Structural Support Vector Machines for Log-Linear Approach in Statistical Machine Translation

Katsuhiko Hayashi[†], *Taro Watanabe*[††*], *Hajime Tsukada*[††], *Hideki Isozaki*[††]

Department of Information Science and Technology
[†]University of Doshisha, Japan
dti0708@mail4.doshisha.ac.jp
[††]NTT Comunication Science Labolatories, Japan
{taro,tsukada,isozaki}@cslab.kecl.ntt.co.jp

Abstract

Minimum error rate training (MERT) is a widely used learning method for statistical machine translation. In this paper, we present a SVM-based training method to enhance generalization ability. We extend MERT optimization by maximizing the margin between the reference and incorrect translations under the L2-norm prior to avoid overfitting problem. Translation accuracy obtained by our proposed methods is more stable in various conditions than that obtained by MERT. Our experimental results on the French-English WMT08 shared task show that degrade of our proposed methods is smaller than that of MERT in case of small training data or out-of-domain test data.

1. Introduction

The state of the art statistical machine translation systems have been modeled by the log-linear approach which is a generalization of the noizy-channel approach. This approach has achieved a lot of great advances because it has provided a natural extention to integrate many useful components [1]. To estimate the weights toward these components according to their performance, minimum error rate training (MERT) [2] was introduced by Och (2003). MERT improves statistical machine translation quality by optimizing the parameter of the log-linear function by using such automatic translation evaluation metrics as the BLEU scores [3].

To train a small number of real-valued features used on a standard phrase-based statistical machine translation system like Moses [16], MERT with BLEU-based objective function is very effective due to line-search algorithm proposed by Och (2003). However, MERT tends to overfit to training data because its objective function consists of no regularizer. To enhance generalization ability, we would like to use other state-of-the-art machine learning techniques for machine translation.

Support vector machines (SVM) have proven to be powerful tools for many tasks in natural language processing [6][7]. The core of the form consists of a smooth convex regularizer such as $\frac{1}{2}||\mathbf{w}||^2$ and the empirical risk term of hinge loss. In this paper we present an approach to optimize the parameter of the log-liner model using the primal form of structural SVM [12]. We expect the convex regularizer or the factor of enlarging the margin (between the reference and the incorrect translation) of SVM to reduce the overfitting problem and enhance generalization ability. Using the BLEU scores to define the hinge loss, our proposed method also inherits the advantages of MERT, which enhance the BLEU scores of translations.

In this paper we carried out on a French-English task and exprimentally compared our proposed method with MERT. We achieved more significant improvements than these obtained by conventional MERT, especially in case of small training data or out-of-domain test data.

The remainder of this paper is organized as follows. Section 2 briefly reviews the framework for the MERT and Section 3 performs the formulations of the structural SVM and the cutting-plane algorithms to introduce our proposed method in Section 4. Section 4 describes the integration of the line-search algorithm with S-slack and 1-slack structural SVM. We experimentally compare structural SVM to MERT on the WMT08 French-English task in Section 5. In section 6, we conclude with summary and future work.

2. Minimum Error Rate Training

2.1. Log-Linear Approach

To translate source sentence $\mathbf{f}$ into another target language $\mathbf{e}$, the log-linear approach seeks a maximum solution:

$$\hat{\mathbf{e}} = \underset{\mathbf{e}}{\operatorname{argmax}} \langle \mathbf{w}, \mathbf{h}(\mathbf{e}, \mathbf{f}) \rangle, \qquad (1)$$

where $\mathbf{h}(\mathbf{e}, \mathbf{f})$ is a feature vector and $\mathbf{w}$ is a weight vector that scales the contributions from all features. This approach has the advantage that addtional models or feature functions can be easily integrated into the overall system. However, it must appropriately optimize a weight vector to obtain the

[†] This research was conducted as an internship program 2008 of NTT.

[*] Taro Watanabe now belongs to National Institute of Information and Communications Technology (NICT) in Japan.

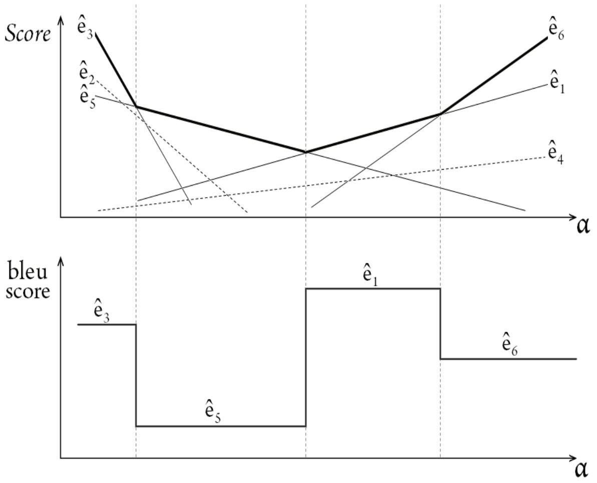

Figure 1: each line of the 6-best translations and BLEU scores with 1-best translation selected by the current parameter α

good translation results.

2.2. Minimum Error Rate Training

Minimum error rate training (MERT) obtains $\mathbf{w}$ that maximizes BLEU scores on S-size training data $\{(\mathbf{r}_s, \mathbf{f}_s)\}_1^S$ and a set of K different candidate translations $\mathbf{C}_s = \{\hat{\mathbf{e}}_{s,1}, \cdots, \hat{\mathbf{e}}_{s,K}\}$ for each input sentence $\mathbf{f}_s$:

$$\underset{\mathbf{w}}{\operatorname{argmax}} \, \mathbf{BLEU}(\left\{ \mathbf{r}_s, \underset{\hat{\mathbf{e}}_s \in \mathbf{C}_s}{\operatorname{argmax}} \langle \mathbf{w}, \mathbf{h}(\hat{\mathbf{e}}_s, \mathbf{f}_s) \rangle \right\}_1^S). \tag{2}$$

If we use the argmin function, we need to calculate the objective function as $1.0 - \mathbf{BLEU}$.

Och's line-search, one of the efficient algorithms for optimizing an objective function, maximizes the function through a sequence of line maximizations along vector directions $\mathbf{d}$. To compute the most probable sentence $\hat{\mathbf{e}}_{s,best}$ from hypotheses $\mathbf{C}_s$, the optimization problem is defined with α as follows:

$$\begin{aligned} \hat{\mathbf{e}}_{s,best} &= \underset{\hat{\mathbf{e}}_s \in \mathbf{C}_s}{\operatorname{argmax}} \langle \mathbf{w} + \alpha \mathbf{d}, \mathbf{h}(\hat{\mathbf{e}}_s, \hat{\mathbf{f}}_s) \rangle \\ &= \underset{\hat{\mathbf{e}}_s \in \mathbf{C}_s}{\operatorname{argmax}} \left\{ \underbrace{\langle \mathbf{w}, \mathbf{h}(\hat{\mathbf{e}}_s, \hat{\mathbf{f}}_s) \rangle}_{intercept} + \alpha \underbrace{\langle \mathbf{d}, \mathbf{h}(\hat{\mathbf{e}}_s, \hat{\mathbf{f}}_s) \rangle}_{slope} \right\}. \end{aligned} \tag{3}$$

Each translation in the hypotheses corresponds to a line with the slope and the intercept in the argmax of Eq.3. Figure.1 shows the example of lines for the 6-best translations. For any particular choice of α, the decoder seeks the translation that yields the largest score. Och's algorithm shifts α from $-\infty$ to ∞ to obtain the best parameter $\mathbf{w}$ while recording the points where two or more lines intersect and selects α which is able to determine the translation obtaining the highest evaluation scores.

2.3. Evaluation Metrics BLEU

The BLEU score [3] used in MERT is defined as follows:

$$\mathbf{BLEU}(\{\hat{\mathbf{e}}\}_1^S; \{\mathbf{r}\}_1^S) = BP \cdot \exp \left\{ \frac{1}{N} \sum_{n=1}^{N} \log p_n(\{\hat{\mathbf{e}}\}_1^S, \{\mathbf{r}\}_1^S) \right\}$$

where $p_n(\cdot)$ is the n-gram precision of hypothesezed translations $\{\mathbf{e}\}_1^S$ given the reference $\{\mathbf{r}\}_1^S$ and BP is a brevity penalty. The BLEU scores are calculated in terms of a whole corpus.

3. Structural Support Vector Machines

In recent years, Support Vector Machines (SVM) for conventional binary classification have been generalized for multi-classification and structured output problems [12][13]. The generalized SVM is called a structural SVM [12]. In the generalized margin-maximization principle we maximize the separation margin, which is the score difference between the reference $\mathbf{r}_s$ and incorrect translation $\hat{\mathbf{e}}_s$ scores. To allow errors in training data, we introduce slack variables ξ_s and optimize with soft-margin criteria:

$$\begin{gathered} \underset{\mathbf{w}, \xi \geq 0}{\operatorname{argmin}} \frac{\lambda}{2} \|\mathbf{w}\|^2 + \frac{1}{S} \sum_{s=1}^{S} \xi_s \\ s.t. \; \forall \hat{\mathbf{e}}_1 \in \mathbf{C}_1 \setminus \mathbf{r}_1 : \;\; \langle \mathbf{w}, \delta \mathbf{h}_1 \rangle \geq 1 - \xi_1 \\ \vdots \\ s.t. \; \forall \hat{\mathbf{e}}_S \in \mathbf{C}_S \setminus \mathbf{r}_S : \;\; \langle \mathbf{w}, \delta \mathbf{h}_S \rangle \geq 1 - \xi_S, \end{gathered} \tag{4}$$

where $\delta \mathbf{h}_s$ is $\mathbf{h}_s(\mathbf{r}_s, \mathbf{f}_s) - \mathbf{h}_s(\hat{\mathbf{e}}_s, \mathbf{f}_s)$. The constraints state that for each training example $(\mathbf{r}_s, \mathbf{f}_s)$, the score $\mathbf{w} \cdot \mathbf{h}(\mathbf{r}_s, \mathbf{f}_s)$ of the correct structure $\mathbf{r}_s$ must be greater than the score $\mathbf{w} \cdot \mathbf{h}(\hat{\mathbf{e}}_s, \mathbf{f}_s)$ of all incorrect structures $\hat{\mathbf{e}}$ by the margin 1.

The standard structural SVM optimization problem has also been generalized in several ways [12][13]. Tsochantaridis et al. (2005) introduced two different ways of using a hinge loss to the convex upper bound of the loss, namely, "margin-rescaling" and "slack-rescaling". In this paper, we use only a margin-rescaling. The margin-rescaling is for the special case of the Hamming loss. Each margin-rescaling constraint has the following form:

$$s.t. \; \forall \hat{\mathbf{e}}_s \in \mathbf{C}_s \setminus \mathbf{r}_s : \;\; \langle \mathbf{w}, \delta \mathbf{h}_s \rangle \geq \triangle(\mathbf{r}_s, \hat{\mathbf{e}}_s) - \xi_s$$

where this rescaling can penalize $\hat{\mathbf{e}} \neq \mathbf{r}$ with high loss $\Delta(\mathbf{r}, \hat{\mathbf{e}})$ more severely than that with small loss.

3.1. Cutting-Plane Algorithm

The optimization problem in Eq.4 with margin-rescaling has $O(S|\mathbf{C}|)$ constraints. In general, $|\mathbf{C}|$ is extremely large or infinite. To cut a large number of these constraints, the cutting-plane algorithm (Algorithm 1) iteratively finds the most violated constraint through the training data:

Algorithm 1 Cutting-plane training for S-slack Formulation with margin-rescaling

```
1: $\textbf{Input} : \mathbf{w}, \mathbf{C}_1^S, \{\mathbf{r}_s, \mathbf{f}_s\}_1^S, \epsilon$
2: $\mathscr{W}_1^S = \{\}$
3: repeat
4:   for $s = 1 \dots S$ do
5:     $\hat{\mathbf{e}}_s = \mathrm{argmax}_{\hat{\mathbf{e}}_s \in \mathbf{C}_s}\{\Delta(\mathbf{r}_s, \hat{\mathbf{e}}_s) - \langle \mathbf{w}, \delta\mathbf{h}_s\rangle\}$
6:     if $\Delta(\mathbf{r}_s, \hat{\mathbf{e}}_s) - \langle \mathbf{w}, \delta\mathbf{h}_s\rangle > \xi_s + \epsilon$ then
7:       $add(\mathscr{W}_s, \hat{\mathbf{e}}_s)$
8:       $\mathbf{w} = \mathrm{argmin}_{\mathbf{w},\xi\geq 0} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{S}\sum_{s=1}^S \xi_s$
             $s.t. \forall \hat{\mathbf{e}}_1 \in \mathscr{W}_1 : \langle \mathbf{w}, \delta\mathbf{h}_1\rangle \geq \triangle(\mathbf{r}_1, \hat{\mathbf{e}}_1) - \xi_1$
                 $\vdots$
                 $\forall \hat{\mathbf{e}}_S \in \mathscr{W}_S : \langle \mathbf{w}, \delta\mathbf{h}_S\rangle \geq \triangle(\mathbf{r}_S, \hat{\mathbf{e}}_S) - \xi_S$
9:     end if
10:  end for
11: until
```

This algorithm iteratively constructs a working set $\mathscr{W} = \mathscr{W}_1 \cup \cdots \cup \mathscr{W}_S$ of constraints. The most violated constraint for margin-rescaling (line 5) is defined as follows:

$$\xi_s^* = \max_{\hat{\mathbf{e}}_s \in \mathbf{C}_s \backslash \mathbf{r}_s} \{\triangle(\mathbf{r}_s, \hat{\mathbf{e}}_s) - \langle \mathbf{w}, \delta\mathbf{h}_s\rangle\}. \tag{5}$$

If this constraint is violated by more than the desired precision ϵ (line 6), the constraint is added to the working set $\mathscr{W}$ and the QP is solved over the $\mathscr{W}$ (line 8).

3.2. 1-Slack Formulation

Joachims et al. (2009) proposed to replace the S cutting-plane models of the hinge loss with a single cutting-plane model for the sum of the hinge losses. This model has only one slack variable ξ that is shared across all constraints:

$$\underset{\mathbf{w},\xi\geq 0}{\mathrm{argmin}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \xi. \tag{6}$$

The constraint for 1-slack formulation with margin-rescaling is

$$s.t. \; \forall(\hat{\mathbf{e}}_1, \cdots, \hat{\mathbf{e}}_S) \in \mathbf{C}^S :$$
$$\frac{1}{S}\sum_{s=1}^{S}\langle \mathbf{w}, \delta\mathbf{h}_s\rangle \geq \frac{1}{S}\sum_{s=1}^{S}\Delta(\mathbf{r}_s, \hat{\mathbf{e}}_s) - \xi.$$

It is proven that the S-slack and 1-slack formulations are equivalent in some points [13]. In the next section we use this formulation to calculate the whole corpus-wise BLEU scores.

Cutting-plane training for 1-slack formulation is showed in Algoritm 2:

Algorithm 2 Cutting-plane training for 1-slack Formulation with margin-rescaling

```
$\textbf{Input} : \mathbf{w}, \mathbf{C}_1^S, \{\mathbf{r}_s, \mathbf{f}_s\}_1^S$
$\mathscr{W} = \{\}$
repeat
  $\mathbf{w} = \mathrm{argmin}_{\mathbf{w},\xi\geq 0} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{S}\sum_{s=1}^S \xi_s$
    $s.t. \; \forall(\hat{\mathbf{e}}_1, \cdots, \mathbf{e}_s) \in \mathscr{W} : \frac{\lambda}{S}\sum_{s=1}^S\langle \mathbf{w}, \delta\mathbf{h}_s\rangle \geq \frac{1}{S}\sum_{s=1}^S \triangle(\mathbf{r}_s, \hat{\mathbf{e}}_s) - \xi_s$
  for $s = 1 \dots S$ do
    $\hat{\mathbf{e}}_s = \mathrm{argmax}_{\hat{\mathbf{e}}_s \in \mathbf{C}_s}\{\Delta(\mathbf{r}_s, \hat{\mathbf{e}}_s) - \langle \mathbf{w}, \delta\mathbf{h}_s\rangle\}$
  end for
  $add(\mathscr{W}, (\hat{\mathbf{e}}_1, \cdots, \hat{\mathbf{e}}_S))$
until
```

It iteratively constructs a working set $\mathscr{W}$ of constrains. In each iteration, this algorithm optimizes the parameter over the current working set $\mathscr{W}$, find the most violated constrait, and add it to the working set. Unlike the S-slack algorithm, only a single costraint is added in each iteration.

4. Minimum Error Rate Training based on Structural SVM

In this section we describe how to apply the structural SVM to the MERT objective function. The first subsection addresses how to calculate the feature score of the "correct" sentence. In the second subsection we define loss function $\Delta(\mathbf{r}_s, \hat{\mathbf{e}}_s)$ using the BLEU scores, and the third subsection extends Och's line-search algorithm as the optimization algorithm for the SVM-based objective function.

4.1. Selecting the Configuration in the K-best list

For structural SVM, we need to label a correct candidate translation for each source sentence from its K-best list of candidates, which is called a configuration. Since the BLEU scores are not cumulative, we cannot efficiently select the best configuration from the K-best list. So we approximate it by a greedy search algorithm [14].

This algorithm considers the impact on the training set score when selecting an alternative translation by subtracting the statistics for the current configuration choice from the accumulated statistics and adding those for the alternative and selects the translation which results in the highest score. Repeat this process and continue untill there are no configuration changes. The configuration obtained by this algorithm specifies the correct candidate for each K-best list, and the BLEU scores are the upper bound for the BLEU scores on the training set.

4.2. Loss Function for Rescaling

4.2.1. *Approximation for the Sentence-wise BLEU*

The BLEU scores are defined in terms of a whole corpus and not over individual sentences. However we need to calculate the sentence-wise BLEU scores to define the loss function $\Delta(\mathbf{r}_s, \hat{\mathbf{e}}_s)$. To calculate sentence-wise BLEU we use the approximated BLEU like the one proposed by Watanabe et al. (2006). Given the configurations for S input sentences $\{\hat{\mathbf{e}}^*_1 \dots \hat{\mathbf{e}}^*_S\}$, this approximated BLEU scores on translation candidate $\hat{\mathbf{e}}_s$ for s-th input sentence are calculated by substituting $\hat{\mathbf{e}}^*_s$ with $\hat{\mathbf{e}}_s$.

4.2.2. Loss Functions based on Sentence-wise and Corpus-wise BLEU

The sentence-wise loss function $\Delta(\mathbf{r}_s, \hat{\mathbf{e}}_s)$ is defined as follows by using this approximated BLEU scores:

$$Q \times \{\mathbf{BLEU}(\{\mathbf{r}_s, \hat{\mathbf{e}}^*_s\}_1^S) - \mathbf{apBLEU}(\{\mathbf{r}_s, \hat{\mathbf{e}}_s\}_1^S)\},$$

where Q is a constant for scaling the BLEU loss function. If we set the parameter Q on high, we regard the loss function as important. We use the sentence-wise BLEU loss function $\Delta(\mathbf{r}_s, \hat{\mathbf{e}}_s)$ to calculate the objective function of the S-slack SVM formulation.

Since we believe that the average sentence-wise BLEU scores are less reliable than the whole corpus-wise BLEU scores, we tried to apply the corpus-wise BLEU to the objective function using a 1-slack formulation SVM assuming $\frac{1}{S}\sum_{s=1}^{S}\Delta(\mathbf{r}_s, \mathbf{e}_s) \propto$ corpus-wise BLEU loss function $\Delta(\{\mathbf{r}_s, \mathbf{e}_s\}_1^S)$. The corpus-wise BLEU loss function $\Delta(\{\hat{\mathbf{e}}^*_s, \hat{\mathbf{e}}_s\}_1^S)$ is as follows:

$$Q \times \{\mathbf{BLEU}(\{\mathbf{r}_s, \hat{\mathbf{e}}^*_s\}_1^S) - \mathbf{BLEU}(\{\mathbf{r}_s, \hat{\mathbf{e}}_s\}_1^S)\}.$$

1-slack formulation with margin-rescaling is constructed using the corpus-wise BLEU loss function $\Delta(\{\hat{\mathbf{e}}^*_s, \hat{\mathbf{e}}_s\}_1^S)$:

$$\begin{aligned} &\operatorname*{argmin}_{\mathbf{w},\xi\geq 0} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \xi. \qquad (7)\\ &\quad s.t.\ \forall(\hat{\mathbf{e}}_1, \cdots, \hat{\mathbf{e}}_S) \in \mathbf{C}^S :\\ &\quad \frac{1}{S}\sum_{s=1}^{S}\langle \mathbf{w}, \delta\mathbf{h}_s\rangle \geq \Delta(\{\hat{\mathbf{e}}^*_s, \hat{\mathbf{e}}_s\}_1^S) - \xi. \end{aligned}$$

Unlike the margin infused relaxed algorithm (MIRA) [9] and S-slack formulation, we can directly apply the corpus-wise BLEU to the SVM objective function without approximating the BLEU scores.

4.3. Optimization Algorithm

4.3.1. Line-search Algorithm

Next we describe extended Och's line-search algorithms for S-slack and 1-slack formulation of the Structural SVM. These pseudocodes for the line-search to optimize parameter $\mathbf{w}$ are given by Algorithm 3,4. In Algorithm 3,4 we can find the range of values along the direction vector $\mathbf{d}$ to which each candidate translation is assigned the best score. Algorithm 3 is the line-search algorithm for S-slack formulation of the structural SVM:

Algorithm 3 extended Och's line-search algorithm for S-slack formulation

```
 1: Input : $\mathbf{w}, \mathbf{d}, \mathbf{C}_1^S, \{\mathbf{r}_s, \hat{\mathbf{e}}^*_s, \mathbf{f}_s\}_1^S$
 2: $\mathscr{I} = \{\}$
 3: for $s = 1 \dots S$ do
 4:     for all $\hat{\mathbf{e}}_s \in \mathbf{C}_s \setminus \hat{\mathbf{e}}^*_s$ do
 5:         $\hat{\mathbf{e}}_s.m = \langle \mathbf{d}, \delta\mathbf{h}_s\rangle$
 6:         $\hat{\mathbf{e}}_s.b = \Delta(\hat{\mathbf{e}}^*_s, \hat{\mathbf{e}}_s) - \langle \mathbf{w}, \delta\mathbf{h}_s\rangle$
 7:     end for
 8:     $i = 0$
 9:     $l_i = \operatorname{argmin}_{\hat{\mathbf{e}}_s\in\mathbf{C}_s\setminus\hat{\mathbf{e}}^*_s} \hat{\mathbf{e}}_s.m$
10:     $x_i = -\infty$
11:     repeat
12:         $i = i + 1$
13:         if $\{\frac{l_{i-1}.b - l_i.b}{l_i.m - l_{i-1}.m}\} > x_{i-1}$ then
14:             $l_i = \operatorname{argmin}_{\hat{\mathbf{e}}_s\in\mathbf{C}_s\setminus\hat{\mathbf{e}}^*_s}\{\frac{l_{i-1}.b - \hat{\mathbf{e}}_s.b}{\hat{\mathbf{e}}_s.m - l_{i-1}.m}\}$
15:             $x_i = \{\frac{l_{i-1}.b - l_i.b}{l_i.m - l_{i-1}.m}\}$
16:         end if
17:     until no more intersections
18:     $add(\mathscr{I}, x_1^i)$
19:     $add(\mathscr{I}, max(\mathscr{I}) + 2\epsilon)$
20:     $x_{best} = \operatorname{argmin}_{x\in\mathscr{I}} Obj(\mathbf{w} + (x - \epsilon)\mathbf{d}, \mathbf{C}_1^s, \{\mathbf{r}_j, \hat{\mathbf{e}}^*_j, \mathbf{f}_j\}_{j=1}^s)$
21:     $\mathbf{w}+= (x_{best} - \epsilon)$
22:     $delete(\mathscr{I}, max(\mathscr{I}) + 2\epsilon)$
23: end for
24: return $\mathbf{w}$
```

In case of S-slack formulation (Algorithm 3), the max function in Eq. 5 corresponds to the argmax function in Eq. 3, so we can find a translation that has the most violated constraint ξ_s by a line-search algorithm with the following slope and intercept (line 5,6) :

$$\operatorname*{argmax}_{\hat{\mathbf{e}}_s\in\mathbf{C}_s}\left\{\underbrace{\Delta(\mathbf{e}^*_s, \mathbf{e}_s) - \langle \mathbf{w}, \delta\mathbf{h}_s\rangle}_{intercept} + \alpha\underbrace{\langle \mathbf{d}, \delta\mathbf{h}_s\rangle}_{slope}\right\}.$$

Algorithm 3 iteratively constructs the line intersections $\mathscr{I}$ through the training examples and estimates the parameter variation $(x - \epsilon)$ which is able to select the translation minimizing the objective function. The process of Algorithm 3 which extracts the most violated translation is similar to the cutting-plane training in Algorithm 1. In case of the S-slack formulation with margin-rescaling, the Obj function (line 20) is constructed by each of the most violated translations, as in Eq. 4, using margin-rescaling constraints.

Algorithm 4 is the line-search algorithm for 1-slack formulation of the structural SVM:

Algorithm 4 extended Och's line-search algorithm for 1-slack formulation

```
1:  Input : $\mathbf{w}, \mathbf{d}, \mathbf{C}_1^S, \{\mathbf{r}_s, \hat{\mathbf{e}}_s^*, \mathbf{f}_s\}_1^S$
2:  $\mathscr{I} = \{\}$
3:  for $s = 1 \ldots S$ do
4:      for all $\hat{\mathbf{e}}_s \in \mathbf{C}_s \setminus \hat{\mathbf{e}}_s^*$ do
5:          $\hat{\mathbf{e}}_s.m = \langle \mathbf{d}, \mathbf{h}_s(\hat{\mathbf{e}}_s, \mathbf{f}_s) \rangle$
6:          $\hat{\mathbf{e}}_s.b = \langle \mathbf{w}, \mathbf{h}_s(\hat{\mathbf{e}}_s, \mathbf{f}_s) \rangle$
7:      end for
8:      $i = 0$
9:      $l_i = \text{argmin}_{\hat{\mathbf{e}}_s \in \mathbf{C}_s \setminus \hat{\mathbf{e}}_s^*} \hat{\mathbf{e}}_s.m$
10:     $x_i = -\infty$
11:     repeat
12:         $i = i + 1$
13:         if $\{\frac{l_{i-1}.b - l_i.b}{l_i.m - l_{i-1}.m}\} > x_{i-1}$ then
14:             $l_i = \text{argmin}_{\hat{\mathbf{e}}_s \in \mathbf{C}_s \setminus \hat{\mathbf{e}}_s^*} \{\frac{l_{i-1}.b - \hat{\mathbf{e}}_s.b}{\hat{\mathbf{e}}_s.m - l_{i-1}.m}\}$
15:             $x_i = \{\frac{l_{i-1}.b - l_i.b}{l_i.m - l_{i-1}.m}\}$
16:         end if
17:     until no more intersections
18:     $add(\mathscr{I}, x_1^i)$
19: end for
20: $add(\mathscr{I}, max(\mathscr{I}) + 2\epsilon)$
21: $x_{best} = \text{argmin}_{x \in \mathscr{I}} Obj(\mathbf{w} + (x - \epsilon)\mathbf{d}, \mathbf{C}_1^S, \{\mathbf{r}_s, \hat{\mathbf{e}}_s^*, \mathbf{f}_s\}_1^S)$
22: return $\mathbf{w} + (x_{best} - \epsilon)\mathbf{d}$
```

For the 1-slack formulation we should calculate slopes m and intercepts b the same as the S-slack formulation, but, to avoid bias toward the line-search procedure by sentence-wise BLEU, we computed them the same as MERT (line 5-6). Unlike the S-slack formulation, this algorithm constructs the line intersections $\mathscr{I}$ over all the training samples and then estimates the best parameter varidation. The process of this algorithm which extracts 1-best translation is similar to that of Algorithm 2 extracting the most violated constraint. In case of the 1-slack formulation with margin-rescaling, the Obj function (line 21) is constructed as Eq. 7.

5. Experimental Results

5.1. Systems

Experiments were conducted using a standard phrase-based statistical MT system called Moses [16] to generate K-best lists (K=1000). Moses employs standard real-valued features:

- N-gram language model: $Pr(\mathbf{e})$ to calculate the fluency of the target side.
- Lexical translation model: $t(e_i|f_j)$, $t(f_j|e_i)$ to calculate the word translation probability.
- Phrase translation model: $\phi(\overline{\mathbf{e}}|\overline{\mathbf{f}})$, $\phi(\overline{\mathbf{f}}|\overline{\mathbf{e}})$ to calculate the phrase translation probability.
- Three orientation types reordering model[17]: $p(m|\overline{\mathbf{f}}, \overline{\mathbf{e}})$, $p(s|\overline{\mathbf{f}}, \overline{\mathbf{e}})$, $p(d|\overline{\mathbf{f}}, \overline{\mathbf{e}})$ to capture the lexicalized information.
- Word , Phrase penalty: To control the target length and the average length of the phrases.

In this paper, we trained these small number of features and phrases were extracted using a typical approach [16] that ran GIZA++ [18]. We used a Katz smoothing 5-gram language model that was created using the SRILM toolkit [19].

5.2. Data Set

For experiments we used the French-English data provided for the Europarl-based WMT08 shared task. Europarl corpus was collected from the proceedings of European Parliament [20]. This training corpus contains about 1.3 M sentences. Parameters were tuned over the provided development set (dev2006) that consisted of 2000 sentences with one reference. We used two open test sets: Europarl test 2008, consisting of 2000 sentences with one reference, and News newstest 2008 (out-of-domain), consisting of 1500 sentences. Table 1 shows these contents in more detail.

Table 1: The Data statistics

Data		Sent.	Word.	Avg. leng
Training	fr	1.28M	39.956M	31.2
	en	1.28M	35.948M	28.1
Development	fr	2.0K	64.331K	32.2
	en	2.0K	58.761K	29.4
Test08	fr	2.0K	65.644K	32.8
	en	2.0K	60.188K	30.1
NewsTest08	fr	1.5K	41.037K	27.4
	en	1.5K	36.438K	24.3

5.3. Results

5.3.1. Tuning Hyperparamers

Figure 2 shows the effect of the hyperprameter λ which emphasizes the convex regularizer on SVM objective function. When we set λ high, the curve is like a quadratic function and the best parameter to optimize the objective function is close to 0. On the other hand, if we set it low, the shape of the function is almost the same MERT's objective function. Hence it is very important for obtaining the good translation results to appropriately set the hyperparameters.

We tuned hyperparameters λ and Q in the SVM-based method by the cross-validation method. We divided dev2006 into two and the first estimation was performed on one (another was used for development set) and the second did on another. In our cross-validation experiments for tuning the hyperparameters, we noticed that the higher Q is, the better

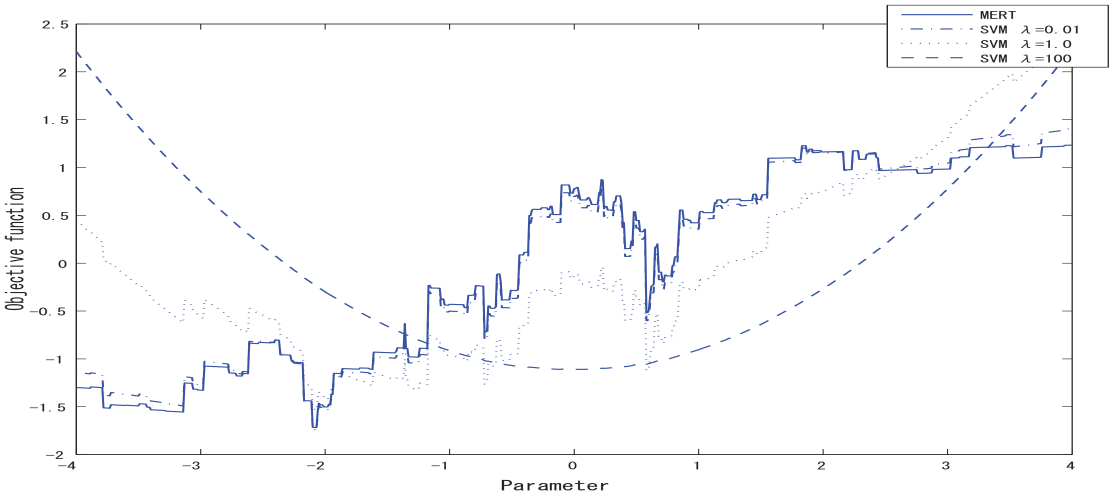

Figure 2: This shows the shapes of BLEU and 1-slack SVM objective function for one parameter. These lines were calculated by 800 development sentences randomly selected from dev06 for development data when the hyperparameter Q is fixed 1000.0.

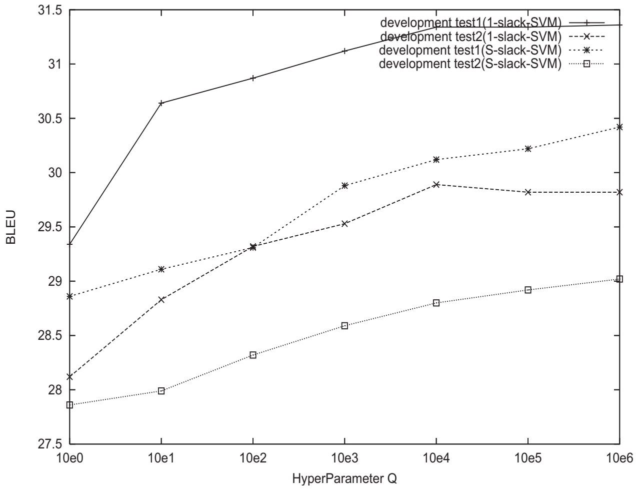

Figure 3: Tuninig test for hyperparameter Q of structural SVM (fixed λ=1.0) by increasing it.

BLEU scores on the development test set are (Figure 3); so in the open test experiments, we always fixed Q=100000.0 , λ=1.0 for S-slack SVM and Q=10000.0 , λ=1.0 for 1-slack SVM.

5.3.2. Test Experiment

We compared our proposed SVM-based method to MERT using the whole dev2006 for development data on 4-gram BLEU scores of two open test sets. Table 2 shows that 1-slack-SVM outperformed MERT. On the other hand, S-slack-SVM was not more effective than MERT because we approximated the BLEU scores to calculate the S-slack-SVM objective function while we evaluated the BLEU scores at the corpus level.

Table 2: BLEU scores on the test08 and news08 test data obtained by models trained by MERT and SVM.

method	test08	news08
untune	30.84	13.75
smoothed-MERT (Och, 03)	31.96	13.76
MERT	32.36	13.81
S-slack-SVM	32.31	**14.02**
1-slack-SVM	**32.42**	**14.13**

Table 2 shows the translation accuracy in both in-domain and out-of-domain test set. The ”untune” means the result from default parameters and we performed MERT and SVM training, starting from these parameters. Our two proposed method based on SVM achieve comparable performance of MERT in in-domain test set (test08) and slightly outperform MERT in out-of-domain test set (news08). This means that SVM has good effects on training the parameter of log-linear model especially in case of the out-of-domain translation.

5.3.3. Experiments in Case of Data Sparseness and Out-of-domain Problems

We also conducted the experiment in case of small development data. Figure 4 shows the performance of Moses with training MERT, S-slack SVM and 1-slack SVM by gradually increasing the development data. The BLEU scores of MERT on the test08 data is drastically degraded when the development data size is getting smaller, and the learning curve

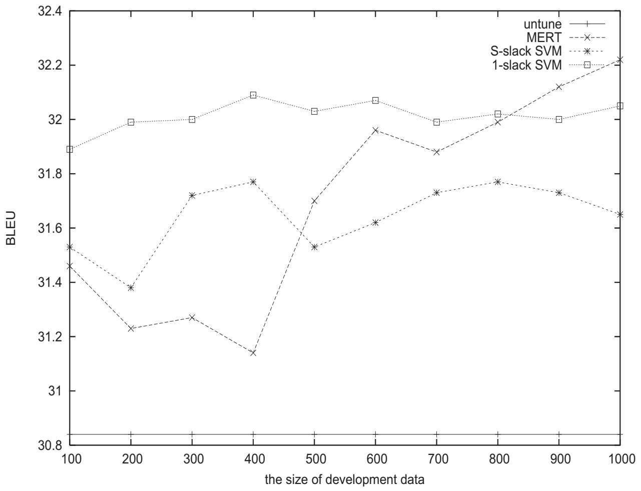

Figure 4: BLEU scores as a function of development data size.

is unstable. On the other hand, S-slack and 1-slack SVM is more stable than MERT and degrade of the smaller development data is fewer than that of MERT. Especially, 1-slack SVM is most stable among these three.

Table 3: The average improvements of BLEU scores on the test08 and news08 (out-of-domain) when we trained the paramenters using only 400 development sentences with MERT and SVM-based algorithms four times.

method	**test08**	**news08**
untune	30.84	13.75
smoothed-MERT (Och, 03)	$+0.22$	-0.08
MERT	$+0.40$	$+0.03$
S-slack-SVM	$+0.45$	$+0.21$
1-slack-SVM	$+0.92$	$+0.40$

Table 3 shows the average improvements of BLEU scores when we train the parameters using only 400 sentences randomly selected from the development data. We repeated the experiment four times and averaged the improvements of BLEU scores. Table 4 show these results in more details. This indicates that two SVM methods reduce the overfitting problem when assuming that only few development data are available or test set is out-of-domain.

Table 4: BLEU scores of two open test sets obtained when training by MERT, S-slack-SVM and 1-slack-SVM using four development sets containing 400 sentences randomly selecting from WMT-08 dev2006.

Method		1	2	3	4
MERT	test08	30.96	31.46	31.21	31.32
	news08	13.66	13.81	13.83	13.81
S-slack-SVM	test08	**31.30**	31.45	31.15	**31.45**
	news08	**13.80**	**13.98**	**14.11**	**13.94**
1-slack-SVM	test08	**31.43**	**32.03**	**31.37**	**32.22**
	news08	**14.01**	**14.24**	**14.18**	**14.24**

6. Related Work

Crammer et al. (2006) proposed Margin infused relaxed algorithm (MIRA) , which was the online large-margin training algorithm for structured classification, and Watanabe et al. (2007) applied it to a discriminative training algorithm for statistical machine translation to estimate a large number of parameters. In some points, our proposed method is diffrent from MIRA. First, the proposed algorithm is a batch style algorithm and using 1-slack formulation of structural SVM proposed by Joachims (2009) we try to use corpus-wise BLEU for the objective function without approximating the BLEU scores. Secondly, we directly apply the line-search algorithm to SVM optimization problem to estimate a small number of paramenters.

Cer and Manning (2008) proposed the other approach to regularize the objective function of the MERT. This regularization was not to search the current best optima but to consider the adjacent evaluation scores with fixed window size during line-search because the objective function had a very deep and narrow optima. This approach was different from our proposed method, but it statistically achieves significant gains when combined with line-search.

7. Conclusion

We presented a new method to regularize the MERT objective function using structural SVM. This function has $\frac{1}{2}||\mathbf{w}||^2$ as a smooth convex regularizer and a factor maximizing the score margin between a reference and an incorrect translation. We also tried to apply the corpus-wise BLEU score to the objective function without approximating the BLEU scores for each sentence. To optimize a small number of real-valued parameters with this function, we directly used Och's line-search algorithm. The experimental results show that a SVM-based methods are more stable than MERT in various conditions. They outperform MERT when only small development data is available or these are mismatch between the training and test conditions.

In the future, we plan to experiment on a decoder that has a large number of features because SVM-based algorithm is expected to work more effectively on the sparse vector space [22]. We also think that a gradient based algorithm such as Pegasos [21] and a software package SVM^{struct} [13] for SVM dual formulation allowing the use of kernels are more appropriate methods optimizing the parameter on such a decoder than Och's line-search algorithm.

8. Acknowledgements

I would like to thank Yoshinobu Tonomura, Jun Suzuki, Seiichi Yamamoto and Shigeru Katagiri because of providing me the chance of this internship program, and Masafumi Nishida and Katsuhito Sudo for very helpful and detailed comments.

9. References

[1] Och, F. J. and Ney, H., “Discriminative training and maximum entropy models for statistical machine translation”, In Proc. ACL, pp. 295-302, 2002.

[2] Och, F. J., “Minimum error rate training in statistical machine translation”, In Proc. ACL, pp. 160-167, 2003.

[3] Papineni, K. A., Roukos, S., Ward, T. and Zhu, W-J., “Bleu: a method for automatic evaluation of machine translation”, In Proc. ACL, pp. 311-318, 2001.

[4] Chiang, D., “A hierachical phrase-based model for statistical machine translation”, In Proc. ACL, pp. 263-270, 2005.

[5] Quirk, C., Menezes, A. and Cherry, C., “Dependency Treelet Translation: Syntactically Informed Phrasal SMT”, In Proc. ACL, pp. 271-279, 2005.

[6] Joachims, T., “Text Categorization with Support Vector Machines: Learning with Many Relevant Features”, In Proc. ECML, 1998.

[7] Kudo, T. and Matsumoto, Y., “Japanese Dependency Structure Analysis Based on Support Vector Machines”, In Proc. EMNLP/VLC, 2000.

[8] Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S. and Singer, Y., “Online passive agressive algorithms”, JMLR, Vol. 7, pp. 551-585, 2006.

[9] Watanabe, T., Suzuki, J., Tsukada, H. and Isozaki, H., “Online Large-Margin Training for Statistical Machine Translation”, In Proc. EMNLP/CoNLL, pp. 764-773, 2007.

[10] McDonald, R., Crammer, K. and Pereira, F., “Online large-margin training of dependency parsers”, In Proc. ACL, pp. 91-98, 2005.

[11] Shimizu, N. and Haas, A., “Exact decoding for jointly labeling and chunking sequneces”, In Proc. COLING/ACL, pp. 763-770, 2006.

[12] Tsochantaridis, I., Joachims, T., Hofmann, T. and Altun, Y. “Large margin methods for structured and interdependent output variables”, JMLR, Vol. 6, pp. 1453-1484, 2005.

[13] Joachims, T., Finley, T., and Chun-Nam Yu. “Cutting-Plane Training of Structural SVMs”, Machine Learning, to appear, 2009.

[14] Venugopal, A. and Vogel, S. “Considerations in maximum mutual information and minimum classiffication error training for statistical machine translation”, In Proc. EAMT, 2005.

[15] Watanabe, T., Suzuki, J., Tsukada, H., and Isozaki, H. “NTT Statistical Machine Translation for IWSLT 2006”, In Proc. IWSLT, pp. 95-102, 2006.

[16] 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. “Moses: Open source toolkit for statistical machine translation”, In Proc. ACL, pp.177-180, 2007.

[17] Koehn, P., Axelrod, A., Mayne, A-B., Callison-Burch, C., Osborne, M., and Talbot, D. “Edinburgh System Description for 2005 IWSLT Speech Translation Evaluation”, In Proc. IWSLT, 2005.

[18] Och, F. J. and Ney, H. “A systematic comparison of various statistical alignment models”, Computational Linguistics, Vol. 29, No. 1, pp. 19-51, 2003.

[19] Stockle, A. “SRILM - an extensible language modeling toolkit”, In Proc. ICSLP, 2002

[20] Koehn, P., “Europarl: A Parallel Corpus for Statistical Machine Translation”, MT Summit, 2005.

[21] Shalev-Shwartz, S., Singer, Y., and Srebro, N., “Pegasos: primal estimated subgradient solver for SVM”, In proc. ICML, pp. 807-814, 2007.

[22] Chiang, D., Knight, K. and Wang, W., “11,001 New Features for Statistical Machine Translation”, In proc. NAACL, 2009.

[23] Cer, D., Jurafsky, D., and Manning, C. D., “Regularization and search for minimum error rate training”, In proc. The Third Workshop on Statistical Machine Translation, pp.26-34, 2008.

[24] Macherey, W., Och, F. J., Thayer, I. and Uskoreit, J., “Lattice-based minimum error rate training for statistical machine translation”, In proc. EMNLP, 2008.

[25] Moore, R. C. and Quirk, C., “Random restarts in minimum error rate training for statistical machine translation”, In proc. COLING, pp. 585-592, 2008.

[26] Brown, P. F., Pietra, S. A. D., Pietra, V. D. J., and Mercer, R. L., “The mathematics of statistical machine translation: Parameter estimation”, Computational Linguistics, Vol. 19, No. 2, pp. 263-312, 1993.