Softmax-Margin Training for Structured Log-Linear Models

Kevin Gimpel   Noah A. Smith

CMU-LTI-10-008

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213
www.lti.cs.cmu.edu

{kgimpel,nasmith}@cs.cmu.edu

Abstract

We describe a method of incorporating task-specific cost functions into standard conditional log-likelihood (CLL) training of linear structured prediction models. Recently introduced in the speech recognition community, we describe the method generally for structured models, highlight connections to CLL and max-margin learning for structured prediction (Taskar et al., 2003), and show that the method optimizes a bound on risk. The approach is simple, efficient, and easy to implement, requiring very little change to an existing CLL implementation. We present experimental results comparing with several commonly-used methods for training structured predictors for named-entity recognition.[1]

1 Introduction

Conditional random fields (CRFs; Lafferty et al, 2001) and other conditional log-linear models (Berger et al., 1996) achieve strong performance for many NLP problems, but the conditional log-likelihood (CLL) criterion optimized when training these models cannot take a task-specific cost function into account.

In this report, we describe a simple approach for training conditional log-linear models with cost functions. The method, which we call **softmax-margin** training, is a generalization of similar approaches used by Sha and Saul (2006) and Povey et al. (2008) for training speech recognition models. We show how softmax-margin relates to other methods, including CLL, max-margin learning for structured prediction (Taskar et al., 2003), and risk, and provide a probabilistic interpretation for softmax-margin in the minimum divergence learning framework (Jelinek, 1997). We also derive a bound on risk via Jensen's inequality—which we call the **Jensen risk bound**—that is easier to implement and more efficient to optimize than risk directly.

We conduct experiments comparing these methods for training a discriminative model for named-entity recognition, showing a statistically significant improvement over CLL. We also experiment with various cost functions to show how training can trade off precision and recall. Finally, we discuss where softmax-margin and the Jensen risk bound fit into the landscape of training approaches for structured linear models and discuss their advantages and limitations.

2 Structured Log-Linear Models

Let $\mathcal{X}$ denote a structured input space and, for a particular $x \in \mathcal{X}$, let $\mathcal{Y}(x)$ denote a structured output space for x. The size of $\mathcal{Y}(x)$ is exponential in the size of x, which differentiates structured prediction from multiclass classification. For named-entity recognition, for example, x might be a sentence and $\mathcal{Y}(x)$ the set of all possible named-entity labelings for the sentence. Where $\mathcal{Y} = \cup_{x \in \mathcal{X}} \mathcal{Y}(x)$, we seek to learn a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ that, given an $x \in \mathcal{X}$, outputs a legal $y \in \mathcal{Y}(x)$. We use a linear model for h. That is, given a vector $\boldsymbol{f}(x, y)$ of feature functions on x and y and a parameter vector $\boldsymbol{\theta}$ containing one component for each feature, the prediction $\hat{y}$ for a new x is given by

$$\hat{y} = \operatorname*{argmax}_{y \in \mathcal{Y}(x)} \boldsymbol{\theta}^\top \boldsymbol{f}(x, y) \tag{1}$$

By exponentiating and normalizing the score $\boldsymbol{\theta}^\top \boldsymbol{f}(x, y)$, we obtain a conditional log-linear model:

$$p_{\boldsymbol{\theta}}(y|x) = \frac{\exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x, y)\}}{\sum_{y' \in \mathcal{Y}(x)} \exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x, y')\}} \tag{2}$$

[1]This is an extended technical report version of a NAACL-HLT 2010 short paper (Gimpel and Smith, 2010).

$$\text{CLL:} \quad \min_{\boldsymbol{\theta}} \sum_{i=1}^{n} -\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y^{(i)}) + \log \sum_{y \in \mathcal{Y}(x^{(i)})} \exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y)\} \tag{4}$$

$$\text{Max-Margin:} \quad \min_{\boldsymbol{\theta}} \sum_{i=1}^{n} -\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y^{(i)}) + \max_{y \in \mathcal{Y}(x^{(i)})} \Big(\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y) + \text{cost}(y^{(i)}, y)\Big) \tag{5}$$

$$\text{Risk:} \quad \min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \sum_{y \in \mathcal{Y}(x^{(i)})} \text{cost}(y^{(i)}, y) \frac{\exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y)\}}{\sum_{y' \in \mathcal{Y}(x^{(i)})} \exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y')\}} \tag{6}$$

$$\text{Softmax-Margin:} \quad \min_{\boldsymbol{\theta}} \sum_{i=1}^{n} -\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y^{(i)}) + \log \sum_{y \in \mathcal{Y}(x^{(i)})} \exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y) + \text{cost}(y^{(i)}, y)\} \tag{7}$$

Figure 1: Objective functions for training linear models. Regularization terms (e.g., $C \sum_{j=1}^{d} \theta_j^2$) are not shown here.

2.1 Training Criteria

Many criteria exist for training the weights $\boldsymbol{\theta}$. We next review three choices in detail. For the following, we assume a training set consisting of n input-output pairs $\{\langle x^{(i)}, y^{(i)} \rangle\}_{i=1}^{n}$. Some criteria will make use of a task-specific cost function that measures the extent to which a structure y differs from the true structure $y^{(i)}$, denoted by $\text{cost}(y^{(i)}, y)$.

2.1.1 Conditional Log-Likelihood

The learning problem for maximizing conditional log-likelihood is shown in Eq. 4 in Fig. 1 (we transform it into a minimization problem for easier comparison). This criterion is commonly used when a probabilistic interpretation of the model is desired.

2.1.2 Max-Margin

An alternative approach to training structured linear classifiers is based on maximum-margin Markov networks (Taskar et al., 2003). The basic idea is to choose weights such that the linear score of each $\langle x^{(i)}, y^{(i)} \rangle$ is better than $\langle x^{(i)}, y \rangle$ for all alternatives $y \in \mathcal{Y}(x^{(i)}) \setminus \{y^{(i)}\}$, with a larger margin for those y with higher cost. The "margin rescaling" form of this training criterion is shown in Eq. 5.[2] Note that the cost function is incorporated into the criterion.

2.1.3 Risk

Risk is defined as the expected value of the cost with respect to the conditional distribution $p_{\boldsymbol{\theta}}(y|x)$; on training data:

$$\textstyle\sum_{i=1}^{n} \sum_{y \in \mathcal{Y}(x^{(i)})} p_{\boldsymbol{\theta}}(y|x^{(i)}) \text{cost}(y^{(i)}, y) \tag{3}$$

With a log-linear model, learning then requires solving the problem shown in Eq. 6. Unlike the previous two criteria, risk is typically non-convex.

Risk minimization first appeared in the speech recognition community (Kaiser et al., 2000; Povey and Woodland, 2002). In NLP, Smith and Eisner (2006) minimized risk using k-best lists to define the distribution over output structures. Li and Eisner (2009) introduced a novel semiring for minimizing risk using dynamic programming; Xiong et al. (2009) minimized risk in a CRF.

[2] An alternative form is obtained via "slack rescaling" (Tsochantaridis et al., 2005), which we will not discuss further here.

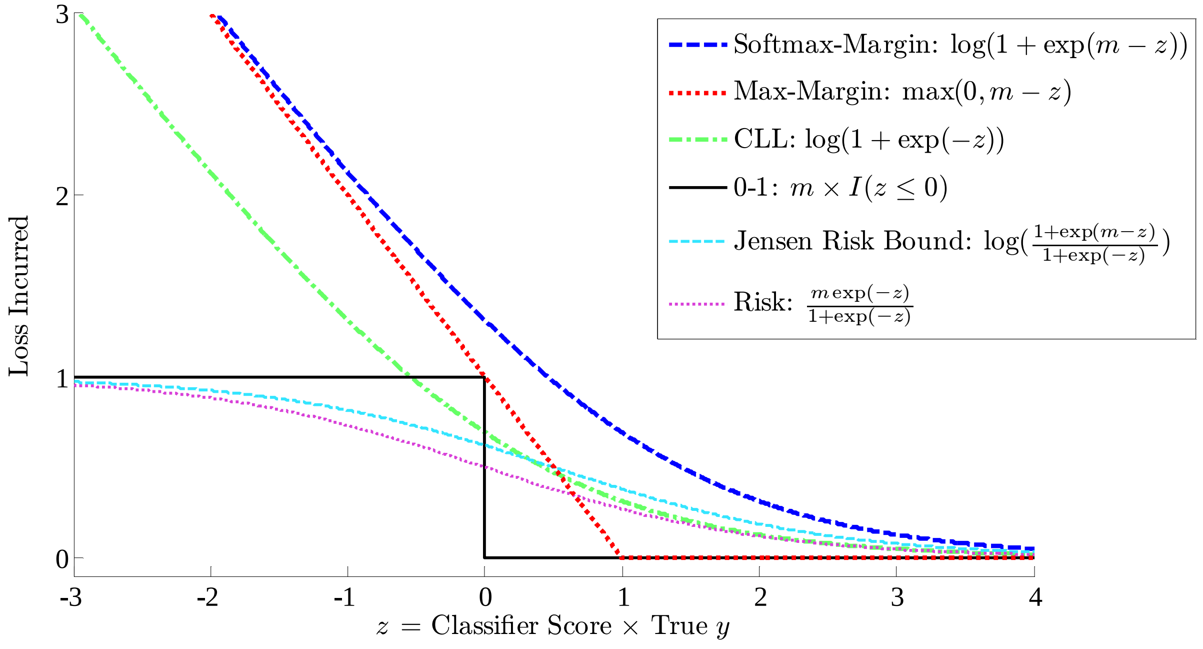

Figure 2: A plot of several loss functions for binary classification, including the loss underlying softmax-margin, which bounds all other functions from above. The constant m is the multiplier for the cost associated with making the wrong classification decision; here, $m = 1$. All objectives except CLL depend on m.

2.1.4 Other Criteria

Many other criteria have been proposed to attempt to tailor training conditions to match task-specific evaluation metrics. These include the average per-label marginal likelihood for sequence labeling (Kakade et al., 2002), minimum error-rate training for machine translation (Och, 2003), F_1 for logistic regression classifiers (Jansche, 2005), and a wide range of possible metrics for sequence labeling and segmentation tasks (Suzuki et al., 2006).

Klein and Manning (2002) compared several objectives for NLP tasks, including the sum of conditional likelihoods instead of the product as used in CLL. Altun et al. (2003) compared the use of entire-sequence and pointwise optimization for sequence labeling, as well as CLL vs. exp-loss, a loss function which considers the fraction of time that an incorrect structure was scored higher than the correct one (not to be confused with the exponential of a cost function used here).

3 Softmax-Margin

The softmax-margin objective is shown as Eq. 7 and is a generalization of that used by Povey et al. (2008) and similar to that used by Sha and Saul (2006). The simple intuition is the same as the intuition in max-margin learning: high-cost outputs for $x^{(i)}$ should be penalized more heavily. Another view says that we replace the probabilistic score inside the $\exp$ function of CLL with the "cost-augmented" score from max-margin. A third view says that we replace the "hard" maximum of max-margin with the "softmax" ($\log \sum \exp$) from CLL; hence we use the name "softmax-margin." Like CLL and max-margin, the objective is convex; a proof is given in Appendix A. Softmax-margin training has a probabilistic interpretation that can be seen by situating it within the minimum divergence framework, a generalization of the well-known maximum entropy learning framework (Jelinek, 1997). This connection is established in Appendix B.

3.1 Relation to Other Objectives

We next consider how softmax-margin upper bounds CLL, max-margin, and risk. Figure 2 plots these objective functions for the case of binary classification, i.e., $\mathcal{Y}(x) = \{-1, 1\}$ for all x. The classification decision is made based on the sign of the classifier's score for an input, so multiplying this score by the true y gives a value z, shown as the abscissa in the plot. The most readily-seen result is that the softmax-margin objective is a smooth upper bound of the "hinge" objective underlying max-margin learning. Indeed, since the softmax function is a differentiable, convex upper bound on the max function, softmax-margin is a differentiable, convex upper bound on the max-margin objective.

To relate softmax-margin to CLL and risk, we first define the following notation: let $Z_i = \sum_{y \in \mathcal{Y}(x^{(i)})} \exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y)\}$ and $p_i(\cdot) = p_{\boldsymbol{\theta}}(\cdot \mid x^{(i)})$. We ignore the L_2 penalty term below for clarity. We have:

$$\begin{aligned}
\text{Softmax-Margin} &= \sum_{i=1}^{n} \left(-\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y^{(i)}) + \log \sum_{y \in \mathcal{Y}(x^{(i)})} \exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y) + \text{cost}(y^{(i)}, y)\} \right) \\
&= \sum_{i=1}^{n} \left(-\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y^{(i)}) + \log \left\{ Z_i \sum_{y \in \mathcal{Y}(x^{(i)})} \frac{\exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y)\} \exp\{\text{cost}(y^{(i)}, y)\}}{Z_i} \right\} \right) \\
&= \sum_{i=1}^{n} \left(-\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y^{(i)}) + \log \left\{ Z_i \mathbb{E}_{p_i}[\exp\{\text{cost}(y^{(i)}, \cdot)\}] \right\} \right) \\
&= \sum_{i=1}^{n} \left(-\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y^{(i)}) + \log Z_i \right) + \sum_{i=1}^{n} \log \mathbb{E}_{p_i}[\exp\{\text{cost}(y^{(i)}, \cdot)\}] \\
&= \text{CLL} + \sum_{i=1}^{n} \log \mathbb{E}_{p_i}[\exp\{\text{cost}(y^{(i)}, \cdot)\}]
\end{aligned}$$

We now focus on the last term. Since $\log$ is concave, we can use Jensen's inequality to obtain, for all i:

$$\log \mathbb{E}_{p_i}[\exp\{\text{cost}(y^{(i)}, \cdot)\}] \geq \mathbb{E}_{p_i}[\log \exp\{\text{cost}(y^{(i)}, \cdot)\}] = \mathbb{E}_{p_i}[\text{cost}(y^{(i)}, \cdot)]$$

Therefore,

$$\sum_{i=1}^{n} \log \mathbb{E}_{p_i}[\exp\{\text{cost}(y^{(i)}, \cdot)\}] \geq \sum_{i=1}^{n} \mathbb{E}_{p_i}[\text{cost}(y^{(i)}, \cdot)] = \text{Risk}$$

Since CLL ≥ 0 and $\log \mathbb{E}_{p_i}[\exp\{\text{cost}(y^{(i)}, \cdot)\}] \geq 0$ (assuming cost is non-negative), softmax-margin is a convex bound on both risk and CLL.

3.2 Jensen Risk Bound

We note that it may also be interesting to consider minimizing the function $\sum_{i=1}^{n} \log \mathbb{E}_{p_i}[\exp\{\text{cost}(y^{(i)}, \cdot)\}]$, since it is an upper bound on risk but requires less computation for computing the gradient. We call this objective the **Jensen risk bound** (JRB) and include it in our experimental comparison below. Figure 2 plots this function for binary classification. For a log-linear model, the objective takes the following form:

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \left(\log \sum_{y \in \mathcal{Y}(x^{(i)})} \exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y) + \text{cost}(y^{(i)}, y)\} - \log \sum_{y \in \mathcal{Y}(x^{(i)})} \exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y)\} \right) \tag{8}$$

Like risk, this objective is not convex (being the difference of two convex functions), but its partial derivative with respect to a parameter θ_j for a single training pair $\langle x^{(i)}, y^{(i)} \rangle$ has an intuitive form:

$$\frac{\partial \log \mathbb{E}_{p_i}[\exp\{\text{cost}(y^{(i)}, \cdot)\}]}{\partial \theta_j} = \mathbb{E}_{p_{\boldsymbol{\theta}}^{\text{cost}}(\cdot|x^{(i)})}[f_j(x^{(i)}, \cdot)] - \mathbb{E}_{p_{\boldsymbol{\theta}}(\cdot|x^{(i)})}[f_j(x^{(i)}, \cdot)] \tag{9}$$

where $p_{\boldsymbol{\theta}}^{\text{cost}}(\cdot|x^{(i)})$ is

$$\frac{\exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, \cdot) + \text{cost}(y^{(i)}, \cdot)\}}{\sum_{y' \in \mathcal{Y}(x)} \exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y') + \text{cost}(y^{(i)}, y')\}}$$

When using this formula within gradient descent, we seek to decrease the weights of features that have larger expected values under the "cost-augmented" distribution over y than under the ordinary distribution without the cost function. The effect is that features that appear frequently in high-cost structures are penalized more heavily than features that mostly appear in low-cost structures. We note that, as with risk, the correct output structure $y^{(i)}$ is only accessed through the cost function; the feature vector for $y^{(i)}$ is not computed. This can be advantageous for situations in which the model may not be able to produce the actual $y^{(i)}$ (e.g., machine translation).

We can contrast Eq. 9 with the analogous formula for minimizing risk:

$$\frac{\partial \mathbb{E}_{p_i}[\text{cost}(y^{(i)}, \cdot)]}{\partial \theta_j} = \mathbb{E}_{p_{\boldsymbol{\theta}}(\cdot|x^{(i)})}[f_j(x^{(i)}, \cdot)\text{cost}(y^{(i)}, \cdot)] - \mathbb{E}_{p_{\boldsymbol{\theta}}(\cdot|x^{(i)})}[f_j(x^{(i)}, \cdot)]\mathbb{E}_{p_{\boldsymbol{\theta}}(\cdot|x^{(i)})}[\text{cost}(y^{(i)}, \cdot)] \tag{10}$$

The presence of expectations of products makes computing Eq. 10 difficult for structured models. In fact, Li and Eisner (2009) developed a novel semiring solely for the purpose of computing these expectations of products. While their approach is elegant, it is non-trivial to implement, and regardless of implementation difficulty, minimizing risk for structured models remains a computationally expensive endeavor. We offer the Jensen risk bound as an alternative to risk that is easier to implement, requires less computation, and performs comparably (see Section 4.3 below).

3.3 Hidden Variables

Ways of combining hidden variables with large-margin training have attracted interest in the machine learning community for several years (Zhu et al., 2008; Yu and Joachims, 2009). The softmax-margin approach provides another way to train structured latent-variable models with task-specific cost functions. Unlike the latent-variable structured SVM of Yu and Joachims (2009) that finds the highest-scoring value for the latent variables, the softmax-margin approach marginalizes over all values. Where the hidden variable ranges over values $h \in \mathcal{H}(x)$ for an input x, softmax-margin with hidden variables is

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \left(-\log \sum_{h \in \mathcal{H}_i} \exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y^{(i)}, h)\} + \log \sum_{y \in \mathcal{Y}_i} \sum_{h \in \mathcal{H}_i} \exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y, h) + \text{cost}(y^{(i)}, y)\} \right)$$

where we have used $\mathcal{H}_i$ for $\mathcal{H}(x^{(i)})$ and $\mathcal{Y}_i$ for $\mathcal{Y}(x^{(i)})$. We have assumed that cost does not depend on the hidden variable, as is commonly the case in NLP. If this is not the case, cost should be included in the numerator term and extended to take h as an additional argument. Like CLL, extending softmax-margin with hidden variables causes it to be non-convex.

3.4 Implementation

Most methods for training structured models with cost functions require the cost function to decompose across the pieces of the structure in the same way as the features, such as the standard methods for maximizing margin and minimizing risk (Taskar et al., 2003; Li and Eisner, 2009). If the same conditions hold,

ℓ_{i-1}	s_j^*	$(i == n)$?	$w_i w_{i+1}$	$p_{i-1} w_i^*$	$s_{i-2} s_{i-1}$
w_j^*	p_j^*	$w_{i-2} w_{i-1}$	$p_{i-2} p_{i-1}$	$p_i p_{i+1}$	$s_{i-1} s_i^*$
o_j^*	$(i == 1)$?	$w_{i-1} w_i^*$	$p_{i-1} p_i^*$	$w_i p_{i+1}$	$s_i s_{i+1}$

Length-k prefix of w_j, $k \in \{1, 2, 3, 4\}^*$	does w_j contain digits and alphabetic characters?*
Length-k suffix of w_j, $k \in \{1, 2, 3, 4\}^*$	does w_j contain digits and '/'?*
is w_j a $\{2, 4\}$-digit number?*	does w_j contain digits and ','?*
is w_j a different number?*	does w_j contain digits and '.'?*
does w_j start with a capital letter?*	w_i in gazetteer for ℓ_i
is w_j all capital letters?*	$w_{i-1} w_i$ in gazetteer for $\ell_{i-1} \ell_i$ (if $\ell_{i-1} == \ell_i$)
does w_j start with a capital letter followed by '.'?*	Disjunction of the previous two features

Table 1: Feature templates used for NER. Let $w_1 w_2 \ldots w_n$ be the sentence under consideration and let i be the position of the current word. We use ℓ for labels, o for lowercased words, s for word "shapes," and p for POS tags. Also, $j \in \{i-1, i, i+1\}$ and all features include the current label ℓ_i. *Indicates that there are two versions, one with ℓ_{i-1} and one without.

softmax-margin training can be implemented atop standard CRF training simply by adding additional "features" to encode the local cost components, *only* when computing the partition function during training.[3] The weights of these "cost features" are not learned.

4 Experiments

We consider the problem of named-entity recognition (NER) and use the English data from the CoNLL 2003 shared task (Tjong Kim Sang and De Meulder, 2003). The data consist of news articles annotated with four entity types: person, location, organization, and miscellaneous. Our experiments focus on comparing training objectives for structured sequential models for this task. For all objectives, we use the same standard set of feature templates, following Kazama and Torisawa (2007) with additional token shape features like those in Collins (2002b) and simple gazetteer features; Table 1 shows the full set of feature templates we used. A feature was included if it occurred at least once in training data (total 1,312,255 features).

4.1 Cost Functions

The task is evaluated using the F_1 score, which is the harmonic mean of precision and recall (computed at the level of entire entities). Since this metric is computed from corpus-level precision and recall, it is not easily decomposable into features used in standard chain CRFs.

However, we may not want to directly optimize F_1 during training. In a blog posting,[4] C. Manning made the observation that optimizing F_1 would cause the model to prefer situations in which it misses an entire entity (which harms recall only) to situations in which part of an entity is labeled correctly (which harms both precision and recall). This may create contradictions for the learner during training. In addition, we believe that identifying entities partially correctly might be more useful for some downstream applications than getting them completely incorrect.

So, we propose a different cost function for optimizing models for NER that avoids this problem and fits the independence assumptions made by the model. For a given word w_j, its true label y_j, and its label under consideration $\hat{y}_j$, we define a **precision error** as the event $A_j = \{y_j = O \wedge \hat{y}_j \neq O\}$, a **recall error** as the event $B_j = \{y_j \neq O \wedge \hat{y}_j = O\}$, and an **entity error** as the event $C_j = \{y_j \neq O \wedge \hat{y}_j \neq O \wedge y_j \neq \hat{y}_j\}$. Our cost functions take the following form:

$$\text{cost}(y, \hat{y}) = \textstyle\sum_{j=1}^{|y|} \alpha I[A_j] + \beta I[B_j] + \gamma I[C_j]$$

[3]Since $\text{cost}(y^{(i)}, y^{(i)}) = 0$ by definition, these "features" will never fire for the numerator and can be ignored.

[4]`http://nlpers.blogspot.com/2006/08/doing-named-entity-recognition-dont.html`

Method	Dev.	Test	(C, m, avg.?)
Perceptron	90.48	83.98	(Y)
MIRA	91.13	85.72	(0.01, 20, Y)
CLL	90.79	85.46	(0.01, N)
Max-Margin	91.17	85.28	(0.01, 1, Y)
Risk	91.14	85.59	(0.01, 10, N)
Jensen Risk Bound	91.05	85.65	(0.01, 1, N)
Softmax-Margin	91.30	85.84	(0.01, 5, N)

Table 2: Results on development and test sets, along with hyperparameter values chosen using development set.

We obtain Hamming cost by setting $\alpha = \beta = \gamma$. To penalize one type of error more than the others, we can set its coefficient appropriately.

4.2 Baselines

We compared softmax-margin to several baselines: the structured perceptron (Collins, 2002a), 1-best MIRA with cost-augmented inference (Crammer et al., 2006), CLL, max-margin, risk, and our Jensen risk bound (JRB) introduced above.

We used L_2 regularization, experimenting with several coefficients C for each method. For MIRA, we used $C \in \{0.001, 0.01, 0.1, 1\}$. For CLL, softmax-margin, max-margin, risk, and JRB we used $C \in \{0.0, 0.01, 0.1, 1\}$. Since risk and JRB are non-convex, we initialized with the final model from CLL training that performed best on the development data (which was the CLL model with $C = 0.01$).[5] All methods except CLL and the perceptron make use of a cost function, for which we used Hamming cost for all experiments unless reported otherwise. We experimented with different fixed multipliers $m = \alpha = \beta = \gamma$ for the Hamming cost, for $m \in \{1, 5, 10, 20\}$. We also experimented with varying α and β for softmax-margin while keeping γ fixed.

The hyperparameters C and m were tuned on the development data and the best-performing combination was used to label the test data. We also tuned the decision to average parameters across all training iterations; this has generally been found to help the perceptron and MIRA, but in our experiments had mixed results for the other methods.

We ran 100 iterations through the training data for each method. For CLL, softmax-margin, risk, and JRB, we used stochastic gradient descent with a fixed step size of 0.01. For max-margin, we used stochastic subgradient descent (Ratliff et al., 2006) also with a fixed step size of 0.01.[6] For the perceptron and MIRA, we used their built-in step size formulas.

4.3 Results

Table 2 shows our results. On test data, softmax-margin is statistically indistinguishable from MIRA, risk, and JRB, but performs significantly better than CLL, max-margin, and the perceptron ($p < 0.03$, paired bootstrap with 10,000 samples; Koehn, 2004). It may be surprising that an improvement of 0.38 in F_1 could be significant, but this indicates that the improvements are not limited to certain categories of phenomena in a small number of sentences but rather appear throughout the majority of the test set.

Figure 3 shows results on the development set when using different weights for each type of error and using $C = 0.01$. We fixed the weight for entity errors (γ) at 5 and varied the weights for precision and recall errors, finding that we can trade precision and recall by training with different cost functions.

[5]When using initialization of all ones for risk and JRB, results were several points below the results here, and with all zeroes, learning failed, resulting in 0.0 F-measure on dev data. Thus, risk and JRB appear sensitive to model initialization.

[6]In preliminary experiments, we tried other fixed and decreasing step sizes for (sub)gradient descent and found that a fixed step

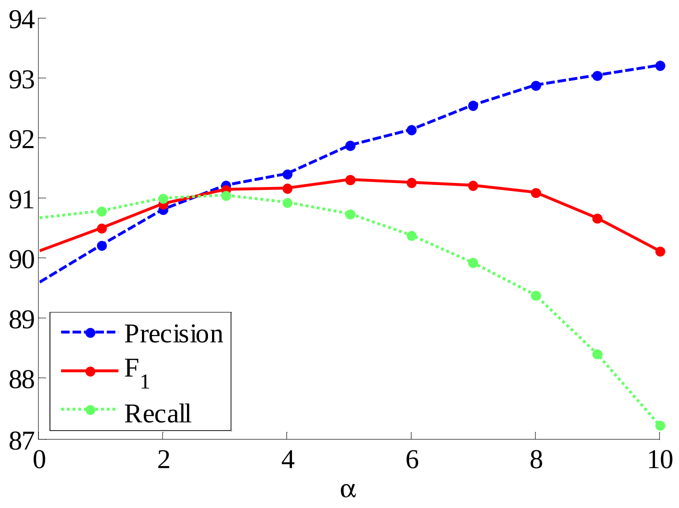

Figure 3: Results on development set when trading precision and recall via different cost functions ($\beta = 10 - \alpha$ and $\gamma = 5$).

Objective/Algorithm	Requirements	Cost Function	Convex	Probabilistic Interpretation
Perceptron	decoding		N/A	
MIRA	cost-augmented decoding	✓	✓	
CLL	summing		✓	✓
Max-Margin	cost-augmented decoding	✓	✓	
Risk	expectations of products	✓		✓
Jensen Risk Bound	cost-augmented summing	✓		✓
Softmax-Margin	cost-augmented summing	✓	✓	✓

Table 3: Comparison of objective functions.

5 Discussion

Table 3 shows a comparison of objectives/algorithms in terms of their properties and requirements. The perceptron, MIRA, and max-margin learning only require decoding, making them attractive in terms of both implementation difficulty and computational requirements (since decoding is typically faster than summing). However, they do not offer a probabilistic interpretation, which can be useful for computing posteriors or learning in the presence of hidden variables.

The softmax-margin approach offers (1) a convex objective, (2) the ability to incorporate task-specific cost functions, and (3) a probabilistic interpretation (which supports, e.g., hidden-variable learning and computation of posteriors). In contrast, max-margin training and MIRA do not provide (3); risk and JRB do not provide (1); and CLL does not support (2). Furthermore, softmax-margin training improves over standard CLL training of CRFs, is straightforward to implement, and requires the same amount of computation as CLL.

Aside from these desiderata, an advantage of risk and JRB over the other methods is that they do not require a model that can assign nonzero score to all $y^{(i)}$ in a training set. This is advantageous for tasks in which the true output is often unreachable by the model, e.g., machine translation. For these situations, we offer the Jensen risk bound, which is easier to implement and faster to train than risk, yet gives comparable

of 0.01 consistently performed well across training objectives, so we used it for all settings for simplicity.

performance.

The primary limitation of all these approaches, including softmax-margin, is that they only support cost functions that factor in the same way as the features of the model. Future work might exploit approximate inference for more expressive cost functions.

Acknowledgments

We thank three anonymous reviewers, John Lafferty, and André Martins for helpful comments and feedback on this work. This research was supported by NSF grant IIS-0844507.

A Convexity of Softmax-Margin

Convexity of Eq. 7 can be shown in several ways. One simple proof proceeds by showing softmax-margin to be the projection of CLL onto a subset of its parameters. Starting with a conditional log-linear model (i.e., Eq. 2) with k features, define the $(k+1)$th feature as the function $\text{cost}(y^*, \cdot)$.[7] Define $g(\boldsymbol{\theta}_{1:k}, \theta_{k+1})$ as the CLL objective for this model, so g is convex in its parameters. Define the set $\mathcal{M} \triangleq \{m\}$, for some $m \in \mathbb{R}$; m is the fixed weight for the cost function. We make use of the fact that the function

$$g'(\boldsymbol{\theta}_{1:k}) = \inf_{\theta \in \mathcal{M}} g(\boldsymbol{\theta}_{1:k}, \theta)$$

is convex in $\boldsymbol{\theta}_{1:k}$ (Boyd and Vandenberghe, 2004). Since $\mathcal{M}$ contains only one element, we have $g'(\boldsymbol{\theta}_{1:k}) = g(\boldsymbol{\theta}_{1:k}, m)$. The function g' is convex and, by construction, $g'(\boldsymbol{\theta}_{1:k})$ equals the softmax-margin objective. $\square$

B Minimum Divergence

In order to provide a probabilistic interpretation for softmax-margin, we relate it to the framework of minimum divergence learning (Jelinek, 1997), which is a generalization of the well-known maximum entropy framework (Berger et al., 1996).

The minimum divergence problem for conditional models can be posed as follows:

$$p^* = \operatorname*{argmin}_{p \in \mathbb{P}_y} \sum_{i=1}^{n} D(p(\cdot|x^{(i)}) \| q(\cdot|x^{(i)})) \tag{11}$$

$$\text{subject to} \sum_{i=1}^{n} \mathbb{E}_{p(\cdot|x^{(i)})}[\boldsymbol{f}(x^{(i)}, \cdot)] = \sum_{i=1}^{n} \boldsymbol{f}(x^{(i)}, y^{(i)}) \tag{12}$$

$$\sum_{y \in \mathcal{Y}(x^{(i)})} p(y|x^{(i)}) = 1, \ \forall i \tag{13}$$

$$p(y|x^{(i)}) \geq 0, \ \forall i, \forall y \in \mathcal{Y}(x^{(i)}) \tag{14}$$

where $\mathbb{P}_y$ is the set of all distributions with support equal to $\mathcal{Y}$ and $D(p\|q)$ is the Kullback-Leibler divergence between p and q. That is, the goal of minimum divergence learning is to find a distribution p^* that is as close as possible to q subject to the constraints that the feature expectations match the empirical feature values. The distribution q remains fixed and is frequently referred to as the **base distribution**. The maximum entropy problem is a special case of minimum divergence in which the base distribution is uniform. Maximum entropy modeling has been used extensively in NLP, while applications of minimum divergence learning have been less common; examples of the latter include language modeling (Jelinek, 1997; Berger and Printz, 1998) and machine translation (Foster, 2000).

[7]While cost considers more information than the other features, this is not problematic for showing convexity.

It is well-known that the dual of the maximum entropy problem is maximum likelihood for a log-linear model (Berger et al., 1996; Smith, forthcoming). With small change to that derivation, it can be shown that the dual of the minimum divergence problem for conditional models is maximum likelihood for the following model (Jelinek, 1997):

$$p(y|x) = \frac{q(y|x)\exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x,y)\}}{\sum_{y' \in \mathcal{Y}(x)} q(y'|x)\exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x,y')\}} \tag{15}$$

The maximum likelihood problem for this model (and the dual of the minimum divergence problem) is

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} -\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y^{(i)}) + \log \sum_{y \in \mathcal{Y}(x^{(i)})} q(y|x^{(i)}) \exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x^{(i)}, y)\} \tag{16}$$

Note that Eq. 16 does not contain the base distribution $q(y^{(i)}|x^{(i)})$ in the numerator term, because it separates out via the $\log$ function and can be removed because it does not depend on $\boldsymbol{\theta}$.

The softmax-margin approach can be seen as minimum divergence with a base distribution q' that depends on the true output y^* in addition to x:

$$q'(y|x, y^*) = \frac{\exp\{\text{cost}(y^*, y)\}}{\sum_{y' \in \mathcal{Y}(x)} \exp\{\text{cost}(y^*, y')\}} \tag{17}$$

Note that this base distribution is not useful for classification since it favors outputs y that have high cost. Ordinarily, when a model learned with minimum divergence is used for prediction, the output structure is chosen according to Eq. 15. However, when training with softmax-margin, prediction simply chooses the output structure according to $\exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x,y)\}$ (Eq. 1), ignoring q.

Typically, minimum divergence is chosen over maximum entropy when a base distribution is available that serves as a useful starting point for the modeling task of interest. For example, when building a domain-specific language model, one might wish to minimize divergence to a broad-domain language model while satisfying certain constraints obtained from the domain-specific data (Jelinek, 1997). Our motivation for softmax-margin training departs from this notion. The base distribution *inflates* the model scores of high-cost output structures, while the training procedure aims to *decrease* their scores. In this case, the base distribution acts as a handicap, forcing the training procedure to work extra hard to penalize high-cost outputs. Once the base distribution is removed at prediction time, the remaining score (i.e., $\exp\{\boldsymbol{\theta}^\top \boldsymbol{f}(x,y)\}$) is strongly biased away from these high-cost outputs. Broadly, the intuition is similar to that of max-margin training for structured prediction, while here the margin constraints are captured through the use of a particular base distribution in the minimum divergence framework.

References

Y. Altun, M. Johnson, and T. Hofmann. 2003. Investigating loss functions and optimization methods for discriminative learning of label sequences. In *Proc. of EMNLP*.

A. Berger and H. Printz. 1998. A comparison of criteria for maximum entropy/minimum divergence feature selection. In *Proc. of EMNLP*.

A. Berger, V. J. Della Pietra, and S. A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

S. Boyd and L. Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press.

M. Collins. 2002a. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proc. of EMNLP*.

M. Collins. 2002b. Ranking algorithms for named-entity extraction: Boosting and the voted perceptron. In *Proc. of ACL*.

K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

G. Foster. 2000. A maximum entropy/minimum divergence translation model. In *Proc. of ACL*.

K. Gimpel and N. A. Smith. 2010. Softmax-margin CRFs: Training log-linear models with cost functions. In *Proc. of NAACL-HLT*.

M. Jansche. 2005. Maximum expected F-measure training of logistic regression models. In *Proc. of HLT-EMNLP*.

F. Jelinek. 1997. *Statistical methods for speech recognition*. MIT Press.

J. Kaiser, B. Horvat, and Z. Kacic. 2000. A novel loss function for the overall risk criterion based discriminative training of HMM models. In *Proc. of ICSLP*.

S. Kakade, Y. W. Teh, and S. Roweis. 2002. An alternate objective function for Markovian fields. In *Proc. of ICML*.

J. Kazama and K. Torisawa. 2007. A new perceptron algorithm for sequence labeling with non-local features. In *Proc. of EMNLP-CoNLL*.

D. Klein and C. D. Manning. 2002. Conditional structure vs. conditional estimation in NLP models. In *Proc. of EMNLP*.

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

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*.

Z. Li and J. Eisner. 2009. First- and second-order expectation semirings with applications to minimum-risk training on translation forests. In *Proc. of EMNLP*.

F. J. Och. 2003. Minimum error rate training for statistical machine translation. In *Proc. of ACL*.

D. Povey and P. C. Woodland. 2002. Minimum phone error and I-smoothing for improved discrimative training. In *Proc. of ICASSP*.

D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah. 2008. Boosted MMI for model and feature space discriminative training. In *Proc. of ICASSP*.

N. Ratliff, J. A. Bagnell, and M. Zinkevich. 2006. Subgradient methods for maximum margin structured learning. In *ICML Workshop on Learning in Structured Output Spaces*.

F. Sha and L. K. Saul. 2006. Large margin hidden Markov models for automatic speech recognition. In *Proc. of NIPS*.

D. A. Smith and J. Eisner. 2006. Minimum risk annealing for training log-linear models. In *Proc. of COLING-ACL*.

N. A. Smith. forthcoming. *Linguistic Structure Prediction*. Synthesis Lectures on Human Language Technologies. Morgan and Claypool.

J. Suzuki, E. McDermott, and H. Isozaki. 2006. Training conditional random fields with multivariate evaluation measures. In *Proc. of COLING-ACL*.

B. Taskar, C. Guestrin, and D. Koller. 2003. Max-margin Markov networks. In *Proc. of NIPS*.

E. F. Tjong Kim Sang and F. De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proc. of CoNLL*.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6.

Y. Xiong, J. Zhu, H. Huang, and H. Xu. 2009. Minimum tag error for discriminative training of conditional random fields. *Information Sciences*, 179(1-2):169–179.

C. J. Yu and T. Joachims. 2009. Learning structural SVMs with latent variables. In *Proc. of ICML*.

J. Zhu, E. P. Xing, , and B. Zhang. 2008. Partially observed maximum entropy discrimination Markov networks. In *Proc. of NIPS*.