypergraphs. Its application on HIERO brings significant improvement in BLEU, compared to algorithms that are specially designed for MT tuning such as MERT and PRO.

## Acknowledgment

Part of this work was done during K. Z.'s internship at IBM. We thank Martin Čmejrek for guidance, the anonymous reviewers for comments, David Chiang for pointing us to `pycdec`, Dan Gildea for forced decoding (Footnote 2), and Lemao Liu for discussions. This work is supported by DARPA FA8750-13-2-0041 (DEFT), DARPA HR0011-12-C-0015 (BOLT), and a Google Faculty Research Award.

## References

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

Phil Blunsom, Trevor Cohn, and Miles Osborne. 2008. A discriminative latent variable model for statistical machine translation. In *ACL*, pages 200–208.

Victor Chahuneau, Noah Smith, and Chris Dyer. 2012. pycdec: A python interface to cdec. *Prague Bulletin of Mathematical Linguistics*, (98).

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of ACL*, pages 173–180, Ann Arbor, Michigan, June.

David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proceedings of EMNLP 2008*.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL*.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP*.

Chris Dyer, Adam Lopez, Juri Ganitkevitch, Johnathan Weese, Ferhan Ture, Phil Blunsom, Hendra Setiawan, Vladimir Eidelman, and Philip Resnik. 2010. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the ACL*.

Jeffrey Flanigan, Chris Dyer, and Jaime Carbonell. 2013. Large-scale discriminative training for statistical machine translation using held-out line search. In *Proceedings of NAACL 2013*.

Spence Green, Sida Wang, Daniel Cer, and Christopher D Manning. 2013. Fast and adaptive online training of feature-rich translation models. *to appear) ACL*.

Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of EMNLP*.

Liang Huang and David Chiang. 2007. Forest rescoring: Fast decoding with integrated language models. In *Proceedings of ACL*, Prague, Czech Rep., June.

Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proceedings of NAACL*.

Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of the ACL: HLT*, Columbus, OH, June.

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. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of ACL*.

Shankar Kumar, Wolfgang Macherey, Chris Dyer, and Franz Och. 2009. Efficient minimum error rate training and minimum bayes-risk decoding for translation hypergraphs and lattices. In *Proceedings of the Joint Conference of ACL and AFNLP*.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of ACL*.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of COLING-ACL*, Sydney, Australia, July.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd ACL*.

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

Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proceedings of EMNLP-CoNLL*.

Heng Yu, Liang Huang, Haitao Mi, and Kai Zhao. 2013. Max-violation perceptron and forced decoding for scalable MT training. In *Proceedings of EMNLP*.

Hao Zhang, Liang Huang, Kai Zhao, and Ryan McDonald. 2013. Online learning with inexact hypergraph search. In *Proceedings of EMNLP*.

Kai Zhao and Liang Huang. 2013. Minibatch and parallelization for online large margin structured learning. In *Proceedings of NAACL 2013*.