Optimization Strategies for Online Large-Margin Learning in Machine Translation

Vladimir Eidelman
UMIACS Laboratory for Computational Linguistics and Information Processing
Department of Computer Science
University of Maryland, College Park, MD
vlad@umiacs.umd.edu

Abstract

The introduction of large-margin based discriminative methods for optimizing statistical machine translation systems in recent years has allowed exploration into many new types of features for the translation process. By removing the limitation on the number of parameters which can be optimized, these methods have allowed integrating millions of sparse features. However, these methods have not yet met with wide-spread adoption. This may be partly due to the perceived complexity of implementation, and partly due to the lack of standard methodology for applying these methods to MT. This papers aims to shed light on large-margin learning for MT, explicitly presenting the simple passive-aggressive algorithm which underlies many previous approaches, with direct application to MT, and empirically comparing several widespread optimization strategies.

1 Introduction

Statistical machine translation (SMT) systems represent knowledge sources in the form of features, and rely on parameters, or weights, on each feature, to score alternative translations. As in all statistical models, these parameters need to be learned from the data. In recent years, there has been a growing trend of moving away from discriminative training using batch log-linear optimization, with Minimum-Error Rate Training (MERT) (Och, 2003) being the principle method, to online linear optimization (Chiang et al., 2008; Watanabe et al., 2007; Arun and Koehn, 2007). The major motivation for this has been that while MERT is able to efficiently optimize a small number of parameters directly toward an external evaluation metric, such as BLEU (Papineni et al., 2002), it has been shown that its performance can be erratic, and it is unable to scale to a large set of features (Foster and Kuhn, 2009; Hopkins and May, 2011). Furthermore, it is designed for batch learning, which may be prohibitive or undesirable in certain scenarios, for instance if we have a large tuning set. One or both of these limitations have led to recent introduction of alternative optimization strategies, such as minimum-risk (Smith and Eisner, 2006), PRO (Hopkins and May, 2011), Structured SVM (Cherry and Foster, 2012), and RAMPION (Gimpel and Smith, 2012), which are batch learners, and online large-margin structured learning (Chiang et al., 2009; Watanabe et al., 2007; Watanabe, 2012).

A popular method of large-margin optimization is the margin-infused relaxed algorithm (MIRA) (Crammer et al., 2006), which has been shown to perform well for machine translation, as well as other structured prediction tasks, such as parsing. (McDonald et al., 2005). This is an attractive method because we have a simple analytical solution for the optimization problem at each step, which reduces to dual coordinate descent when using 1-best MIRA. It is also quite easy to implement, as will be shown below.

Despite the proven success of MIRA-based large-margin optimization for both small and large numbers of features, these methods have not yielded wide adoption in the community. Part of the reason for this is a perception that these methods are complicated to implement, which has been cited as motivation for other work (Hopkins and May, 2011; Gimpel and Smith, 2012). Furthermore, there is a di-

Proceedings of the 7th Workshop on Statistical Machine Translation, pages 480–489,
Montréal, Canada, June 7-8, 2012.

vergence between the standard application of these methods in machine learning, and our application in machine translation (Gimpel and Smith, 2012), where in machine learning there are usually clear correct outputs and no latent structures. As a consequence of the above, there is a lack of standard practices for large-margin learning for MT, which has resulted in numerous different implementations of MIRA-based optimizers, which further add to the confusion.

This paper aims to shed light on practical concerns with online large margin training. Specifically, our contribution is first, to present the MIRA passive-aggressive update, which underlies all MIRA-based training, with an eye to application in MT. Then, we empirically compare several widespread as well as novel optimization strategies for large-margin training on Czech-to-English (cs-en) and French-to-English (fr-en) translation. Analyzing the findings, we recommend an optimization strategy which should ensure convergence and stability.

2 Large-Margin Learning

2.1 Description

MIRA is an online large-margin learner, and belongs to a class of passive-aggressive (PA) algorithms (Crammer et al., 2006). Although the exact procedure it employs is different from other subgradient optimizers, in essence it is performing a subgradient descent step, where the step size is adjusted based on each example. The underlying objective of MIRA is the same as that of the margin rescaled Structural SVM (Tsochantaridis et al., 2004; Martins et al., 2010), where we want to predict the correct output over the incorrect one by a margin at least as large as the cost incurred by predicting the incorrect output. However, the norm constraint from SVM is replaced with a proximity constraint, indicating we want to update our parameters, but keep them as close as possible to the previous parameter estimates. In the original formulation for separable classification (Crammer and Singer, 2003), if no constraints are violated, no update occurs. However, when there is a loss, the algorithm updates the parameters to satisfy the constraints. To allow for noise in the data, i.e. nonseparable instances, a slack variable ξ_i is introduced for each example, and we optimize a soft-margin. The usual presentation of MIRA is then given as:

$$\begin{aligned}&\mathbf{w_{t+1}} = \arg\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w} - \mathbf{w_t}||^2 + C\xi_i \\ &\text{s.t. } \mathbf{w}^\top \boldsymbol{f}(x_i, y_i) - \mathbf{w}^\top \boldsymbol{f}(x_i, y') \geq \text{cost}(y_i, y') - \xi_i\end{aligned} \tag{1}$$

where $\boldsymbol{f}(x_i, y_i)$ is a vector of feature functions[1], $\mathbf{w}$ is a vector of corresponding parameters, $y' \in \mathcal{Y}(x_i)$, where $\mathcal{Y}(x_i)$ is the space of possible translations we are able to produce from x,[2] and $\text{cost}(y_i, \cdot)$ is computed using an external measure of quality, such as BLEU.

The underlying structured hinge loss objective function can be rewritten as:

$$\begin{aligned}&\ell_h = -\mathbf{w}^\top \boldsymbol{f}(x_i, y_i) + \\ &\max_{y' \in \mathcal{Y}(x_i)} \Big(\mathbf{w}^\top \boldsymbol{f}(x_i, y') + \text{cost}(y_i, y')\Big)\end{aligned} \tag{2}$$

2.2 Hypothesis Selection

Our training corpus $\mathcal{T} = (x_i, y_i)_{i=1}^T$ for selecting the parameters $\mathbf{w}$ that optimize this objective consists of input sentences x_i in the source language paired with reference translations y_i in the target language. Notice that ℓ_h depends on computing the margin between $y' \in \mathcal{Y}(x_i)$ and the *correct* output, y_i. However, there is no guarantee that $y_i \in \mathcal{Y}(x_i)$ since our decoder is often incapable of producing the reference translation y_i. Since we need to have some notion of the correct output in order to compute its feature vector for the margin, in practice we revert to using surrogate references in place of y_i. These are often referred to as oracles, y^+, which are selected from the hypothesis space $\mathcal{Y}(x_i)$ of the decoder.

We are also faced with the problem of how best to select the most appropriate y' to shy away from, which we will refer to as y^-. Since optimization will proceed by setting parameters to increase the score of y^+, and decrease the score of y^-, the selection of these two hypotheses is crucial to success. The range of possibilities is presented in Eq. 3 below.

[1] More appropriately, since we only observe translations y_i, which may have many possible derivations d_j, we model the derivations as a latent variable, and our feature functions are actually computed over derivation and translation pairs $\boldsymbol{f}(x_i, y_i, d_j)$. We omit d_j for clarity.

[2] The entire hypergraph in hierarchical translation or lattice in phrase based translation.

$$\ell_r = -\max_{y^+ \in \mathcal{Y}(x_i)} \left(\gamma^+ \mathbf{w}^\top \boldsymbol{f}(x_i, y^+) - \beta^+ \text{cost}(y_i, y^+) \right) + \max_{y^- \in \mathcal{Y}(x_i)} \left(\gamma^- \mathbf{w}^\top \boldsymbol{f}(x_i, y^-) + \beta^- \text{cost}(y_i, y^-) \right) \quad (3)$$

Although this formulation has commonly been referred to as the hinge loss in previous literature, Gimpel and Smith (2012) have recently pointed out that we are in fact optimizing losses that are closer to different variants of the structured ramp loss. The difference in definition between the two is subtle, in that for the ramp loss, y_i is replaced with y^+. Each setting of $\gamma^\pm$ and $\beta^\pm$ corresponds to optimizing a different loss function. Several definitions of ℓ_r have been explored in the literature, and we discuss them below with corresponding settings of $\gamma^\pm$ and $\beta^\pm$.

In selecting y^+, we vary the settings of γ^+ and β^+. Assuming our cost function is based on BLEU, in setting $\beta^+ \rightarrow 1$ and $\gamma^+ \rightarrow 0$, if $\mathcal{Y}(x_i)$ is taken to be the entire space of possible translations, we are selecting the hypothesis with the highest BLEU overall. This is referred to in past work as max-BLEU (Tillmann and Zhang, 2006) (MB). If we approximate the search space by restricting $\mathcal{Y}(x_i)$ to a k-best list, we have the local-update (Liang et al., 2006), where we select the highest BLEU candidate from those hypotheses that the model considers good (LU). With increasing k-best size, the max-BLEU and local-update strategies begin to converge.

Setting both $\beta^+ \rightarrow 1$ and $\gamma^+ \rightarrow 1$, we obtain the cost-diminished hypothesis, which considers both the model and the cost, and corresponds to the "hope" hypothesis in Chiang et al. (2008) (M-C). This can be computed over the entire space of hypotheses or a k-best list. In a sense, this is the intuition that local-updating is after, but expressed more directly.

The alternatives for selecting y^- are quite similar. Setting $\beta^- \rightarrow 1$ and $\gamma^- \rightarrow 0$, we select the hypothesis with the highest cost (MC). Setting $\beta^- \rightarrow 0$ and $\gamma^- \rightarrow 1$, we have the highest scoring hypothesis according to the model, which corresponds to prediction-based selection (Crammer et al., 2006) (PB). Setting both to 1, we have the cost-augmented hypothesis, which is referred to as the "fear" (Chiang et al., 2008), and max-loss (Crammer et al., 2006) (M+C). This hypothesis is considered the most dangerous because it has a high model score along with a high cost.

Considering the settings for both parts of Eq. 3, γ^+, β^+ and γ^-, β^-, assigning all $\gamma^\pm$ and $\beta^\pm$ to 1 corresponds to the most commonly used loss function in MT (Gimpel and Smith, 2012; Chiang et al., 2009). This is the "hope"/"fear" pairing, where we use the cost-diminished hypothesis y^+ and cost-augmented hypothesis y^-. Other loss functions have also been explored, such as $\gamma^\pm \rightarrow 1$, $\beta^+ \rightarrow 1$, $\beta^- \rightarrow 0$ (Liang et al., 2006), and something approximating $\gamma^\pm \rightarrow 1$, $\beta^+ \rightarrow 0$, $\beta^- \rightarrow 1$ (Cherry and Foster, 2012), which is closer to the usual loss used for max-margin in machine learing. To our best knowledge, other loss functions explored below are novel to this work.

Since our external metric, BLEU, is a gain, we can think of the first term in Eq. 3 as the model score plus the BLEU score, and the second term as the model minus the BLEU score. That is, with all $\gamma^\pm$ and $\beta^\pm$ set to 1, we want y^+ to be the hypothesis with a high model score, as well as being close to the reference translation, as indicated by a high BLEU score. While for y^-, we want a high model score, but it should be far away from the reference, as indicated by a low BLEU score. The motivation for choosing y^- in this fashion is grounded in the fact that since we are penalized by this term in the ramp loss objective, we should try to optimize on it directly. In practice, we can compute the cost for both terms as (1-BLEU(y,y_i)), or use that as the cost of the first term, and after selecting y^+, compute the cost of y^- by taking the difference between BLEU(y^+,y_i) and BLEU(y,y_i).

The ramp loss objectives are non-convex, and by separately computing the $\max$ for both y^+ and y^-, we are theoretically prohibited from online learning since we are no longer guaranteed to be optimizing the desired loss. This is one motivation for the batch learner, RAMPION (Gimpel and Smith, 2012). However, as with many non-convex optimization problems in NLP, such as those involving latent variables, in practice online learning in this setting behaves quite well.

2.3 Parameter Update

The major practical concern with these methods for SMT is that oftentimes the implementation aspect is unclear, a problem which is further exacerbated by the apparent difficulty of implementation. This is further compounded with a lack of standard practices; both theoretical, such as the objective to optimize, and practical, such as efficient parallelization. The former is a result of the disconnect between the standard machine learning setting, which posits reachable references and lack of latent variables, and our own application. The latter is an active engineering problem. Both of these aspects have been receiving recent attention (McAllester et al., 2010; Mcallester and Keshet, 2011; Gimpel and Smith, 2012; McDonald et al., 2010), and although certain questions remain as to the exact loss being optimized, we now have a better understanding of the theoretical underpinnings of this method of optimization.

The first adaptations of MIRA-based learning for structured prediction in NLP utilized a set of k constraints, either for y^+, y^-, or both. This complicated the optimization by creating a QP problem with a set of linear constraints which needed to be solved with either Hildreth's algorithm or SMO style optimization, thereby precluding the possibility of a simple analytical solution. Later, Chiang (2012) introduced a cutting-plane algorithm, like that of Structural SVM's (Tsochantaridis et al., 2004), which optimizes on a small set of active constraints.

While these methods of dealing with structured prediction may perform better empirically, they come with a higher computational cost. Crammer et al. (2006) shows that satisfying the single most violated margin constraint, commonly referred to as 1-best MIRA, is amenable to a simple analytical solution for the optimization problem at each step. Furthermore, the 1-best MIRA update is conceptually and practically much simpler, while retaining most of the optimization power of the more advanced methods. Thus, this is the method we present below.

Since the MIRA optimization problem is an instance of a general structured problem with an ℓ_2 norm, the update at each step reduces to dual coordinate descent (Smith, 2011). In our soft-margin setting, this is analogous to the PA-I update of Crammer et al. (2006). In fact, this update remains largely intact as the inner core within k-best constraint or cutting plane optimization. Algorithm 1 presents the entire training regime necessary for 1-best MIRA training of a machine translation system. As can be seen, the parameter update at step 11 depends on the difference between the features of y^+ and y^-, where δ is the step size, which is controlled by the regularization parameter C; indicating how far we are willing to move at each step. $\mathcal{Y}(x_i)$ may be a k-best list or the entire space of hypotheses.[3]

Algorithm 1 MIRA Training

Require: : Training set $T = (x_i, y_i)_{i=1}^{T}$, **w**, C
1: **for** $j \leftarrow 1$ to N **do**
2: **for** $i \leftarrow 1$ to T **do**
3: $\mathcal{Y}(x_i) \leftarrow$Decode($x_i$,w)
4: $y^+ \leftarrow$ FindOracle($\mathcal{Y}(x_i)$)
5: $y^- \leftarrow$ FindPrediction($\mathcal{Y}(x_i)$)
6: margin $\leftarrow \mathbf{w}^\top \boldsymbol{f}(x_i, y^-) - \mathbf{w}^\top \boldsymbol{f}(x_i, y^+)$
7: cost $\leftarrow \text{BLEU}(y_i, y^+) - \text{BLEU}(y_i, y^-)$
8: loss = margin + cost
9: **if** loss > 0 **then**
10: $\delta \leftarrow \min\left(C, \frac{\text{loss}}{\|\boldsymbol{f}(x_i,y^+) - \boldsymbol{f}(x_i,y^-)\|^2}\right)$
11: **w**$\leftarrow$ **w**+ $\delta\left(\boldsymbol{f}(x_i, y^+) - \boldsymbol{f}(x_i, y^-)\right)$
12: **end if**
13: **end for**
14: **end for**
15: **return w**

Algorithm 2 FindOracle

Require: : $\mathcal{Y}(x_i)$
1: **if** γ^+=0 and β^+=1 **then**
2: $y^+ \leftarrow \arg\max_{y \in \mathcal{Y}(x_i)} -\text{cost}(y_i, y)$
3: **else if** $\gamma^+ = \beta^+ = 1$ **then**
4: $y^+ \leftarrow \arg\max_{y \in \mathcal{Y}(x_i)} \mathbf{w}^\top \boldsymbol{f}(x_i, y) - \text{cost}(y_i, y)$
5: **end if**
6: **return** y^+

[3] For a more in depth examination and derivation of large-margin learning in MT, see (Chiang, 2012).

Algorithm 3 FindPrediction

Require: : $\mathcal{Y}(x_i)$

if γ^-=0 and β^-=1 **then**
  $y^- \leftarrow \arg\max_{y \in \mathcal{Y}(x_i)} \text{cost}(y_i, y)$
else if γ^-=1 and β^-=0 **then**
  $y^- \leftarrow \arg\max_{y \in \mathcal{Y}(x_i)} \mathbf{w}^\top \boldsymbol{f}(x_i, y)$
else if $\gamma^- = \beta^- = 1$ **then**
  $y^- \leftarrow \arg\max_{y \in \mathcal{Y}(x_i)} \mathbf{w}^\top \boldsymbol{f}(x_i, y) + \text{cost}(y_i, y)$
end if
return y^-

3 Experiments

3.1 Setup

To empirically analyze which loss, and thereby which strategy, for selecting y^+ and y^- is most appropriate for machine translation, we conducted a series of experiments on Czech-to-English and French-to-English translation. The parallel corpora are taken from the WMT2012 shared translation task, and consist of Europarl data along with the News Commentary corpus. All data were tokenized and lowercased, then filtered for length and aligned using the GIZA++ implementation of IBM Model 4 (Och and Ney, 2003) to obtain bidirectional alignments, which were symmetrized using the grow-diag-final-and method (Koehn et al., 2003). Grammars were extracted from the resulting parallel text and used in our hierarchical phrase-based system using cdec (Dyer et al., 2010) as the decoder. We constructed a 5-gram language model from the provided English News monolingual training data as well as the English side of the parallel corpus using the SRI language modeling toolkit with modified Kneser-Ney smoothing (Chen and Goodman, 1996). This was used to create a KenLM (Heafield, 2011).

As the tuning set for both language pairs, we used the 2051 sentences in news-test2008 (NT08), and report results on the 2525 sentences of news-test2009 (NT09) and 2489 of news-test2010 (NT10).

Corpus	Sentences	Tokens	
		en	*
cs-en	764K	20.5M	17.5M
fr-en	2M	57M	63M

Table 1: Corpus statistics

pair	1	500	50k	100k
cs-en	17.9	24.9	29.4	29.7
fr-en	20.25	29.9	33.8	34.1

Table 2: Oracle score for model 1-best (baseline) and for k-best of size 500, 50k, and 100k on NT08

We approximate cost-augmented decoding by obtaining a k-best list with k=500 unique best from our decoder at each iteration, and selecting the respective hypotheses for optimization from it. To approximate max-BLEU decoding using a k-best list, we set k=50k unique best hypotheses.[4] As can be seen in Table 2, we found this size was sufficient for our purposes as increasing size led to small improvements in oracle BLEU score. C is set to 0.01.

For comparison with MERT, we create a baseline model which uses a small standard set of features found in translation systems: language model probability, phrase translation probabilities, lexical weighting probabilities, and source word, pass-through, and word penalties.

While BLEU is usually calculated at the corpus level, we need to approximate the metric at the sentence level. In this, we mostly follow previous approaches, where in the first iteration through the corpus we use a smoothed sentence level BLEU approximation, similar to Lin and Och (2004), and in subsequently iterations, the BLEU score is calculated in the context of the previous set of 1-best translations of the entire tuning set.

To make parameter estimation more efficient, some form of parallelization is preferred. While earlier versions of MIRA training had complex parallelization procedures which necessitated passing information between learners, performing iterative parameter mixing (McDonald et al., 2010) has been shown to be just as effective (Chiang, 2012). We use a simple implementation of this regime, where we divide the tuning set into n shards and distribute them amongst n learners, along with the parameter vector $\mathbf{w}$. Each learner decodes and updates parame-

[4] We are able to theoretically extract more constraints from a large list, in the spirit of k-constraints or a cutting plane, but Chiang (2012) showed that cutting plane performance is approximately 0.2-0.4 BLEU better than a single constraint, so although there is a trade off between the simplicity of a single constraint and performance, it is not substantial.

cs-en	NT09		NT10	
	LU	M-C	LU	M-C
PB	16.4	18.3	17	19.3
MC	**18.5**	16	**19.1**	17.5
M+C	17.8	**18.7**	18.4	**19.6**

Table 3: Results with different strategies on cs-en translation. MERT baseline is 18.4 for NT09 and 19.7 for NT10

fr-en	NT09		NT10	
	LU	M-C	LU	M-C
PB	20.5	23.1	22.2	25
MC	**23.9**	23	**25.8**	24.8
M+C	22.2	**23.6**	24	**25.4**

Table 4: Results with different strategies on fr-en translation. MERT baseline is 24.2 for NT09 and 26 for NT10

ters on its shard of the tuning set, and once all learners are finished, these n parameter vectors are averaged to form the initial parameter vector for the next iteration. In our experiments, n=20.

3.2 Results

The results of using different optimization strategies for cs-en and fr-en are presented in Tables 3 and 4 below. For all experiments, all settings are kept exactly the same, with the only variation being the selection of the oracle y^+ and prediction y^-. The first column in each table indicates the method for selecting the prediction, y^-. PB indicates prediction-based, MC is the hypothesis with the highest cost, and M+C is cost-augmented selection. Analogously, the headings across the table indicate oracle selection strategies, with LU indicating local updating, and M-C being cost-diminished selection.

From the cs-en results in Table 3, we can see that two settings fair the best: LU oracle selection paired with MC prediction selection (LU/MC), and M-C oracle selection paired with M+C prediction selection (M$\pm$C). On both sets, (M$\pm$C) performs better, but the results are comparable. Pairing M-C with PB is also a viable strategy, while no other pairing is successful for LU.

When comparing with MERT, note that we use a hypergraph based MERT (Kumar et al., 2009), while the MIRA updates are computed from a k-best list. For max-BLEU oracle selection paired with MC, the performance decreases substantially, to 15.4 and 16.6 BLEU on NT09 and NT10, respectively. Using the augmented k-best list did not significantly affect performance for M-C oracle selection.

For fr-en, we see much the same behavior as in cs-en. However, here LU/MC slightly outperforms M$\pm$C. From both tasks, we can see that LU is more sensitive to prediction selection, and can only optimize effectively when paired with MC. M-C on the other hand, is more forgiving, and can make progress with PB and MC, albeit not as effectively as with M+C.

3.3 Large Feature Set

Since one of the primary motivations for large-margin learning is the ability to effectively handle large quantities of features, we further evaluate the ability of the strategies by introducing a large number of sparse features into our model. We introduce sparse binary indicator features of the form commonly found in MT research (Chiang et al., 2009; Watanabe et al., 2007). Specifically, we introduce two types of features based on word alignment from hierarchical phrase pairs and a target bigram feature. The first type, a word pair feature, fires for every word pair (e_i, f_j) observed in the phrase pair. The second, insertion features, account for spurious words on the target side of a phrase pair by firing for unaligned target words, associating them with every source word, i.e. $(e_i, f_j), (e_i, f_{j+1}), etc.$. The target bigram feature fires for every pair of consecutive words on the target side (e_i, e_{i+1}). In all, we introduce 650k features for cs-en, and 1.1M for fr-en. Taking the two best performing strategies from the baseline model, LU/MC and M$\pm$C, we compare their performance with the larger feature set in Table 5.

Although integrating these features does not significantly alter the performance on either task, our purpose was to establish once again that the large-margin learning framework is capable of effectively optimizing parameters for a large number of sparse features in the MT setting.

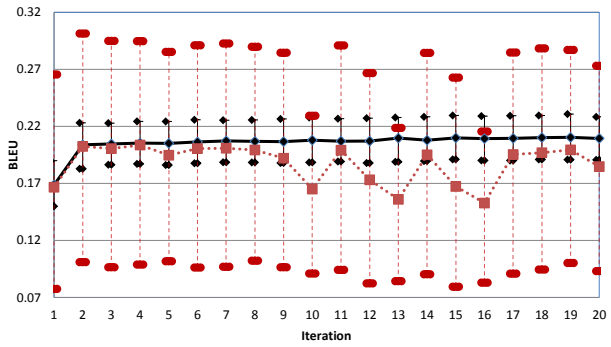

Figure 1: Comparison of performance on development set for cs-en when using LU/MC and M±C selection.

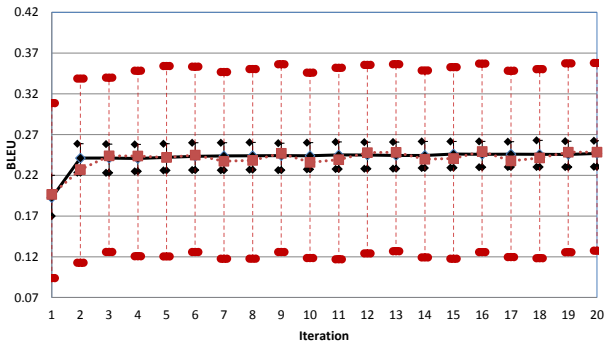

Figure 2: Comparison of performance on development set for fr-en when using LU/MC and M±C selection.

	fr-en		cs-en	
	NT09	NT10	NT09	NT10
LU/MC	23.9	25.7	18.5	19.6
M±C	23.8	25.4	18.6	19.6

Table 5: Results on cs-en and fr-en with extended feature set.

4 Discussion

Although the performance of the two strategies is competitive on the evaluation sets, this does not relay the entire story. For a more complete view of the differences between optimization strategies, we turn to Figures 1-6. Figure 1 and 2 present the comparison of performance on the NT08 development set for cs-en and fr-en, respectively, when using LU/MC to select the oracle and prediction versus M±C selection. M±C is indicated with a solid black line, while LU/MC is a dotted red line. The corpus-level oracle and prediction BLEU scores at each iteration are indicated with error bars around each point, using solid lines for M±C and dotted lines for LU/MC. As can be seen in Figure 1, while optimizing with M±C is stable and smooth, where we converge on our optimum after several iterations, optimizing with LU/MC is highly unstable. This is at least in part due to the wide range in BLEU scores for the oracle and prediction, which are in the range of 10 BLEU points higher or lower than the current model best. On the contrary, the range of BLEU scores for the M±C optimizer is on the order of 2 BLEU points, leading to more gradual changes.

We see a similar, albeit slightly less pronounced behavior on fr-en in Figure 2. M±C optimization is once again smooth, and converges quickly, with a small range for the oracle and prediction scores around the model best. LU/MC remains unstable, oscillating up to 2 BLEU points between iterations.

Figures 3-6 compare the different optimization strategies further. In Figures 3 and 5, we use M-C as the oracle, and show performance on the development set while using the three prediction selection strategies, M+C with a solid blue line, PB with a dotted green line, and MC with a dashed red line. Error bars indicate the oracle and prediction BLEU scores for each pairing as before. In all three cases, the oracle BLEU score is in about the same range, as expected, since all are using the same oracle selection strategy. We can immediately observe that PB has no error bars going down, indicating that the PB method for selecting the prediction keeps pace with the model best at each iteration. On the other hand, MC selection also stands out, since it is the only one with a large drop in prediction BLEU score. Crucially, all learners are stable, and move toward convergence smoothly, which serves to validate our earlier observation that M-C oracle selection can be paired with any prediction selection strategy and optimize effectively. In both cs-en and fr-en, we can observe that M±C performs the best.

In Figures 4 and 6, we use LU as the oracle, and show performance using the three prediction selection strategies, with each line representing the same strategy as described above. The major difference, which is immediately evident, is that the optimizers are highly unstable. The only pairing which shows some stability is LU/MC, with both the other predic-

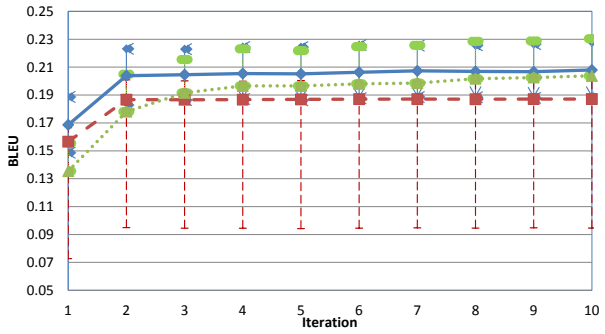

Figure 3: Comparison of performance on development set for cs-en of the three prediction selection strategies when using M-C selection as oracle.

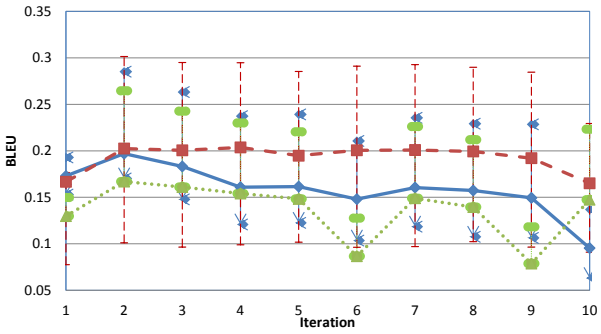

Figure 4: Comparison of performance on development set for cs-en of the three prediction selection strategies when using LU selection as oracle.

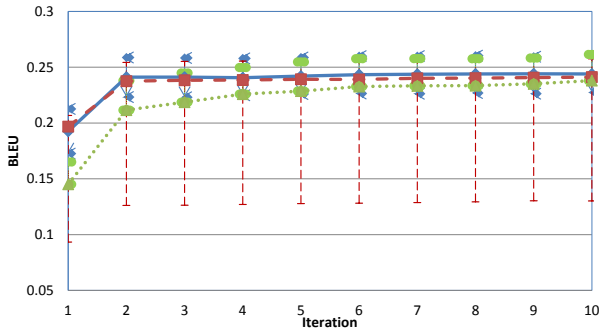

Figure 5: Comparison of performance on development set for fr-en of the three prediction selection strategies when using M-C selection as oracle.

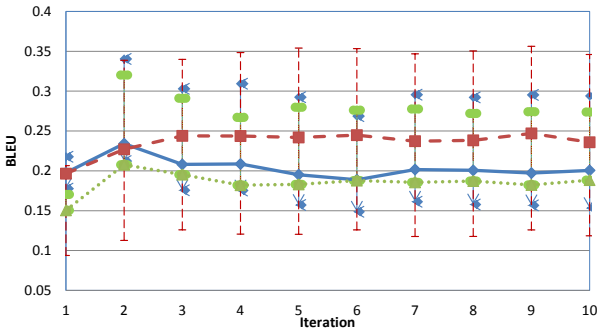

Figure 6: Comparison of performance on development set for fr-en of the three prediction selection strategies when using LU selection as oracle.

tion selection methods, PB and M+C significantly underperforming it.

Given that the translation performance of optimizing the loss functions represented by LU/MC and M±C selection is comparable on the evaluation sets for fr-en and cs-en, it may be premature to make a general recommendation for one over the other. However, taking the unstable nature of LU/MC into account, the extent of which may depend on the tuning set, as well as other factors which need to be further examined, the current more prudent alternative is selecting the oracle and prediction pair based on M±C.

5 Conclusion

In this paper, we strove to elucidate aspects of large-margin structured learning with concrete application to the MT setting. Towards this goal, we presented the MIRA passive-aggressive algorithm, which can be used directly to effectively tune a statistical MT system with millions of parameters, in the hope that some confusion surrounding MIRA-based methods may be cleared, and more MT researchers can adopt it for their own use. We then used the presented algorithm to empirically compare several widespread loss functions and strategies for selecting hypotheses for optimization. We showed that although there are two competing strategies with comparable performance, one is an unstable learner, and before we understand more regarding the nature of the instability, the preferred alternative is to use M±C as the hypothesis pair in optimization.

Acknowledgments

We would like to thank the anonymous reviewers for their comments. The author is supported by the Department of Defense through the National Defense Science and Engineering Graduate Fellow-

ship. Any opinions, findings, conclusions, or recommendations expressed are the author's and do not necessarily reflect those of the sponsors.

References

Abishek Arun and Philipp Koehn. 2007. Online learning methods for discriminative training of phrase based statistical machine translation. In *MT Summit XI*.

Stanley F. Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 310–318.

Colin Cherry and George Foster. 2012. Batch tuning strategies for statistical machine translation. In *Proceedings of NAACL*.

David Chiang, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, Honolulu, Hawaii, October.

David Chiang, Kevin Knight, and Wei Wang. 2009. 11,001 new features for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 218–226.

David Chiang. 2012. Hope and fear for discriminative training of statistical translation models. *To appear in J. Machine Learning Research*.

Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *J. Mach. Learn. Res.*, 3:951–991, March.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *J. Mach. Learn. Res.*, 7:551–585.

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

George Foster and Roland Kuhn. 2009. Stabilizing minimum error rate training. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 242–249, Athens, Greece, March. Association for Computational Linguistics.

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

Kenneth Heafield. 2011. Kenlm: faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, WMT '11, pages 187–197.

Mark Hopkins and Jonathan May. 2011. Tuning as ranking. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, Stroudsburg, PA, USA.

Shankar Kumar, Wolfgang Macherey, Chris Dyer, and Franz Och. 2009. Efficient minimum error rate training and minimum bayes-risk decoding for translation hypergraphs and lattices. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 163–171.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 761–768.

Chin-Yew Lin and Franz Josef Och. 2004. Orange: a method for evaluating automatic evaluation metrics for machine translation. In *Proceedings of the 20th international conference on Computational Linguistics*.

A. F. T. Martins, K. Gimpel, N. A. Smith, E. P. Xing, P. M. Q. Aguiar, and M. A. T. Figueiredo. 2010. Learning structured classifiers with dual coordinate descent. Technical Report CMU-ML-10-109, Carnegie Mellon University.

David Mcallester and Joseph Keshet. 2011. Generalization bounds and consistency for latent structural probit and ramp loss. In J. Shawe-Taylor, R.S. Zemel, P. Bartlett, F.C.N. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2205–2212.

David McAllester, Tamir Hazan, and Joseph Keshet. 2010. Direct loss minimization for structured prediction. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1594–1602.

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

Association for Computational Linguistics, ACL '05. Association for Computational Linguistics.

Ryan McDonald, Keith Hall, and Gideon Mann. 2010. Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 456–464, Los Angeles, California.

Franz Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. In *Computational Linguistics*, volume 29(21), pages 19–51.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

David A. Smith and Jason Eisner. 2006. Minimum risk annealing for training log-linear models. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, Sydney, Australia, July. Association for Computational Linguistics.

Noah A. Smith. 2011. *Linguistic Structure Prediction*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool, May.

Christoph Tillmann and Tong Zhang. 2006. A discriminative global training algorithm for statistical mt. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 721–728.

Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. 2004. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04.

Taro Watanabe, Jun Suzuki, Hajime Tsukada, and Hideki Isozaki. 2007. Online large-margin training for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, Prague, Czech Republic, June. Association for Computational Linguistics.

Taro Watanabe. 2012. Optimized online rank learning for machine translation. In *Proceedings of NAACL*.