Optimized Online Rank Learning for Machine Translation

Taro Watanabe
National Institute of Information and Communications Technology
3-5 Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-0289 JAPAN
{taro.watanabe}@nict.go.jp

Abstract

We present an online learning algorithm for statistical machine translation (SMT) based on stochastic gradient descent (SGD). Under the online setting of rank learning, a corpus-wise loss has to be approximated by a batch local loss when optimizing for evaluation measures that cannot be linearly decomposed into a sentence-wise loss, such as BLEU. We propose a variant of SGD with a larger batch size in which the parameter update in each iteration is further optimized by a passive-aggressive algorithm. Learning is efficiently parallelized and line search is performed in each round when merging parameters across parallel jobs. Experiments on the NIST Chinese-to-English Open MT task indicate significantly better translation results.

1 Introduction

The advancement of statistical machine translation (SMT) relies on efficient tuning of several or many parameters in a model. One of the standards for such tuning is minimum error rate training (MERT) (Och, 2003), which directly minimize the loss of translation evaluation measures, i.e. BLEU (Papineni et al., 2002). MERT has been successfully used in practical applications, although, it is known to be unstable (Clark et al., 2011). To overcome this instability, it requires multiple runs from random starting points and directions (Moore and Quirk, 2008), or a computationally expensive procedure by linear programming and combinatorial optimization (Galley and Quirk, 2011).

Many alternative methods have been proposed based on the algorithms in machine learning, such as averaged perceptron (Liang et al., 2006), maximum entropy (Och and Ney, 2002; Blunsom et al., 2008), Margin Infused Relaxed Algorithm (MIRA) (Watanabe et al., 2007; Chiang et al., 2008b), or pairwise rank optimization (PRO) (Hopkins and May, 2011). They primarily differ in the mode of training; online or MERT-like batch, and in their objectives; max-margin (Taskar et al., 2004), conditional log-likelihood (or softmax loss) (Berger et al., 1996), risk (Smith and Eisner, 2006; Li and Eisner, 2009), or ranking (Herbrich et al., 1999).

We present an online learning algorithm based on stochastic gradient descent (SGD) with a larger batch size (Shalev-Shwartz et al., 2007). Like Hopkins and May (2011), we optimize ranking in n-best lists, but learn parameters in an online fashion. As proposed by Haddow et al. (2011), BLEU is approximately computed in the local batch, since BLEU is not linearly decomposed into a sentence-wise score (Chiang et al., 2008a), and optimization for sentence-BLEU does not always achieve optimal parameters for corpus-BLEU. Setting the larger batch size implies the more accurate corpus-BLEU, but at the cost of slower convergence of SGD. Therefore, we propose an optimized update method inspired by the passive-aggressive algorithm (Crammer et al., 2006), in which each parameter update is further rescaled considering the tradeoff between the amount of updates to the parameters and the ranking loss. Learning is efficiently parallelized by splitting training data among *shards* and by merging parameters in each round (McDonald et al., 2010). Instead

2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 253–262,
Montréal, Canada, June 3-8, 2012.

of simple averaging, we perform an additional line search step to find the optimal merging across parallel jobs.

Experiments were carried out on the NIST 2008 Chinese-to-English Open MT task. We found significant gains over traditional MERT and other tuning algorithms, such as MIRA and PRO.

2 Statistical Machine Translation

SMT can be formulated as a maximization problem of finding the most likely translation e given an input sentence f using a set of parameters θ (Brown et al., 1993)

$$\hat{e} = \arg\max_{e} p(e|f;\theta). \tag{1}$$

Under this maximization setting, we assume that $p(\cdot)$ is represented by a linear combination of feature functions $\mathbf{h}(f,e)$ which are scaled by a set of parameters $\mathbf{w}$ (Och and Ney, 2002)

$$\hat{e} = \arg\max_{e} \mathbf{w}^\top \mathbf{h}(f,e). \tag{2}$$

Each element of $\mathbf{h}(\cdot)$ is a feature function which captures different aspects of translations, for instance, log of n-gram language model probability, the number of translated words or log of phrasal probability.

In this paper, we concentrate on the problem of learning $\mathbf{w}$, which is referred to as *tuning*. One of the standard methods for parameter tuning is minimum error rate training (Och, 2003) (MERT) which directly minimizes the task loss $\ell(\cdot)$, i.e. negative BLEU (Papineni et al., 2002), given training data $D = \{(f^1, \mathbf{e}^1), ..., (f^N, \mathbf{e}^N)\}$, sets of paired source sentence f^i and its reference translations $\mathbf{e}^i$

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \ell(\left\{\arg\max_{e} \mathbf{w}^\top \mathbf{h}(f^i, e)\right\}_{i=1}^{N}, \{\mathbf{e}^i\}_{i=1}^{N}). \tag{3}$$

The objective in Equation 3 is discontinuous and non-convex, and it requires decoding of all the training data given $\mathbf{w}$. Therefore, MERT relies on a derivative-free unconstrained optimization method, such as Powell's method, which repeatedly chooses one direction to optimize using a line search procedure as in Algorithm 1. Expensive decoding is approximated by an n-best merging technique in which decoding is carried out in each epoch of iterations t and the maximization in Eq. 3 is approximated by search over the n-bests merged across iterations. The merged n-bests are also used in the line search procedure to efficiently draw the error surface for efficient computation of the outer minimization of Eq. 3.

Algorithm 1 MERT

```
Initialize $\mathbf{w}^1$
for $t = 1, ..., T$ do            ▷ Or, until convergence
    Generate $n$-bests using $\mathbf{w}^t$
    Learn new $\mathbf{w}^{t+1}$ by Powell's method
end for
return $\mathbf{w}^{T+1}$
```

3 Online Rank Learning

3.1 Rank Learning

Instead of the direct task loss minimization of Eq. 3, we would like to find $\mathbf{w}$ by solving the L_2-regularized constrained minimization problem

$$\arg\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|_2^2 + \ell(\mathbf{w}; D) \tag{4}$$

where $\lambda > 0$ is a hyperparameter controlling the fitness to the data. The loss function $\ell(\cdot)$ we consider here is inspired by a pairwise ranking method (Hopkins and May, 2011) in which pairs of correct translation and incorrect translation are sampled from n-bests and suffer a hinge loss

$$\frac{1}{M(\mathbf{w}; D)} \sum_{(f,\mathbf{e}) \in D} \sum_{e^*, e'} \max\left\{0, 1 - \mathbf{w}^\top \Phi(f, e^*, e')\right\} \tag{5}$$

where

$$\begin{aligned} &e' \in \textsc{NBest}(\mathbf{w}; f) \setminus \textsc{Oracle}(\mathbf{w}; f, \mathbf{e}) \\ &e^* \in \textsc{Oracle}(\mathbf{w}; f, \mathbf{e}) \\ &\Phi(f, e^*, e') = \mathbf{h}(f, e^*) - \mathbf{h}(f, e'). \end{aligned}$$

$\textsc{NBest}(\cdot)$ is the n-best translations of f generated with the parameter $\mathbf{w}$, and $\textsc{Oracle}(\cdot)$ is a set of oracle translations chosen among $\textsc{NBest}(\cdot)$. Note that each e' (and e^*) implicitly represents a derivation consisting of a tuple (e', ϕ), where ϕ is a latent structure, i.e. phrases in a phrase-based SMT, but we omit ϕ for brevity. $M(\cdot)$ is a normalization constant which is equal to the number of paired loss terms $\Phi(f, e^*, e')$ in Equation 5. Since it is impossible to enumerate all possible translations, we follow the convention of approximating the domain of translation by n-bests. Unlike Hopkins and May (2011), we do not randomly sample from all the pairs in the n-best translations, but extract pairs by selecting one oracle translation and one other translation in the n-bests other than those in $\textsc{Oracle}(\cdot)$. Oracle translations are selected by minimizing the task loss,

$$\ell(\{e' \in \textsc{NBest}(\mathbf{w}; f^i)\}_{i=1}^{N}, \{\mathbf{e}^i\}_{i=1}^{N})$$

i.e. negative BLEU, with respect to a set of reference translations $\mathbf{e}$. In order to compute oracles with corpus-BLEU, we apply a greedy search strategy over n-bests (Venugopal, 2005). Equation 5 can be easily interpreted as a constant loss "1" for choosing a wrong translation under current parameters $\mathbf{w}$, which is in contrast with the direct task-loss used in max-margin approach to structured output learning (Taskar et al., 2004).

As an alternative, we would also consider a softmax loss (Collins and Koo, 2005) represented by

$$\frac{1}{N} \sum_{(f,\mathbf{e}) \in D} - \log \frac{Z_{\mathrm{O}}(\mathbf{w}; f, \mathbf{e})}{Z_{\mathrm{N}}(\mathbf{w}; f)} \tag{6}$$

where

$$Z_{\mathrm{O}}(\mathbf{w}; f, \mathbf{e}) = \textstyle\sum_{e^* \in \textsc{Oracle}(\mathbf{w}; f, \mathbf{e})} \exp(\mathbf{w}^\top \mathbf{f}(f, e^*))$$
$$Z_{\mathrm{N}}(\mathbf{w}; f) = \textstyle\sum_{e' \in \textsc{NBest}(\mathbf{w}; f)} \exp(\mathbf{w}^\top \mathbf{f}(f, e')).$$

Equation 6 is a log-linear model used in common NLP tasks such as tagging, chunking and named entity recognition, but differ slightly in that multiple correct translations are discriminated from the others (Charniak and Johnson, 2005).

Algorithm 2 Stochastic Gradient Descent

```
 1: $k = 1, \mathbf{w}_1 \leftarrow \mathbf{0}$
 2: for $t = 1, ..., T$ do
 3:     Choose $B_t = \{b_1^t, ..., b_K^t\}$ from $D$
 4:     for $b \in B_t$ do
 5:         Compute $n$-bests and oracles of $b$
 6:         Set learning rate $\eta_k$
 7:         $\mathbf{w}_{k+\frac{1}{2}} \leftarrow \mathbf{w}_k - \eta_k \nabla(\mathbf{w}_k; b)$
            ▷ Our proposed algorithm solve Eq. 12 or 16
 8:         $\mathbf{w}_{k+1} \leftarrow \min \left\{1, \frac{1/\sqrt{\lambda}}{\|\mathbf{w}_{k+\frac{1}{2}}\|_2}\right\} \mathbf{w}_{k+\frac{1}{2}}$
 9:         $k \leftarrow k + 1$
10:     end for
11: end for
12: return $\mathbf{w}_k$
```

3.2 Online Approximation

Hopkins and May (2011) applied a MERT-like procedure in Alg. 1 in which Equation 4 was solved to obtain new parameters in each iteration. Here, we employ stochastic gradient descent (SGD) methods as presented in Algorithm 2 motivated by Pegasos (Shalev-Shwartz et al., 2007). In each iteration, we randomly permute D and choose a set of batches $B^t = \{b_1^t, ..., b_K^t\}$ with each b_j^t consisting of N/K training data. For each batch b in B^t, we generate n-bests from the source sentences in b and compute oracle translations from the newly created n-bests (line 5) using a batch local corpus-BLEU (Haddow et al., 2011). Then, we optimize an approximated objective function

$$\arg\min_{\mathbf{w}} \frac{\lambda}{2} \|\mathbf{w}\|_2^2 + \ell(\mathbf{w}; b) \tag{7}$$

by replacing D with b in the objective of Eq. 4. The parameters $\mathbf{w}_k$ are updated by the sub-gradient of Equation 7, $\nabla(\mathbf{w}_k; b)$, scaled by the learning rate η_k (line 7). We use an exponential decayed learning rate $\eta_k = \eta_0 \alpha^{k/K}$, which converges very fast in practice (Tsuruoka et al., 2009)[1]. The sub-gradient of Eq.7 with the hinge loss of Eq. 5 is

$$\lambda \mathbf{w}_k - \frac{1}{M(\mathbf{w}_k; b)} \sum_{(f,\mathbf{e}) \in b} \sum_{e^*, e'} \Phi(f, e^*, e') \tag{8}$$

such that

$$1 - \mathbf{w}_k^\top \Phi(f, e^*, e') > 0. \tag{9}$$

We found that the normalization term by $M(\cdot)$ was very slow in convergence, thus, instead, we used $M'(\mathbf{w}; b)$, which was the number of paired loss terms satisfied the constraints in Equation 9. In the case of the softmax loss objective of Eq. 6, the sub-gradient is

$$\lambda \mathbf{w}_k \quad - \quad \frac{1}{|b|} \sum_{(f,\mathbf{e}) \in b} \frac{\partial}{\partial \mathbf{w}} \mathcal{L}(\mathbf{w}; f, \mathbf{e}) \Big|_{\mathbf{w} = \mathbf{w}_k} \tag{10}$$

[1] We set $\alpha = 0.85$ and $\eta_0 = 0.2$ which converged well in our preliminary experiments.

where $\mathcal{L}(\mathbf{w}; f, \mathbf{e}) = \log\left(Z_{\mathrm{O}}(\mathbf{w}; f, \mathbf{e})/Z_{\mathrm{N}}(\mathbf{w}; f)\right)$. After the parameter update, $\mathbf{w}_{k+\frac{1}{2}}$ is projected within the L_2-norm ball (Shalev-Shwartz et al., 2007).

Setting smaller batch size implies frequent updates to the parameters and a faster convergence. However, as briefly mentioned in Haddow et al. (2011), setting batch size to a smaller value, such as $|b| = 1$, does not work well in practice, since BLEU is devised for a corpus based evaluation, not for an individual sentence-wise evaluation, and it is not linearly decomposed into a sentence-wise score (Chiang et al., 2008a). Thus, the smaller batch size may also imply less accurate batch-local corpus-BLEU and incorrect oracle translation selections, which may lead to incorrect sub-gradient estimations or slower convergence. In the next section we propose an optimized parameter update which works well when setting a smaller batch size is impractical due to its task loss setting.

4 Optimized Online Rank Learning

4.1 Optimized Parameter Update

In line 7 of Algorithm 2, parameters are updated by the sub-gradient of each training instance in a batch b. When the sub-gradient in Equation 8 is employed, the update procedure can be rearranged as

$$\mathbf{w}_{k+\frac{1}{2}} \leftarrow (1-\lambda\eta_k)\mathbf{w}_k + \sum_{(f,\mathbf{e})\in b, e^*, e'} \frac{\eta_k}{M(\mathbf{w}_k; b)} \Phi(f, e^*, e') \tag{11}$$

in which each individual loss term $\Phi(\cdot)$ is scaled uniformly by a constant $\eta_k/M(\cdot)$.

Instead of the uniform scaling, we propose to update the parameters in two steps: First, we suffer the sub-gradient from the L_2 regularization

$$\mathbf{w}_{k+\frac{1}{4}} \leftarrow (1 - \lambda\eta_k)\mathbf{w}_k.$$

Second, we solve the following problem

$$\arg\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w} - \mathbf{w}_{k+\frac{1}{4}}\|_2^2 + \eta_k \sum_{(f,\mathbf{e})\in b, e^*, e'} \xi_{f,e^*,e'} \tag{12}$$

such that

$$\mathbf{w}^\top \Phi(f, e^*, e') \geq 1 - \xi_{f,e^*,e'}$$
$$\xi_{f,e^*,e'} \geq 0.$$

The problem is inspired by the passive-aggressive algorithm (Crammer et al., 2006) in which new parameters are derived through the tradeoff between the amount of updates to the parameters and the margin-based loss. Note that the objective in MIRA is represented by

$$\arg\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w} - \mathbf{w}_k\|_2^2 + \sum_{(f,\mathbf{e})\in b, e^*, e'} \xi_{f,e^*,e'} \tag{13}$$

If we treat $\mathbf{w}_{k+\frac{1}{4}}$ as our previous parameters and set $\lambda = 1/\eta_k$, they are very similar. Unlike MIRA, the learning rate η_k is directly used as a tradeoff parameter which decays as training proceeds, and the sub-gradient of the global L_2 regularization term is also combined in the problem through $\mathbf{w}_{k+\frac{1}{4}}$.

The Lagrangian dual of Equation 12 is

$$\arg\min_{\tau_{e^*,e'}} \frac{1}{2}\|\sum_{(f,\mathbf{e})\in b, e^*, e'} \tau_{e^*,e'}\Phi(f, e^*, e')\|_2^2 - \sum_{(f,\mathbf{e})\in b, e^*, e'} \tau_{e^*,e'} \left\{1 - \mathbf{w}_{k+\frac{1}{4}}^\top \Phi(f, e^*, e')\right\} \tag{14}$$

subject to

$$\sum_{(f,\mathbf{e})\in b, e^*, e'} \tau_{e^*,e'} \leq \eta_k.$$

We used a dual coordinate descent algorithm (Hsieh et al., 2008)[2] to efficiently solve the quadratic program (QP) in Equation 14, leading to an update

$$\mathbf{w}_{k+\frac{1}{2}} \leftarrow \mathbf{w}_{k+\frac{1}{4}} + \sum_{(f,\mathbf{e})\in b, e^*, e'} \tau_{e^*,e'}\Phi(f, e^*, e'). \tag{15}$$

When compared with Equation 11, the update procedure in Equation 15 rescales the contribution from each sub-gradient through the Lagrange multipliers $\tau_{e^*,e'}$. Note that if we set $\tau_{e^*,e'} = \eta_k/M(\cdot)$, we satisfy the constraints in Eq. 14, and recover the update in Eq. 11.

In the same manner as Eq. 12, we derive an optimized update procedure for the softmax loss, which replaces the update with Equation 10, by solving the

[2] Specifically, each parameter is bound constrained $0 \leq \tau \leq \eta_k$ but is not summation constrained $\sum \tau \leq \eta_k$. Thus, we renormalize τ after optimization.

following problem

$$\arg\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w} - \mathbf{w}_{k+\frac{1}{4}}\|_2^2 + \eta_k \sum_{(f,\mathbf{e})\in b} \xi_f \qquad (16)$$

such that

$$\mathbf{w}^\top \Psi(\mathbf{w}_k; f, \mathbf{e}) \geq -\mathcal{L}(\mathbf{w}_k; f, \mathbf{e}) - \xi_f$$
$$\xi_f \geq 0$$

in which $\Psi(\mathbf{w}'; f, \mathbf{e}) = \frac{\partial}{\partial \mathbf{w}} \mathcal{L}(\mathbf{w}; f, \mathbf{e})\big|_{\mathbf{w}=\mathbf{w}'}$. Equation 16 can be interpreted as a cutting-plane approximation for the objective of Eq. 7, in which the original objective of Eq. 7 with the softmax loss in Eq. 6 is approximated by $|b|$ linear constraints derived from the sub-gradients at point $\mathbf{w}_k$ (Teo et al., 2010). Eq. 16 is efficiently solved by its Lagrange dual, leading to an update

$$\mathbf{w}_{k+\frac{1}{2}} \leftarrow \mathbf{w}_{k+\frac{1}{4}} + \sum_{(f,\mathbf{e})\in b} \tau_f \Psi(\mathbf{w}_k; f, \mathbf{e}) \qquad (17)$$

subject to $\sum_{(f,\mathbf{e})\in b} \tau_f \leq \eta_k$. Similar to Eq. 15, the parameter update by $\Psi(\cdot)$ is rescaled by its Lagrange multipliers τ_f in place of the uniform scale of $1/|b|$ in the sub-gradient of Eq. 10.

4.2 Line Search for Parameter Mixing

For faster training, we employ an efficient parallel training strategy proposed by McDonald et al. (2010). The training data D is split into S disjoint *shards*, $\{D_1, ..., D_S\}$. Each shard learns its own parameters in each single epoch t and performs parameter mixing by averaging parameters across shards.

We propose an optimized parallel training in Algorithm 3 which performs better mixing with respect to the task loss, i.e. negative BLEU. In line 5, $\mathbf{w}^{t+\frac{1}{2}}$ is computed by averaging $\mathbf{w}^{t+1,s}$ from all the shards after local training using their own data D_s. Then, the new parameters $\mathbf{w}^{t+1}$ are obtained by linearly interpolating with the parameters from the previous epoch $\mathbf{w}^t$. The linear interpolation weight ρ is efficiently computed by a line search procedure which directly minimizes the negative corpus-BLEU. The procedure is exactly the same as the line search strategy employed in MERT using $\mathbf{w}^t$ as our starting point with the direction $\mathbf{w}^{t+\frac{1}{2}} - \mathbf{w}^t$. The idea of using the line search procedure is to find the optimum parameters under corpus-BLEU without a batch-local approximation. Unlike MERT, however, we do not memorize nor merge all the n-bests generated across iterations, but keep only n-bests in each iteration for faster training and for memory saving. Thus, the optimum ρ obtained by the line search may be suboptimal in terms of the training objective, but potentially better than averaging for minimizing the final task loss.

Algorithm 3 Distributed training with line search

```
1: $\mathbf{w}^1 \leftarrow \mathbf{0}$
2: for $t = 1, ..., T$ do
3:     $\mathbf{w}^{t,s} \leftarrow \mathbf{w}^t$                          ▷ Distribute parameters
4:     Each shard learns $\mathbf{w}^{t+1,s}$ using $D_s$
                                                  ▷ Line 3–10 in Alg. 2
5:     $\mathbf{w}^{t+\frac{1}{2}} \leftarrow 1/S \sum_s \mathbf{w}^{t+1,s}$                    ▷ Mixing
6:     $\mathbf{w}^{t+1} \leftarrow (1-\rho)\mathbf{w}^t + \rho\mathbf{w}^{t+\frac{1}{2}}$   ▷ Line search
7: end for
8: return $\mathbf{w}^{T+1}$
```

5 Experiments

Experiments were carried out on the NIST 2008 Chinese-to-English Open MT task. The training data consists of nearly 5.6 million bilingual sentences and additional monolingual data, English Gigaword, for 5-gram language model estimation. MT02 and MT06 were used as our tuning and development testing, and MT08 as our final testing with all data consisting of four reference translations.

We use an in-house developed hypergraph-based toolkit for training and decoding with synchronous-CFGs (SCFG) for hierarchical phrase-bassed SMT (Chiang, 2007). The system employs 14 features, consisting of standard Hiero-style features (Chiang, 2007), and a set of indicator features, such as the number of synchronous-rules in a derivation. Two 5-gram language models are also included, one from the English-side of bitexts and the other from English Gigaword, with features counting the number of out-of-vocabulary words in each model (Dyer et al., 2011). For faster experiments, we precomputed translation forests inspired by Xiao et al. (2011). Instead of generating forests from bitexts in each iteration, we construct and save translation forests by intersecting the source side of SCFG with input sentences and by keeping the target side of the intersected rules. n-bests are generated from the precomputed forests on the fly using the forest rescoring framework (Huang and Chiang, 2007) with additional non-local features, such as 5-gram language models.

We compared four algorithms, MERT, PRO, MIRA and our proposed online settings, online rank optimization (ORO). Note that ORO without our optimization methods in Section 4 is essentially the same as Pegasos, but differs in that we employ the algorithm for ranking structured outputs with varied objectives, hinge loss or softmax loss[3]. MERT learns parameters from forests (Kumar et al., 2009) with 4 restarts and 8 random directions in each iteration. We experimented on a variant of PRO[4], in which the objective in Eq. 4 with the hinge loss of Eq. 5 was solved in each iteration in line 4 of Alg. 1 using an off-the-shelf solver[5]. Our MIRA solves the problem in Equation 13 in line 7 of Alg. 2. For a systematic comparison, we used our exhaustive oracle translation selection method in Section 3 for PRO, MIRA and ORO. For each learning algorithm, we ran 30 iterations and generated duplicate removed 1,000-best translations in each iteration. The hyperparameter λ for PRO and ORO was set to 10^{-5}, selected from among $\{10^{-3}, 10^{-4}, 10^{-5}\}$, and 10^2 for MIRA, chosen from $\{10, 10^2, 10^3\}$ by preliminary testing on MT06. Both decoding and learning are parallelized and run on 8 cores. Each online learning took roughly 12 hours, and PRO took one day. It took roughly 3 days for MERT with 20 iterations. Translation results are measured by case sensitive BLEU.

	MT06	MT08
MERT	31.45†	24.13†
PRO	31.76†	24.43†
MIRA-L	31.42†	24.15†
ORO-L$_{\text{hinge}}$	29.76	21.96
O-ORO-L$_{\text{hinge}}$	**32.06**	**24.95**
ORO-L$_{\text{softmax}}$	30.77	23.07
O-ORO-L$_{\text{softmax}}$	31.16†	23.20

Table 1: Translation results by BLEU. Results without significant differences from the MERT baseline are marked †. The numbers in boldface are significantly better than the MERT baseline (both measured by the bootstrap resampling (Koehn, 2004) with $p > 0.05$).

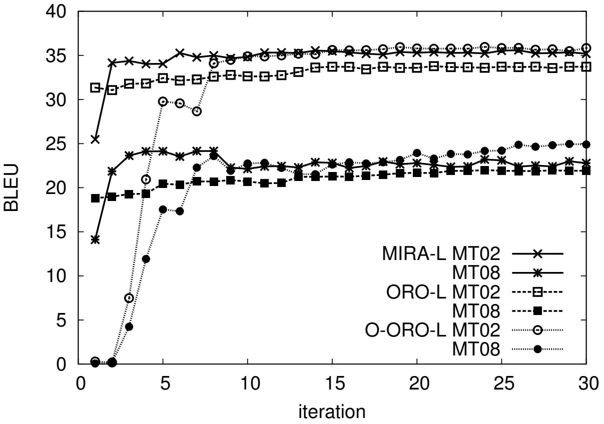

Figure 1: Learning curves for three algorithms, MIRA-L, ORO-L$_{\text{hinge}}$ and O-ORO-L$_{\text{hinge}}$.

Table 1 presents our main results. Among the parameters from multiple iterations, we report the outputs that performed the best on MT06. With Moses (Koehn et al., 2007), we achieved 30.36 and 23.64 BLEU for MT06 and MT08, respectively. We denote the “O-” prefix for the optimized parameter updates discussed in Section 4.1, and the “-L” suffix for parameter mixing by line search as described in Section 4.2. The batch size was set to 16 for MIRA and ORO. In general, our PRO and MIRA settings achieved the results very comparable to MERT. The hinge-loss and softmax objective OROs were lower than those of the three baselines. The softmax objective with the optimized update (O-ORO-L$_{\text{softmax}}$) performed better than the non-optimized version, but it was still lower than our baselines. In the case of the hinge-loss objective with the optimized update (O-ORO-L$_{\text{hinge}}$), the gain in MT08 was significant, and achieved the best BLEU.

Figure 1 presents the learning curves for three algorithms MIRA-L, ORO-L$_{\text{hinge}}$ and O-ORO-L$_{\text{hinge}}$, in which the performance is measured by BLEU

[3] The other major difference is the use of a simpler learning rate, $\frac{1}{\lambda k}$, which was very slow in our preliminary studies.

[4] Hopkins and May (2011) minimized logistic loss sampled from the merged n-bests, and sentence-BLEU was used for determining ranks.

[5] We used liblinear (Fan et al., 2008) at `http://www.csie.ntu.edu.tw/~cjlin/liblinear` with the solver type of 3.

	MT06	MT08
MIRA	30.95	23.06
MIRA-L	31.42†	24.15†
ORO$_{\text{hinge}}$	29.09	21.93
ORO-L$_{\text{hinge}}$	29.76	21.96
ORO$_{\text{softmax}}$	30.80	23.06
ORO-L$_{\text{softmax}}$	30.77	23.07
O-ORO$_{\text{hinge}}$	31.15†	23.20
O-ORO-L$_{\text{hinge}}$	**32.06**	**24.95**
O-ORO$_{\text{softmax}}$	31.40†	23.93†
O-ORO-L$_{\text{softmax}}$	31.16†	23.20

Table 2: Parameter mixing by line search.

on the training data (MT02) and on the test data (MT08). MIRA-L quickly converges and is slightly unstable in the test set, while ORO-L$_{\text{hinge}}$ is very stable and slow to converge, but with low performance on the training and test data. The stable learning curve in ORO-L$_{\text{hinge}}$ is probably influenced by our learning rate parameter $\eta_0 = 0.2$, which will be investigated in future work. O-ORO-L$_{\text{hinge}}$ is less stable in several iterations, but steadily improves its BLEU. The behavior is justified by our optimized update procedure, in which the learning rate η_k is used as a tradeoff parameter. Thus, it tries a very aggressive update at the early stage of training, but eventually becomes conservative in updating parameters.

Next, we compare the effect of line search for parameter mixing in Table 2. Line search was very effective for MIRA and O-ORO$_{\text{hinge}}$, but less effective for the others. Since the line search procedure directly minimizes a task loss, not objectives, this may hurt the performance for the softmax objective, where the margins between the correct and incorrect translations are softly penalized.

Finally, Table 3 shows the effect of batch size selected from $\{1, 4, 8, 16\}$. There seems to be no clear trends in MIRA, and we achieved BLEU score of 24.58 by setting the batch size to 8. Clearly, setting smaller batch size is better for ORO, but it is the reverse for the optimized variants of both the hinge and softmax objectives. Figure 2 compares ORO-L$_{\text{hinge}}$ and O-ORO-L$_{\text{hinge}}$ on MT02 with different batch size settings. ORO-L$_{\text{hinge}}$ converges faster when the batch size is smaller and fine tuning of the learning rate parameter will be required for a larger batch size. As discussed in Section 3, the smaller batch size means frequent updates to parameters and a faster convergence, but potentially leads to a poor performance since the corpus-BLEU is approximately computed in a local batch. Our optimized update algorithms address the problem by adjusting the tradeoff between the amount of update to parameters and the loss, and perform better for larger batch sizes with a more accurate corpus-BLEU.

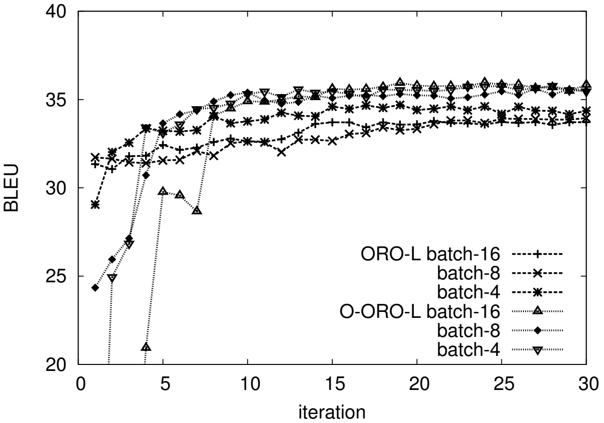

Figure 2: Learning curves on MT02 for ORO-L$_{\text{hinge}}$ and O-ORO-L$_{\text{hinge}}$ with different batch size.

6 Related Work

Our work is largely inspired by pairwise rank optimization (Hopkins and May, 2011), but runs in an online fashion similar to (Watanabe et al., 2007; Chiang et al., 2008b). Major differences come from the corpus-BLEU computation used to select oracle translations. Instead of the sentence-BLEU used by Hopkins and May (2011) or the corpus-BLEU statistics accumulated from previous translations generated by different parameters (Watanabe et al., 2007; Chiang et al., 2008b), we used a simple batch local corpus-BLEU (Haddow et al., 2011) in the same way as an online approximation to the objectives. An alternative is the use of a Taylor series approximation (Smith and Eisner, 2006; Rosti et al., 2011), which was not investigated in this paper.

Training is performed by SGD with a parameter projection method (Shalev-Shwartz et al., 2007). Slower training incurred by the larger batch size

	MT06				MT08			
batch size	1	4	8	16	1	4	8	16
MIRA-L	31.28†	31.53†	31.63†	31.42†	23.46	23.97†	**24.58**	24.15†
ORO-L$_{\text{hinge}}$	31.32†	30.69	29.61	29.76	23.63	23.12	22.07	21.96
O-ORO-L$_{\text{hinge}}$	31.44†	31.54†	31.35†	**32.06**	23.72	24.02†	24.28†	**24.95**
ORO-L$_{\text{softmax}}$	25.10	31.66†	31.31†	30.77	19.27	23.59	23.50	23.07
O-ORO-L$_{\text{softmax}}$	31.15†	31.17†	30.90	31.16†	23.62	23.31	23.03	23.20

Table 3: Translation results with varied batch size.

for more accurate corpus-BLEU is addressed by optimally scaling parameter updates in the spirit of a passive-aggressive algorithm (Crammer et al., 2006). The derived algorithm is very similar to MIRA, but differs in that the learning rate is employed as a hyperparameter for controlling the fitness to training data which decays when training proceeds. The non-uniform sub-gradient based update is also employed in an exponentiated gradient (EG) algorithm (Kivinen and Warmuth, 1997; Kivinen and Warmuth, 2001) in which parameter updates are maximum-likely estimated using an exponentially combined sub-gradients. In contrast, our approach relies on an ultraconservative update which tradeoff between the amount of updates performed to the parameters and the progress made for the objectives by solving a QP subproblem.

Unlike a complex parallelization by Chiang et al. (2008b), in which support vectors are asynchronously exchanged among parallel jobs, training is efficiently and easily carried out by distributing training data among shards and by mixing parameters in each iteration (McDonald et al., 2010). Rather than simple averaging, new parameters are derived by linearly interpolating with the previously mixed parameters, and its weight is determined by the line search algorithm employed in (Och, 2003).

7 Conclusion

We proposed a variant of an online learning algorithm inspired by a batch learning algorithm of (Hopkins and May, 2011). Training is performed by SGD with a parameter projection (Shalev-Shwartz et al., 2007) using a larger batch size for a more accurate batch local corpus-BLEU estimation. Parameter updates in each iteration is further optimized using an idea from a passive-aggressive algorithm (Crammer et al., 2006). Learning is efficiently parallelized (McDonald et al., 2010) and the locally learned parameters are mixed by an additional line search step. Experiments indicate that better performance was achieved by our optimized updates and by the more sophisticated parameter mixing.

In future work, we would like to investigate other objectives with a more direct task loss, such as max-margin (Taskar et al., 2004), risk (Smith and Eisner, 2006) or softmax-loss (Gimpel and Smith, 2010), and different regularizers, such as L_1-norm for a sparse solution. Instead of n-best approximations, we may directly employ forests for a better conditional log-likelihood estimation (Li and Eisner, 2009). We would also like to explore other mixing strategies for parallel training which can directly minimize the training objectives like those proposed for a cutting-plane algorithm (Franc and Sonnenburg, 2008).

Acknowledgments

We would like to thank anonymous reviewers and our colleagues for helpful comments and discussion.

References

Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22:39–71, March.

Phil Blunsom, Trevor Cohn, and Miles Osborne. 2008. A discriminative latent variable model for statistical machine translation. In *Proc. of ACL-08: HLT*, pages 200–208, Columbus, Ohio, June.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19:263–311, June.

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

David Chiang, Steve DeNeefe, Yee Seng Chan, and Hwee Tou Ng. 2008a. Decomposability of translation metrics for improved evaluation and efficient algorithms. In *Proc. of EMNLP 2008*, pages 610–619, Honolulu, Hawaii, October.

David Chiang, Yuval Marton, and Philip Resnik. 2008b. Online large-margin training of syntactic and structural translation features. In *Proc. of EMNLP 2008*, pages 224–233, Honolulu, Hawaii, October.

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

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proc. of ACL 2011*, pages 176–181, Portland, Oregon, USA, June.

Michael Collins and Terry Koo. 2005. Discriminative reranking for natural language parsing. *Computational Linguistics*, 31:25–70, March.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585, March.

Chris Dyer, Kevin Gimpel, Jonathan H. Clark, and Noah A. Smith. 2011. The cmu-ark german-english translation system. In *Proc. of SMT 2011*, pages 337–343, Edinburgh, Scotland, July.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, June.

Vojtěch Franc and Soeren Sonnenburg. 2008. Optimized cutting plane algorithm for support vector machines. In *Proc. of ICML '08*, pages 320–327, Helsinki, Finland.

Michel Galley and Chris Quirk. 2011. Optimal search for minimum error rate training. In *Proc. of EMNLP 2011*, pages 38–49, Edinburgh, Scotland, UK., July.

Kevin Gimpel and Noah A. Smith. 2010. Softmax-margin crfs: Training log-linear models with cost functions. In *Proc. of NAACL-HLT 2010*, pages 733–736, Los Angeles, California, June.

Barry Haddow, Abhishek Arun, and Philipp Koehn. 2011. Samplerank training for phrase-based machine translation. In *Proc. of SMT 2011*, pages 261–271, Edinburgh, Scotland, July.

Ralf Herbrich, Thore Graepel, and Klaus Obermayer. 1999. Support vector learning for ordinal regression. In *In Proc. of International Conference on Artificial Neural Networks*, pages 97–102.

Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proc. of EMNLP 2011*, pages 1352–1362, Edinburgh, Scotland, UK., July.

Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S. Sathiya Keerthi, and S. Sundararajan. 2008. A dual coordinate descent method for large-scale linear svm. In *Proc. of ICML '08*, pages 408–415, Helsinki, Finland.

Liang Huang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proc. of ACL 2007*, pages 144–151, Prague, Czech Republic, June.

Jyrki Kivinen and Manfred K. Warmuth. 1997. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63, January.

J. Kivinen and M. K. Warmuth. 2001. Relative loss bounds for multidimensional regression problems. *Machine Learning*, 45(3):301–329, December.

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 *Procc of ACL 2007*, pages 177–180, Prague, Czech Republic, June.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. of EMNLP 2004*, pages 388–395, Barcelona, Spain, July.

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 *Proc. of ACL-IJCNLP 2009*, pages 163–171, Suntec, Singapore, August.

Zhifei Li and Jason Eisner. 2009. First- and second-order expectation semirings with applications to minimum-risk training on translation forests. In *Proc. of EMNLP 2009*, pages 40–51, Singapore, August.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proc. of COLING/ACL 2006*, pages 761–768, Sydney, Australia, July.

Ryan McDonald, Keith Hall, and Gideon Mann. 2010. Distributed training strategies for the structured perceptron. In *Proc. of NAACL-HLT 2010*, pages 456–464, Los Angeles, California, June.

Robert C. Moore and Chris Quirk. 2008. Random restarts in minimum error rate training for statistical machine translation. In *Proc. of COLING 2008*, pages 585–592, Manchester, UK, August.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. of ACL 2002*, pages 295–302, Philadelphia, July.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of ACL 2003*, pages 160–167, Sapporo, Japan, July.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL 2002*, pages 311–318, Philadelphia, Pennsylvania, USA, July.

Antti-Veikko Rosti, Bing Zhang, Spyros Matsoukas, and Richard Schwartz. 2011. Expected bleu training for graphs: Bbn system description for wmt11 system combination task. In *Proc. of SMT 2011*, pages 159–165, Edinburgh, Scotland, July.

Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. 2007. Pegasos: Primal estimated sub-gradient solver for svm. In *Proc. of ICML '07*, pages 807–814, Corvalis, Oregon.

David A. Smith and Jason Eisner. 2006. Minimum risk annealing for training log-linear models. In *Proc. of COLING/ACL 2006*, pages 787–794, Sydney, Australia, July.

Ben Taskar, Dan Klein, Mike Collins, Daphne Koller, and Christopher Manning. 2004. Max-margin parsing. In *Proc. of EMNLP 2004*, pages 1–8, Barcelona, Spain, July.

Choon Hui Teo, S.V.N. Vishwanthan, Alex J. Smola, and Quoc V. Le. 2010. Bundle methods for regularized risk minimization. *Journal of Machine Learning Research*, 11:311–365, March.

Yoshimasa Tsuruoka, Jun'ichi Tsujii, and Sophia Ananiadou. 2009. Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *Proc. of ACL-IJCNLP 2009*, pages 477–485, Suntec, Singapore, August.

Ashish Venugopal. 2005. Considerations in maximum mutual information and minimum classification error training for statistical machine translation. In *Proc. of EAMT-05*, page 3031.

Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proc. of EMNLP-CoNLL 2007*, pages 764–773, Prague, Czech Republic, June.

Xinyan Xiao, Yang Liu, Qun Liu, and Shouxun Lin. 2011. Fast generation of translation forest for large-scale smt discriminative training. In *Proc. of EMNLP 2011*, pages 880–888, Edinburgh, Scotland, UK., July.