Max-Margin Synchronous Grammar Induction for Machine Translation

Xinyan Xiao and **Deyi Xiong**[*]
School of Computer Science and Technology
Soochow University
Suzhou 215006, China
xyxiao.cn@gmail.com, dyxiong@suda.edu.cn

Abstract

Traditional synchronous grammar induction estimates parameters by maximizing likelihood, which only has a loose relation to translation quality. Alternatively, we propose a max-margin estimation approach to discriminatively inducing synchronous grammars for machine translation, which directly optimizes translation quality measured by BLEU. In the max-margin estimation of parameters, we only need to calculate Viterbi translations. This further facilitates the incorporation of various non-local features that are defined on the target side. We test the effectiveness of our max-margin estimation framework on a competitive hierarchical phrase-based system. Experiments show that our max-margin method significantly outperforms the traditional two-step pipeline for synchronous rule extraction by 1.3 BLEU points and is also better than previous max-likelihood estimation method.

1 Introduction

Synchronous grammar induction, which refers to the process of learning translation rules from bilingual corpus, still remains an open problem in statistical machine translation (SMT). Although state-of-the-art SMT systems model the translation process based on synchronous grammars (including bilingual phrases), most of them still learn translation rules via a pipeline with word-based heuristics (Koehn et al., 2003). This pipeline first builds word alignments using heuristic combination strategies, then heuristically extracts rules that are consistent with word alignments. Such heuristic pipeline is not elegant theoretically. It brings an undesirable gap that separates modeling and learning in an SMT system.

Therefore, researchers have proposed alternative approaches to learning synchronous grammars directly from sentence pairs without word alignments, via generative models (Marcu and Wong, 2002; Cherry and Lin, 2007; Zhang et al., 2008; DeNero et al., 2008; Blunsom et al., 2009; Cohn and Blunsom, 2009; Neubig et al., 2011; Levenberg et al., 2012) or discriminative models (Xiao et al., 2012). Theoretically, these approaches describe how sentence pairs are generated by applying sequences of synchronous rules in an elegant way. However, they learn synchronous grammars by maximizing likelihood,[1] which only has a loose relation to translation quality (He and Deng, 2012). Moreover, generative models are normally hard to be extended to incorporate useful features, and the discriminative synchronous grammar induction model proposed by Xiao et al. (2012) only incorporates *local* features defined on parse trees of the source language. *Non-local* features, which encode information from parse trees of the target language, have never been exploited before due to the computational complexity of normalization in max-likelihood estimation.

Consequently, we would like to learn synchronous grammars in a discriminative way that can directly maximize the end-to-end translation quality measured by BLEU (Papineni et al., 2002), and is also able to incorporate non-local features from target parse trees.

We thus propose a max-margin estimation method

[*] Corresponding author

[1] More precisely, the discriminative model by Xiao et al. (2012) maximizes conditional likelihood.

Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 255–264,
Seattle, Washington, USA, 18-21 October 2013. ©2013 Association for Computational Linguistics

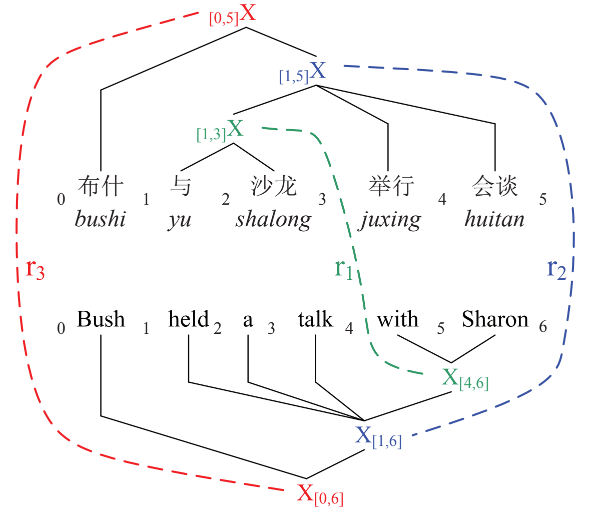
to discriminatively induce synchronous grammar directly from sentence pairs without word alignments. We try to maximize the margin between a reference translation and a candidate translation with translation errors that are measured by BLEU. The more serious the translation errors, the larger the margin. In this way, our max-margin method is able to learn synchronous grammars according to their translation performance. We further incorporate various *non-local* features defined on target parse trees. We efficiently calculate the non-local feature values of a translation over its exponential derivation space using the inside-outside algorithm. Because our max-margin estimation optimizes feature weights only by the feature values of Viterbi and reference translations, we are able to efficiently perform optimization even with non-local features.

We apply the proposed max-margin estimation method to learn synchronous grammars for a hierarchical phrase-based translation system (Chiang, 2007) which typically produces state-of-the-art performance. With non-local features defined on target parse trees, our max-margin method significantly outperforms the baseline that uses synchronous rules learned from the traditional pipeline by 1.3 BLEU points on large-scale Chinese-English bilingual training data.

The remainder of this paper is organized as follows. Section 2 presents the discriminative synchronous grammar induction model with the non-local features. In Section 3, we elaborate our max-margin estimation method which is able to directly optimize BLEU, and discuss how we induce grammar rules. Local and non-local features are described in Section 4. Finally, in Section 5, we verify the effectiveness of our method through experiments by comparing it against both the traditional pipeline and max-likelihood estimation method.

2 Discriminative Model with Non-local Features

Let $\mathcal{S}$ denotes the set of all strings in a source language. Given a source sentence $\mathbf{s} \in \mathcal{S}$, $\mathcal{T}(s)$ denotes all candidate translations in the target language that can be generated by a synchronous grammar $\mathbf{G}$. A translation $t \in \mathcal{T}(s)$ is generated by a sequence of translation steps $(r_1, ..., r_n)$, where we apply a synchronous rule $r \in \mathbf{G}$ in one step. We refer to such a sequence of translation steps as a **derivation** (See Figure 1) and denote it as $\mathbf{d} \in \mathcal{D}(\mathbf{s})$, where $\mathcal{D}(\mathbf{s})$ represents the derivation space of a source sentence. Given an input source sentence $\mathbf{s}$, we output a pair $\langle \mathbf{t}, \mathbf{d} \rangle$ in SMT. Thus, we study the triple $\langle \mathbf{s}, \mathbf{t}, \mathbf{d} \rangle$ in SMT.


[0,5]X
[1,5]X
[1,3]X
0 布什 1 与 2 沙龙 3 举行 4 会谈 5
bushi
yu
shalong
juxing
huitan
r3
r1
r2
0 Bush 1 held 2 a 3 talk 4 with 5 Sharon 6
X[4,6]
X[1,6]
X[0,6]

r_1: $\langle$ *yu shalong* ⇒ with Sharon $\rangle$
r_2: $\langle$ *X juxing huitan* ⇒ *held a talk X* $\rangle$
r_3: $\langle$ *bushi X* ⇒ Bush *X* $\rangle$

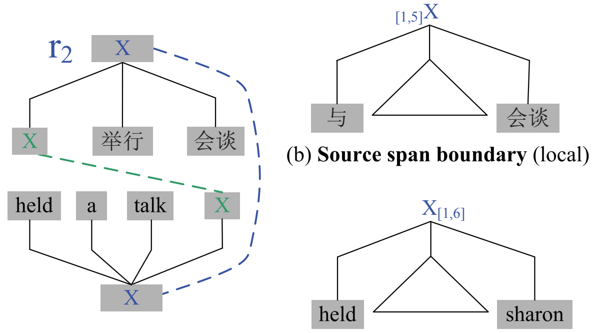
Figure 1: A derivation of a sentence pair represented by a synchronous tree. The above and below part are the parses in the source language side and the target language side respectively. Left subscript of a node X denotes the source span, while right subscript denotes the target span. A dashed line denotes an alignment from a source span to a target span. The annotation for a dashed line corresponds to the rewriting rule used in the corresponding step of the derivation.

In our discriminative model, we calculate the value of a triple $\langle \mathbf{s}, \mathbf{t}, \mathbf{d} \rangle$ according to the following **scoring function**:

$$f(\mathbf{s}, \mathbf{t}, \mathbf{d}) = \theta^T \Phi(\mathbf{s}, \mathbf{t}, \mathbf{d}) \qquad (1)$$

where $\theta \in \mathbf{\Theta}$ is a feature weight vector, and Φ is the **feature function**.

There are exponential outputs in SMT. Therefore it is necessary to factorize the feature function in order to perform efficient calculation over the SMT output space using dynamic programming. We decompose the feature function of a triple $\langle \mathbf{s}, \mathbf{t}, \mathbf{d} \rangle$ into



(a) **Rule** (local)

(b) **Source span boundary** (local)

(c) **Target span boundary** (non-local)

Figure 2: Example features for the derivation in Figure 1. Shaded nodes denote information encoded in the feature.

a sum of values of each synchronous rule in the derivation $\mathbf{d}$.

$$\Phi(\mathbf{s},\mathbf{t},\mathbf{d}) = \sum_{r\in\mathbf{d}} \underbrace{\phi(r,\mathbf{s})}_{\textbf{local}} + \sum_{r\in\mathbf{d}} \underbrace{\phi(r,\mathbf{s},\mathbf{t})}_{\textbf{non-local}} \qquad (2)$$

Our feature functions include both local and non-local features. A feature is a **local** feature if and only if it can be factored among the translation steps in a derivation. In other words, the value of a local feature for $\langle\mathbf{s},\mathbf{t},\mathbf{d}\rangle$ can be calculated as a sum of local scores in each translation step, and the calculation of each local score only requires to look at the rule used in corresponding step and the input sentence. Otherwise, the feature is a **non-local** feature. Our discriminative model allows to incorporate non-local features that are defined on target translations.

For example, a rule feature in Figure 2(a), which indicates the application of a specific rule in a derivation, is a local feature. A source span boundary feature in Figure 2(b) that is defined on the source parse tree is also a local feature. However, a target span boundary feature in Figure 2(c), which assesses the target parse structure, is a non-local feature. According to Figure 1, the span is parsed in step r_2, but it also depends on the translation boundary word "held" generated in previous step r_1. We will describe the details of both local and non-local features that we use in Section 4.

Non-local features enable us to model the target parse structure in a derivation. However, it is computationally expensive to calculate the expected values of non-local features over $\mathcal{D}(\mathbf{s})$, as non-local features require to record states of target boundary words and result in an extremely large number of states during dynamic programming. Fortunately, when integrating out derivations over the derivation space $\mathcal{D}(\mathbf{s},\mathbf{t})$ of a source sentence and its translation, we can efficiently calculate the non-local features. Because all derivations in $\mathcal{D}(\mathbf{s},\mathbf{t})$ share the same translation, there is no need to maintain states for target boundary words. We will discuss this computational problem in details in Section 3.3. In the proposed max-margin estimation described in next section, we only need to integrate out derivation for a Viterbi translation and a reference translation when updating feature weights. Therefore, the defined non-local features allow us to not only explore useful knowledge on the target parse trees, but also compute them efficiently over $\mathcal{D}(\mathbf{s},\mathbf{t})$ during max-margin estimation.

$\mathbf{s}$, $\mathbf{S}$, $\mathcal{S}$	$\mathbf{s}$ is a sentence in a source language; $\mathbf{S}$ means source training sentences; $\mathcal{S}$ denotes all the possible sentences;
$\mathbf{t}$, $\mathbf{T}$, $\mathcal{T}$	symbols for the target language that similar to $\mathbf{s}$, $\mathbf{S}$, $\mathcal{S}$;
$\mathbf{d}$, $\mathcal{D}$	derivation and derivation space;
$\mathcal{D}(\mathbf{s})$	space of derivations for a source sentence;
$\mathcal{D}(\mathbf{s},\mathbf{t})$	space of derivations for a source sentence with its translation;
$\mathcal{H}(\mathbf{s})$	hypergraph that represents $\mathcal{D}(\mathbf{s})$;
$\mathcal{H}(\mathbf{s},\mathbf{t})$	hypergraph that represents $\mathcal{D}(\mathbf{s},\mathbf{t})$;

Table 1: Notations in this paper. We give an abstract of related notations for clarity.

3 Max-Margin Estimation

In this section, we describe how we use a parallel training corpus $\{\mathbf{S},\mathbf{T}\} = \{(\mathbf{s}^{(i)},\mathbf{t}^{(i)})\}_{i=1}^{N}$ to estimate feature weights θ, which contain parameters of the induced synchronous grammars and the defined non-local features.

We choose the parameters that maximize the translation quality measured by BLEU using the *max-margin estimation* (Taskar et al., 2004). Margin refers to the difference of the model score between a reference translation $\mathbf{t}^{(i)}$ and a candidate translation $\mathbf{t}$. We hope that the worse the translation quality of $\mathbf{t}$, the larger the margin between $\mathbf{t}$ and $\mathbf{t}^{(i)}$. In this way, we penalize larger translation errors more severely than smaller ones. This intuition is expressed by the following equation.

$$\begin{aligned} &\min\ \frac{1}{2}\|\theta\|^2 \\ &s.t.\ f(\mathbf{s}^{(i)},\mathbf{t}^{(i)}) - f(\mathbf{s}^{(i)},\mathbf{t}) \geq cost(\mathbf{t}^{(i)},\mathbf{t}) \\ &\quad \forall \mathbf{t} \in \mathcal{T}(\mathbf{s}^{(i)}) \end{aligned} \tag{3}$$

Here, $f(\mathbf{s},\mathbf{t})$ is the feature function of a translation, and **cost function** $cost(\mathbf{t}^{(i)},\mathbf{t})$ measures the translation errors of a candidate translation $\mathbf{t}$ comparing with a reference translation $\mathbf{t}^{(i)}$. We define the cost function via the widely-used translation evaluation metric BLEU. We use the smoothed sentence level BLEU-4 (Lin and Och, 2004) here:

$$cost(\mathbf{t}^{(i)},\mathbf{t}) = 1 - \text{BLEU-4}(\mathbf{t}^{(\mathrm{i})},\mathbf{t}) \tag{4}$$

In Section 3.1, we will discuss how we use the scoring function $f(\mathbf{s},\mathbf{t},\mathbf{d})$ to calculate $f(\mathbf{s},\mathbf{t})$. Then in Section 3.2, we recast the equation (3) as an unconstrained empirical loss minimization problem, and describe the learning algorithm for optimizing θ and inducing $\mathbf{G}$. Finally, we give the details of inference for the learning algorithm in Section 3.3.

3.1 Integrate Out Derivation by Averaging

Although we only model the triple $\langle \mathbf{s},\mathbf{t},\mathbf{d}\rangle$ in the equation (1), it's necessary to calculate the scoring function $f(\mathbf{s},\mathbf{t})$ of a translation by integrating out the variable of derivation as derivation is not observed in the training data.

We use an averaging computation over all possible derivations of a translation $\mathcal{D}(\mathbf{s},\mathbf{t})$. We call this an average derivation based estimation:

$$f(\mathbf{s},\mathbf{t}) = \frac{1}{|\mathcal{D}(\mathbf{s},\mathbf{t})|} \sum_{\mathbf{d}\in\mathcal{D}(\mathbf{s},\mathbf{t})} f(\mathbf{s},\mathbf{t},\mathbf{d}) \tag{5}$$

The "average derivation" can be considered as the geometric central point in the space $\mathcal{D}(\mathbf{s},\mathbf{t})$.

Another possible way to deal with the latent derivation is max-derivation, which uses the max-operator over $\mathcal{D}(\mathbf{s},\mathbf{t})$. The max derivation method sets $f(\mathbf{s},\mathbf{t})$ as $\max_{\mathbf{d}\in\mathcal{D}(\mathbf{s},\mathbf{t})} f(\mathbf{s},\mathbf{t},\mathbf{d})$. It is often adopted in traditional SMT systems. Nevertheless, we instead use average-derivation for two reasons.[2] First, as a translation has an exponential number of derivations, finding the max derivation of a reference translation for learning is nontrivial (Chiang et al., 2009). Second, the max derivation estimation will result in a low rule coverage, as rules in a max derivation only covers a small fraction of rules in the $\mathcal{D}(\mathbf{s},\mathbf{t})$. Because rule coverage is important in synchronous grammar induction, we would like to explore the entire derivation space using the average operator.

3.2 Learning Algorithm

We reformulate the equation (3) as an unconstrained empirical loss minimization problem as follows:

$$\min \frac{\lambda}{2}\|\theta\|^2 + \frac{1}{N}\sum_{n=1}^{N} L(\mathbf{s}^{(i)},\mathbf{t}^{(i)},\theta) \tag{6}$$

Where λ denotes the regularization strength for L2-norm. The loss function of a sentence pair $L(\mathbf{s}^{(i)},\mathbf{t}^{(i)},\theta)$ is a convex hinge loss function denoted by:

$$\begin{aligned} &\max\{0, -f(\mathbf{s}^{(i)},\mathbf{t}^{(i)}) \\ &\quad + \max_{\mathbf{t}\in\mathcal{T}(\mathbf{s}^{(i)})} \Big(f(\mathbf{s}^{(i)},\mathbf{t}) + cost(\mathbf{t}^{(i)},\mathbf{t})\Big)\} \end{aligned} \tag{7}$$

According to the second max-operator in the hinge loss function, the optimization towards BLEU is expressed by **cost-augmented** inference. Cost-augmented inference finds a translation that has a maximum model score augmented with cost.

$$\hat{\mathbf{t}} = \max_{\mathbf{t}\in\mathcal{T}(\mathbf{s}^{(i)})} \Big(f(\mathbf{s}^{(i)},\mathbf{t}) + cost(\mathbf{t}^{(i)},\mathbf{t})\Big) \tag{8}$$

We applied the Pegasos algorithm for the optimization of equation (6) (Shalev-Shwartz et al., 2007). This is an online algorithm, which alternates between stochastic gradient descent steps and projection steps. When the loss function is non-zero, it updates weights according to the sub-gradient of the hinge loss function. Using the average scoring function in the equation (5), the sub-gradient of hinge loss function for a sentence pair is the difference of average feature values between a Viterbi translation

[2] Imagine that $\mathcal{H}(\mathbf{s},\mathbf{t})$ in the Algorithm 1 is replaced by a maximum derivation in $\mathcal{H}(\mathbf{s},\mathbf{t})$.

```
Algorithm 1 UPDATE($\mathbf{s}, \mathbf{t}, \theta, \mathbf{G}$)        ▷ One step in online algorithm. $\mathbf{s}, \mathbf{t}$ are short for $\mathbf{s}^{(i)}$, $\mathbf{t}^{(i)}$ here
1: $\mathcal{H}(\mathbf{s}, \mathbf{t}) \leftarrow$ BIPARSE($\mathbf{s}, \mathbf{t}, \theta$)        ▷ Build hypergraph of reference translation
2: $\mathbf{G} \leftarrow \mathbf{G} + \mathcal{H}(\mathbf{s}, \mathbf{t})$        ▷ Discover rules from $\mathcal{H}(\mathbf{s}, \mathbf{t})$
3: $\hat{\mathbf{t}}, \hat{\mathbf{d}} \leftarrow \arg\max_{\langle \mathbf{t}', \mathbf{d}' \rangle \in \mathcal{D}(\mathbf{s})} f(\mathbf{s}, \mathbf{t}', \mathbf{d}') + cost(\mathbf{t}, \mathbf{t}')$        ▷ Find Viterbi translation
4: $\mathcal{H}(\mathbf{s}, \hat{\mathbf{t}}) \leftarrow$ BIPARSE($\mathbf{s}, \hat{\mathbf{t}}, \theta$)        ▷ Build hypergraph of Viterbi translation
5: if $f(\mathbf{s}, \mathbf{t}) < f(\mathbf{s}, \hat{\mathbf{t}}) + cost(\mathbf{t}, \hat{\mathbf{t}})$ then
6:     $\theta \leftarrow (1 - \eta\lambda)\theta + \eta \times \frac{\partial \mathbf{L}}{\partial \theta}(\mathcal{H}(\mathbf{s}, \mathbf{t}), \mathcal{H}(\mathbf{s}, \hat{\mathbf{t}}))$        ▷ Update $\theta$ by gradient $\frac{\partial \mathbf{L}}{\partial \theta}$ and learning rate $\eta$
7:     $\theta \leftarrow \min\{1, \frac{1/\sqrt{\lambda}}{\|\theta\|}\} \times \theta$        ▷ Projection by scaling
8: return $\mathbf{G}, \theta$
```

and a reference translation.

$$\frac{\partial \mathbf{L}}{\partial \theta} = \frac{1}{|\mathcal{D}(\mathbf{s}^{(i)}, \mathbf{t}^{(i)})|} \sum_{\mathbf{d} \in \mathcal{D}(\mathbf{s}^{(i)}, \mathbf{t}^{(i)})} \Phi(\mathbf{s}^{(i)}, \mathbf{t}^{(i)}, \mathbf{d}) - \frac{1}{|\mathcal{D}(\mathbf{s}^{(i)}, \hat{\mathbf{t}})|} \sum_{\mathbf{d} \in \mathcal{D}(\mathbf{s}^{(i)}, \hat{\mathbf{t}})} \Phi(\mathbf{s}^{(i)}, \hat{\mathbf{t}}, \mathbf{d}) \qquad (9)$$

Algorithm 1 shows the procedure of one step in the online optimization algorithm. The procedure discovers rules and updates weights in an online fashion. In the procedure, we first biparse the sentence pair to construct a synchronous hypergraph of a reference translation (line 1). In the biparsing algorithm, synchronous rules for constructing hyperedges are not required to be in $\mathbf{G}$, but can be any rules that follow the form defined in Chiang (2007). Thus, the biparsing algorithm can discover new rules that are not in $\mathbf{G}$. Then we collect the translation rules discovered in the hypergraph of the reference translation (line 2), which are rules indicated by hyperedges in the hypergraph. We then calculate the Viterbi translation according to the scoring function and cost function (see Section 3.3) (line 3), and build the synchronous hypergraph for the Viterbi translation (line 4). Finally, we update weights according to the Pegasos algorithm (line 5). The sub-gradient is calculated based on the hypergraph of Viterbi translation and reference translation.

In practice, in order to process the data in a parallel manner, we use a larger step size of 1000 for the learning algorithm. In each step of our online optimization algorithm, we first biparse 1000 reference sentence pairs in parallel. Then, we collect grammar rules from the generated reference hypergraphs. After that, we compute the gradients of 1000 sentence pairs in parallel, by calculating feature weights over reference hypergraphs and Viterbi hypergraphs. Finally, we update the feature weights using the sum of these gradients.

3.3 Inference

There are two parts that need to be calculated in the learning algorithm: finding a cost-augmented Viterbi translation according to the scoring function and cost function (Equation 8), and constructing synchronous hypergraphs for the Viterbi and reference translation so as to discover rules and calculate average feature values in Equation (9). Following the traditional decoding procedure, we resort to the cube-pruning based algorithm for approximation.

To find the Viterbi translation, we run the traditional translation decoding algorithm (Chiang, 2007) to get the best derivation. Then we use the translation yielded by the best derivation as the Viterbi translation. In order to obtain the BLEU score in the cost function, we need to calculate the ngram precision. It is calculated in a way similar to the calculation of the ngram language model. The computation of BLEU-4 requires to record 3 boundary words in both the left and right side during dynamic programming. Therefore, even when we use a language model whose order is less than 4, we still expands the states to record 3 boundary words so as to calculate the cost measured by BLEU.

We build synchronous hypergraphs using the cube-pruning based biparsing algorithm (Xiao et al., 2012). Algorithm 2 shows the procedure. Using a chart, the biparsing algorithm constructs k-best alignments for every source word (lines 1-5) and k-best hyperedges for every source span (lines 6-13) from the bottom up. Thus, a synchronous hypergraph is generated during the construction of the chart. More specifically, for a source span, it first creates cubes $\mathcal{L}$ for all source parses γ that are in-

Algorithm 2 BiParse($\mathbf{s}, \mathbf{t}, \theta$) ▷ (Xiao et al., 2012)

```
    ▷ Create k-best alignments for each source word
 1: for $i \leftarrow 1, .., |\mathbf{s}|$ do
 2:     for $j \leftarrow 1, .., |\mathbf{t}|$ do
 3:         $L_j \leftarrow \{\varepsilon, t_j\}$            ▷ $s_i$ aligns to $t_j$ or not
 4:     $\mathcal{L} \leftarrow \langle L_1, ..., L_{|t|} \rangle$
 5:     $chart[s, i] \leftarrow$ KBest($\mathcal{L}$,$\otimes$,$\theta$)
    ▷ Create k-best hyperedges for each source span
 6: $\mathcal{H} \leftarrow \emptyset$
 7: for $h \leftarrow 1, .., |\mathbf{s}|$ do               ▷ $h$ is the size of span
 8:     for all $i, j$ s.t. $j - i = h$ do
 9:         $\mathcal{L} \leftarrow \emptyset$
10:         for $\gamma$ inferable from $chart$ do
11:             $\mathcal{L} \leftarrow \mathcal{L} + \langle chart[\gamma_1], ..., chart[\gamma_{|\gamma|}] \rangle$
12:         $chart[X, i, j] \leftarrow$ KBest($\mathcal{L}$,$\otimes$,$\theta$)
13:         $\mathcal{H} \leftarrow \mathcal{H} + chart[X, i, j]$   ▷ save hyperedges
14: return $\mathcal{H}$
```

ferable from the chart (lines 9-11). Here γ_i is a partial source parse that covers either a single source word or a span of source words. Then it uses the cube pruning algorithm to keep the top k derivations among all partial derivations that share the same source span $[i, j]$ (line 12). Notably, this biparsing algorithm does not require specific translation rules as input. Instead, it is able to discover new synchronous grammar rules when constructing a synchronous hypergraph: extracting each hyperedge in the hypergraph as a synchronous rule.

Based on the biparsing algorithm, we are able to construct the reference hypergraph $\mathcal{H}(\mathbf{s}^{(i)}, \mathbf{t}^{(i)})$ and Viterbi hypergraph $\mathcal{H}(\mathbf{s}^{(i)}, \hat{\mathbf{t}})$. By the reference hypergraph, we collect new synchronous translation rules and record them in the grammar **G**. We also calculate the average feature values of hypergraphs using the inside-outside algorithm (Li et al., 2009), so as to compute the gradients.

4 Features

One advantage of the discriminative method is that it enables us to incorporate arbitrary features. As shown in Section 2, our model incorporates both local and non-local features.

4.1 Local Features

Rule features We associate each rule with an indicator feature. Each indicator feature counts the number of times that a rule appears in a derivation. In this way, we are able to learn a weight for every rule according to the entire structure of sentence.

Word association features Lexicalized features are widely used in traditional SMT systems. Here we adopt two lexical weights called noisy-or features (Zens and Ney, 2004). The noisy-or feature is estimated by word translation probabilities output by GIZA++. We set the initial weight of these two lexical scores with equivalent positive values. The lexical weights enable our system to score and rank the hyperedges at the beginning. Although word alignment features are used, we do not constrain the derivation space of a sentence pair by prefixed word alignment, and do not require any heuristic alignment combination strategy.

Length feature We integrate the length of target translation that is used in traditional SMT system as our feature.

Source span boundary features We use this kind of feature to assess the source parse tree in a derivation. Previous work (Xiong et al., 2010) has shown the importance of phrase boundary features for translation. Actually, this kind of feature is a good cue for deciding the boundary where a rule is to be learnt. Following Taskar et al. (2004), for a bispan $[i, j, k, l]$ in a derivation, we define the feature templates that indicates the boundaries of a span by its beginning and end words: $\{B : s_{i+1}; E : s_j; BE : s_{i+1}, s_j\}$.

Source span orientation features Orientation features are only used for those spans that are swapping. In Figure 1, the translation of source span $[1, 3]$ is swapping with that of span $[4, 5]$ by r_2, thus orientation feature for span $[1, 3]$ is activated. We also define three feature templates for a swapping span similar to the boundary features: $\{B : s_{i+1}; E : s_j; BE : s_{i+1}, s_j\}$. In practice, we add a prefix to the orientation features so as to distinguish these features from the boundary features.

4.2 Non-local Features

Target span boundary features We also want to assess the target tree structure in a derivation. We define these features in a way similar to source span boundary features. For a bispan $[i, j, k, l]$ in a derivation, we define the feature templates that indicates

System	Grammar Size	MT03	MT04	MT05	Avg.
Moses	302.5M	34.26	36.56	32.69	34.50
Baseline	77.8M	33.83	35.81	33.23	34.29
Max-margin	59.4M	34.62	37.14	34.00	35.25
+Sparse feature		35.48	37.31	34.07	35.62

Table 2: Experiment results. Baseline is an in-house implementation of hierarchical phrase based system. *Moses* denotes the implementation of hierarchical phrased-model in Moses (Koehn et al., 2007). $+Sparsefeature$ means that those sparse features used in the grammar induction are also used during decoding. The improvement of max-margin over Baseline is statistically significant ($p < 0.01$).

target span boundary as: $\{B : t_{k+1}; E : t_l; BE : t_{k+1}, t_l\}$.

Target span orientation features Similar target orientation features are used for a swapping span $[i, j, k, l]$ with feature templates $\{B : t_{k+1}; E : t_l; BE : t_{k+1}, t_l\}$.

Relative position features Following Blunsom and Cohn (Blunsom and Cohn, 2006), we integrate features indicating the closeness to the alignment matrix diagonal. For an aligned word pair with source position i and target position j, the value of this feature is $|\frac{i}{|\mathbf{s}|} - \frac{j}{|\mathbf{t}|}|$. As this feature depends on the length of the target sentence, it is a non-local feature.

Language model We also incorporate an ngram language model which is an important component in SMT. For efficiency, we use a 3-gram language model trained on the target side of our training data during the induction of synchronous grammars.

5 Experiment

In this section, we present our experiments on the NIST Chinese-to-English translation tasks. We first compare our max-margin based method with the traditional pipeline on a large bitext which contains 1.1 million sentences. We then present a detailed comparison on a smaller dataset, in order to analyze the effectiveness of max-margin estimation comparing with the max likelihood estimation (Xiao et al., 2012), and also the effectiveness of the non-local features that are defined on the target side.

5.1 Setup

The baseline system is the hierarchical phrase based system (Chiang, 2007). We used a bilingual corpus that contains 1.1M sentences (44.6 million words) of up to length 40 from the LDC data.[3] Our 5-gram language model was trained by SRILM toolkit (Stolcke, 2002). The monolingual training data includes the Xinhua section of the English Gigaword corpus and the English side of the entire LDC data (432 million words).

We used the NIST 2002 (MT02) as our development set, and the NIST 2003-2005 (MT03-05) as the test set. Case-insensitive NIST BLEU-4 (Papineni et al., 2002) is used to measure translation performance, and also the cost function in the max-margin estimation. Statistical significance in BLEU differences was tested by paired bootstrap re-sampling (Koehn, 2004). We used minimum error rate training (MERT) (Och, 2003) to optimize feature weights for the traditional log-linear model.

We used the same decoder as the baseline system in all estimation methods. Without special explanation, we used the same features as those in the traditional pipeline: forward and backward translation probabilities, forward and backward lexical weights, count of extracted rules, count of glue rules, length of translation, and language model. For the lexical weights we used the noisy-or in all configurations including the baseline system. For the discriminative grammar induction, rule translation probabilities were calculated using the expectations of rules in the synchronous hypergraphs of sentence pairs.

As our max-margin synchronous grammar induction is trained on the entire bitext, it is necessary to load all the rules into the memory during training. To control the size of rule table, we used Viterbi-

[3] Including LDC2002E18, LDC2003E07, LDC2003E14, LDC2004T07, LDC2005T06 and Hansards portion of LDC2004T08.

System	Feature Function	MT03	MT04	MT05	Avg.
Baseline	—	31.76	33.08	31.06	31.96
Max-likelihood	local	32.84	34.54	31.61	33.00
Max-margin	local	32.97	34.92	31.99	33.29
	local,non-local	33.27	34.83	32.32	33.47

Table 3: Comparison of Max-margin and Max-likelihood estimation on a smaller corpus. For max-margin method, we present two results according to the usages of non-local features. The max-margin with non-local features significantly outperforms the Baseline ($p < 0.01$) and also the max-likelihood estimation ($p < 0.05$).

pruning (Huang, 2008) when collecting rules as shown in line 2 of optimization procedure in Section 3.2. Furthermore, we aggressively discarded those large rules (The number of source symbols or the number of target symbols are more than two) that occur only in one sentence. Whenever the learning algorithm processes 50K sentences, we performed this discarding operation for large rules.

5.2 Result on Large Dataset

Table 2 shows the translation results. Our method induces 59.4 million synchronous rules, which are 76.3% of the grammar size of baseline. Note that Moses allows the boundary words of a phrase to be unaligned, while our baseline constraints the initial phrase to be tightly consistent with word alignment. Therefore, Moses extract a much larger rule table than that of our baseline.

With fewer translation rules, our method obtains an average improvement of +0.96 BLEU points on the three test sets over the Baseline. As the difference between the baseline and our max-margin synchronous grammar induction model only lies in the grammar, this result clearly denotes that our learnt grammar does outperform the grammar extracted by the traditional two-step pipeline.

We also incorporate the sparse features during decoding in a way similar to Xiao et al. (2012) and Dyer et al. (2011). In order to optimize these sparse features with the dense features by MERT, we group features of the same type into one coarse "summary feature", and get three such features including: rule, phrase-boundary and phrase orientation features. In this way, we rescale the weights of the three "summary features" with the 8 dense features by MERT. We achieve a further improvement of +0.37 BLEU points. Therefore, our training algorithm is able to learn the useful information encoded by the sparse features for translation.

5.3 Comparison of Estimation Objective and Non-Local Feature

We want to investigate whether the max-margin estimation is able to outperform the max-likelihood estimation method (Xiao et al., 2012). Therefore we carried out experiments to compare them directly. As the max-margin method is able to use non-local features, we compare two settings of features for the max-margin method. One uses only local features, the other uses both local and non-local features. Because the training procedure need to run on the entire corpus, which is time consuming, we therefore use a smaller corpus containing 50K sentences from the entire bitext for comparison.

Table 3 shows the results. When using only local features, the max-margin method consistently outperforms the max-likelihood method in all three test sets. This clearly shows the advantage of learning grammars by optimizing BLEU over likelihood.

When incorporating the non-local features into the max-margin method, we achieve further improvement against the max-margin method without non-local features. With non-local features, our max-margin estimation method outperforms the baseline by 1.5 BLEU points, and is better than the max-likelihood estimation by 0.5 BLEU points. Based on these results, we believe that non-local features, which encode information from target parse structures, are helpful for grammar induction. This further confirms the advance of the max-margin estimation, as it provides us a convenient way to use non-local features.

6 Related Work

As the synchronous grammar is the key component in SMT systems, researchers have proposed various methods to improve the quality of grammars. In addition to the generative and discriminative models introduced in Section 1, researchers also have made efforts on word alignment and grammar weight rescoring.

The first line is to modify word alignment by exploring information of syntactic structures (May and Knight, 2007; DeNero and Klein, 2010; Pauls et al., 2010; Burkett et al., 2010; Riesa et al., 2011). Such syntactic information is combined with word alignment via a discriminative framework. These methods prefer word alignments that are consistent with syntactic structure alignments. However, labeled word alignment data are required in order to learn the discriminative model.

Yet another line is to rescore the weights of translation rules. This line of work tries to improve the relative frequency estimation used in the traditional pipeline. They rescore the weights or probabilities of extracted rules. The rescoring is done by using the similar latent log-linear model as ours (Blunsom et al., 2008; Kääriäinen, 2009; He and Deng, 2012), or incorporating various features using labeled word aligned bilingual data (Huang and Xiang, 2010). However, in rescoring, translation rules are still extracted by the heuristic two-step pipeline. Therefore these previous work still suffers from the inelegance problem of the traditional pipeline.

Our work also relates to the discriminative training (Och, 2003; Watanabe et al., 2007; Chiang et al., 2009; Xiao et al., 2011; Gimpel and Smith, 2012) that has been widely used in SMT systems. Notably, these discriminative training methods are not used to learn grammar. Instead, they assume that grammar are extracted by the traditional two-step pipeline.

7 Conclusion

In this paper we have presented a max-margin estimation for discriminative synchronous grammar induction. By associating the margin with the translation quality, we directly learn translation rules that optimize the translation performance measured by BLEU. Max-margin estimation also provides us a convenient way to incorporate non-local features. Experiment results validate the effectiveness of optimizing parameters by BLEU, and the importance of incorporating non-local features defined on the target language. These results confirm the advantage of our max-margin estimation framework as it can both optimize BLEU and incorporate non-local features.

Feature engineering is very important for discriminative models. Researchers have proposed various types of features for machine translation, which are often estimated from word alignments. We would like to investigate whether further improvement can be achieved by incorporating such features, especially the context model (Shen et al., 2009) in the future. Because our proposed model is quite general, we are also interested in applying this method to induce linguistically motivated synchronous grammars for syntax-based SMT.

Acknowledgments

The first author was partially supported by 863 State Key Project (No. 2011AA01A207) and National Key Technology R&D Program (No. 2012BAH39B03). We are grateful to the anonymous reviewers for their insightful comments. We also thank Yi Lin for her invaluable feedback.

References

Phil Blunsom and Trevor Cohn. 2006. Discriminative word alignment with conditional random fields. In *Prob. ACL 2006*, July.

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

Phil Blunsom, Trevor Cohn, Chris Dyer, and Miles Osborne. 2009. A gibbs sampler for phrasal synchronous grammar induction. In *Proc. ACL 2009*.

David Burkett, John Blitzer, and Dan Klein. 2010. Joint parsing and alignment with weakly synchronized grammars. In *Proc. NAACL 2010*.

Colin Cherry and Dekang Lin. 2007. Inversion transduction grammar for joint phrasal translation modeling. In *Proc. SSST 2007, NAACL-HLT Workshop on Syntax and Structure in Statistical Translation*, April.

David Chiang, Kevin Knight, and Wei Wang. 2009. 11,001 new features for statistical machine translation. In *Proc. NAACL 2009*.

David Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

Trevor Cohn and Phil Blunsom. 2009. A Bayesian model of syntax-directed tree to string grammar induction. In *Proc. EMNLP 2009*.

John DeNero and Dan Klein. 2010. Discriminative modeling of extraction sets for machine translation. In *Proc. ACL 2010*.

John DeNero, Alexandre Bouchard-Côté, and Dan Klein. 2008. Sampling alignment structure under a Bayesian translation model. In *Proc. EMNLP 2008*.

Chris Dyer, Kevin Gimpel, Jonathan H. Clark, and Noah A. Smith. 2011. The cmu-ark german-english translation system. In *Proc. WMT 2011*.

Kevin Gimpel and Noah A. Smith. 2012. Structured ramp loss minimization for machine translation. In *Proc. NAACL 2012*.

Xiaodong He and Li Deng. 2012. Maximum expected bleu training of phrase and lexicon translation models. In *Proc. ACL 2012*.

Fei Huang and Bing Xiang. 2010. Feature-rich discriminative phrase rescoring for smt. In *Proc. Coling 2010*.

Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proc. ACL 2008*.

Matti Kääriäinen. 2009. Sinuhe – statistical machine translation using a globally trained conditional exponential family translation model. In *Proc. EMNLP 2009*.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. HLT-NAACL 2003*.

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

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

Abby Levenberg, Chris Dyer, and Phil Blunsom. 2012. A bayesian model for learning scfgs with discontiguous rules. In *Proc. EMNLP 2012*. Association for Computational Linguistics, July.

Zhifei Li, Jason Eisner, and Sanjeev Khudanpur. 2009. Variational decoding for statistical machine translation. In *Proc. ACL 2009*.

Chin-Yew Lin and Franz Josef Och. 2004. Orange: a method for evaluating automatic evaluation metrics for machine translation. In *Pro. Coling 2004*.

Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proc. EMNLP 2002*.

Jonathan May and Kevin Knight. 2007. Syntactic realignment models for machine translation. In *Proc. EMNLP 2007*.

Graham Neubig, Taro Watanabe, Eiichiro Sumita, Shinsuke Mori, and Tatsuya Kawahara. 2011. An unsupervised model for joint phrase alignment and extraction. In *Proc. ACL 2011*.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. ACL 2003*.

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

Adam Pauls, Dan Klein, David Chiang, and Kevin Knight. 2010. Unsupervised syntactic alignment with inversion transduction grammars. In *Proc. NAACL 2010*.

Jason Riesa, Ann Irvine, and Daniel Marcu. 2011. Feature-rich language-independent syntax-based alignment for statistical machine translation. In *Proc. EMNLP 2011*.

Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. 2007. Pegasos: Primal estimated sub-gradient solver for svm. In *Proc. ICML 2007*.

Libin Shen, Jinxi Xu, Bing Zhang, Spyros Matsoukas, and Ralph Weischedel. 2009. Effective use of linguistic and contextual information for statistical machine translation. In *Proc. EMNLP 2009*.

Andreas Stolcke. 2002. Srilm – an extensible language modeling toolkit.

Ben Taskar, Dan Klein, Mike Collins, Daphne Koller, and Christopher Manning. 2004. Max-margin parsing. In *Proc. EMNLP 2004*.

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

Xinyan Xiao, Yang Liu, Qun Liu, and Shouxun Lin. 2011. Fast generation of translation forest for large-scale smt discriminative training. In *Proc. EMNLP 2011*.

Xinyan Xiao, Deyi Xiong, Yang Liu, Qun Liu, and Shouxun Lin. 2012. Unsupervised discriminative induction of synchronous grammar for machine translation. In *Proc. Coling 2012*.

Deyi Xiong, Min Zhang, and Haizhou Li. 2010. Learning translation boundaries for phrase-based decoding. In *Proc. NAACL2010*.

Richard Zens and Hermann Ney. 2004. Improvements in phrase-based statistical machine translation. In *Prob. NAACL 2004*.

Hao Zhang, Chris Quirk, Robert C. Moore, and Daniel Gildea. 2008. Bayesian learning of non-compositional phrases with synchronous parsing. In *Proc. ACL 2008*.