Minimum-Risk Training of Approximate CRF-Based NLP Systems

Veselin Stoyanov and **Jason Eisner**
HLTCOE and Center for Language and Speech Processing
Johns Hopkins University
Baltimore, MD 21218
{ves, jason}@cs.jhu.edu

Abstract

Conditional Random Fields (CRFs) are a popular formalism for structured prediction in NLP. It is well known how to train CRFs with certain topologies that admit exact inference, such as linear-chain CRFs. Some NLP phenomena, however, suggest CRFs with more complex topologies. Should such models be used, considering that they make exact inference intractable? Stoyanov et al. (2011) recently argued for training parameters to minimize the task-specific loss of whatever *approximate* inference and decoding methods will be used at test time. We apply their method to three NLP problems, showing that (i) using more complex CRFs leads to improved performance, and that (ii) minimum-risk training learns more accurate models.

1 Introduction

Conditional Random Fields (CRFs) (Lafferty et al., 2001) are often used to model dependencies among linguistic variables. CRF-based models have improved the state of the art in a number of natural language processing (NLP) tasks ranging from part-of-speech tagging to information extraction and sentiment analysis (Lafferty et al., 2001; Peng and McCallum, 2006; Choi et al., 2005).

Robust and theoretically sound training procedures have been developed for CRFs when the model can be used with exact inference and decoding.[1] However, some NLP problems seem to call for higher-treewidth graphical models in which exact inference is expensive or intractable. These "loopy" CRFs have cyclic connections among the output and/or latent variables. Alas, standard learning procedures assume exact inference: they do not compensate for approximations that will be used at test time, and can go surprisingly awry if approximate inference is used at training time (Kulesza and Pereira, 2008).

While NLP research has been consistently evolving toward more richly structured models, one may hesitate to add dependencies to a graphical model if there is a danger that this will end up *hurting* performance through approximations. In this paper we illustrate how to address this problem, even for extremely interconnected models in which every pair of output variables is connected.

Wainwright (2006) showed that if approximate inference will be used at test time, it may be beneficial to use a learning procedure that does not converge to the true model but to one that performs well *under the approximations*. Stoyanov et al. (2011) argue for minimizing a certain non-convex training objective, namely the empirical risk of the entire system comprising the CRF *together* with whatever approximate inference and decoding procedures will be used at test time. They regard this entire system as simply a complex decision rule, analogous to a neural network, and show how to use back-propagation to tune its parameters to locally minimize the empirical risk (i.e., the average task-specific loss on training data). Stoyanov et al. (2011) show that

[1] "Inference" typically refers to computing posterior marginal or max-marginal probability distributions of output random variables, given some evidence. "Decoding" derives a single structured output from the results of inference.

on certain *synthetic-data* problems, this frequentist training regimen significantly reduced test-data loss compared to approximate maximum likelihood estimation (MLE). However, this method has not been evaluated on *real-world* problems until now.

We will refer to the Stoyanov et al. (2011) approach as "ERMA"—Empirical Risk Minimization under Approximations. ERMA is attractive for NLP because the freedom to use arbitrarily structured graphical models makes it possible to include latent linguistic variables, predict complex structures such as parses (Smith and Eisner, 2008), and do collective prediction in relational domains (Ji and Grishman, 2011; Benson et al., 2011; Dreyer and Eisner, 2009). In training, ERMA considers not only the approximation method but also the task-specific loss function. This means that ERMA is careful to use the additional variables and dependencies only in ways that help training set performance. (Overfitting on the enlarged parameter set should be avoided through regularization.)

We have developed a simple syntax for specifying CRFs with complex structures, and a software package (available from `http://www.clsp.jhu.edu/~ves/software.html`) that allows ERMA training of these CRFs for several popular loss functions (e.g., accuracy, mean-squared error, F-measure). In this paper, we use these tools to revisit three previously studied NLP applications that can be modeled naturally with approximate CRFs (we will use **approximate CRFs** to refer to CRF-based systems that are used with approximations in inference or decoding). We show that (i) natural language can be modeled more effectively with CRFs that are not restricted to a linear structure and (ii) that ERMA training represents an improvement over previous learning methods.

The first application, predicting congressional votes, has not been previously modeled with CRFs. By using a more principled probabilistic approach, we are able to improve the state-of-the-art accuracy from 71.2% to 78.2% when training to maximize the approximate log-likelihood of the training data. By switching to ERMA training, we improve this result further to 85.1%.

The second application, information extraction from seminar announcements, has been modeled previously with skip-chain CRFs (Sutton and McCallum, 2005; Finkel et al., 2005). The skip-chain CRF introduces loops and requires approximate inference, which motivates minimum risk training. Our results show that ERMA training improves F-measures from 89.5 to 90.9 (compared to 87.1 for the model without skip-chains).

Finally, for our third application, we perform collective multi-label text classification. We follow previous work (Ghamrawi and McCallum, 2005; Finley and Joachims, 2008) and use a fully connected CRF to model all pairwise dependencies between labels. We observe similar trends for this task: switching from a maximum entropy model that does not model label dependencies to a loopy CRF leads to an improvement in F-measure from 81.6 to 84.0, and using ERMA leads to additional improvement (84.7).

2 Preliminaries

2.1 Conditional Random Fields

A **conditional random field** (CRF) is an undirected graphical model defined by a tuple $(\mathcal{X}, \mathcal{Y}, \mathcal{F}, f, \theta)$. $\mathcal{X} = (X_1, X_2, \ldots)$ is a set of random variables and $\mathcal{Y} = (Y_1, Y_2, \ldots)$ is a set of output random variables.[2] We use $\mathbf{x} = (x_1, x_2, \ldots)$, to denote a possible assignment of values to $\mathcal{X}$, and similarly for $\mathbf{y}$, with $\mathbf{xy}$ denoting the joint assignment. Each $\alpha \in \mathcal{F}$ is a subset of the random variables, $\alpha \subseteq \mathcal{X} \cup \mathcal{Y}$, and we write $\mathbf{xy}_\alpha$ to denote the restriction of $\mathbf{xy}$ to α. Finally, for each $\alpha \in \mathcal{F}$, the CRF specifies a function $\vec{f}_\alpha$ that extracts a feature vector $\in \mathbb{R}^d$ from the restricted assignment $\mathbf{xy}_\alpha$. We define the overall feature vector $\vec{f}(\mathbf{x}, \mathbf{y}) = \sum_{\alpha \in \mathcal{F}} \vec{f}_\alpha(\mathbf{xy}_\alpha) \in \mathbb{R}^d$. The model defines conditional probabilities

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{\exp \vec{\theta} \cdot \vec{f}(\mathbf{x}, \mathbf{y})}{\sum_{\mathbf{y}'} \exp \vec{\theta} \cdot \vec{f}(\mathbf{x}, \mathbf{y}')} \qquad (1)$$

where $\vec{\theta} \in \mathbb{R}^d$ is a global weight vector (to be learned). This is a log-linear model; the denominator (traditionally denoted $Z_\mathbf{x}$) sums over all possible output assignments to normalize the distribution.

Provided that all probabilities needed at training or test time are conditioned on an observation of the

[2]Stoyanov et al. (2011) distinguished some of the $\mathcal{Y}$ variables as latent (i.e., unsupervised and ignored by the loss function). We omit this possibility, to simplify the notation.

form $\mathcal{X} = \mathbf{x}$, CRFs can include arbitrary overlapping features of the input without having to explicitly model input feature dependencies.

2.2 Inference in CRFs

Inference in general CRFs is intractable (Koller and Friedman, 2009). Nevertheless, there exist several approximate algorithms that have theoretical motivation and tend to exhibit good performance in practice. Those include variational methods such as loopy belief propagation (BP) (Murphy et al., 1999) and mean-field, as well as Markov Chain Monte Carlo methods.

ERMA training is applicable to any approximation that corresponds to a differentiable function, even if the function has no simple closed form but is computed by an iterative update algorithm. In this paper we select BP, which is exact when the factor graph is a tree, such as a linear-chain CRF, but whose results can be somewhat distorted by loops in the factor graph, as in our settings. BP computes beliefs about the marginal distribution of each random variable using iterative updates. We standardly approximate the *posterior* CRF marginals given the input observations by running BP over a CRF that enforces those observations.

2.3 Decoding

Conditional random fields are models of probability. A **decoder** is a procedure for converting these probabilities into system outputs. Given $\mathbf{x}$, the decoder would ideally choose $\mathbf{y}$ to minimize the loss $\ell(\mathbf{y}, \mathbf{y}^*)$, where ℓ compares a candidate assignment $\mathbf{y}$ to the true assignment $\mathbf{y}^*$. But of course we do not know the truth at test time. Instead we can average over *possible* values $\mathbf{y}'$ of the truth:

$$\operatorname*{argmin}_{\mathbf{y}} \sum_{\mathbf{y}'} p(\mathbf{y}' \mid \mathbf{x}) \cdot \ell(\mathbf{y}, \mathbf{y}') \tag{2}$$

This is the **minimum Bayes risk (MBR)** principle from statistical decision theory: choose $\mathbf{y}$ to minimize the *expected* loss (i.e., the risk) according to the CRF's posterior beliefs given $\mathbf{x}$.

In the NLP literature, CRFs are often decoded by choosing $\mathbf{y}$ to be the maximum posterior probability assignment (e.g., Sha and Pereira (2003), Sutton et al. (2007)). This is the MBR procedure for the 0-1 loss function that simply tests whether $\mathbf{y} = \mathbf{y}^*$. For other loss functions, however, the corresponding MBR procedure is preferable. For some loss functions it is tractable given the posterior marginals of p, while in other cases approximations are needed.

In our experiments we use MBR decoding (or a tractable approximation) but substitute the *approximate* posterior marginals of p as computed by BP. For example, if the loss of y is the number of incorrectly recovered output variables, MBR says to separately pick the most probable value for each output variable, according to its (approximate) marginal.

3 Minimum-Risk CRF Training

This section briefly describes the ERMA training algorithm from Stoyanov et al. (2011) and compares it to related structured learning methods. We assume a standard ML setting, with a set of training inputs $\mathbf{x}^i$ and corresponding correct outputs $\mathbf{y}^{i*}$. All the methods below are regularized in practice, but we omit mention of regularizers for simplicity.

3.1 Related Structured Learning Methods

When inference and decoding can be performed exactly, the CRF parameters $\vec{\theta}$ are often trained by maximum likelihood estimation (MLE):

$$\operatorname*{argmax}_{\theta} \sum_{i} \log p_\theta(\mathbf{y}^{i*} \mid \mathbf{x}^i) \tag{3}$$

The gradient of each summand $\log p_\theta(\mathbf{y}^{i*} \mid \mathbf{x}^i)$ can be computed by performing inference in two settings, one with $\mathbf{x}^i, \mathbf{y}^{i*}$ observed and one with only the conditioning events $\mathbf{x}^i$ observed. The gradient emerges as the difference between the feature expectations in the two cases. If exact inference is intractable, one can compute approximate feature expectations by loopy BP. Computing the approximate gradient in this way, and training the CRF with some gradient-based optimization method, has been shown to work relatively well in practice (Vishwanathan et al., 2006; Sutton and McCallum, 2005).

The above method takes into account neither the loss function that will be used for evaluation, nor the approximate algorithms that have been selected for inference and decoding at test time. Other structure learning methods do consider loss, though it is not obvious how to make them consider approximations. Those include maximum margin (Taskar et al., 2003; Finley and Joachims, 2008) and softmax-margin (Gimpel and Smith, 2010). The idea of margin-based methods is to choose weights $\vec{\theta}$ so that the correct alternative $\mathbf{y}^{i*}$ always gets a better score than each possible alternative $\mathbf{y}^i \in \mathcal{Y}$. The loss is incorporated in these methods by requiring the margin $(\vec{\theta} \cdot \vec{f}(\mathbf{x}^i, \mathbf{y}^{i*}) - \vec{\theta} \cdot \vec{f}(\mathbf{x}^i, \mathbf{y}^i)) \geq \ell(\mathbf{y}^i, \mathbf{y}^{i*})$, with penalized slack in these constraints. The softmax-margin method uses a different criterion—it resembles MLE but modifies the denominator of (1) to $Z_{\mathbf{x}} = \sum_{\mathbf{y}' \in \mathcal{Y}} \exp(\vec{\theta} \cdot \vec{f}(\mathbf{x}, \mathbf{y}') + \ell(\mathbf{y}', \mathbf{y}^*))$.

In our experiments we compare against MLE training (which is common) and softmax-margin, which incorporates loss and which Gimpel and Smith (2010) show is either better or competitive when compared to other margin methods on an NLP task. We adapt these methods to the loopy case in the obvious way, by replacing exact inference with loopy BP and keeping everything else the same.

3.2 Minimum-Risk Training

We wish to consider the approximate inference and decoding algorithms and the loss function that will be used during testing. Thus, we want θ to minimize the expected loss under the true data distribution P:

$$\underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_{\mathbf{xy} \sim P}[\ell(\delta_\theta(\mathbf{x}), \mathbf{y})] \tag{4}$$

where δ_θ is the decision rule (parameterized by θ), which decodes the results of inference under p_θ.

In practice, we do not know the true data distribution, but we can do **empirical risk minimization** (ERM), instead averaging the loss over our sample of $(\mathbf{x}^i, \mathbf{y}^i)$ pairs. ERM for structured prediction was first introduced in the speech community (Bahl et al., 1988) and later used in NLP (Och, 2003; Kakade et al., 2002; Suzuki et al., 2006; Li and Eisner, 2009, etc.). Previous applications of risk minimization assume *exact* inference, having defined the hypothesis space by a precomputed n-best list, lattice, or packed forest over which exact inference is possible.

The ERMA approach (Stoyanov et al., 2011) works with approximate inference and computes exact gradients of the output loss (or a differentiable surrogate) in the context of the approximate inference and decoding algorithms. To determine the gradient of $\ell(\delta_\theta(\mathbf{x}^i), \mathbf{y}^i)$ with respect to θ, the method relies on automatic differentiation in the reverse mode (Griewank and Corliss, 1991), a general technique for sensitivity analysis in computations. The intuition behind automatic differentiation is that the entire computation is a sequence of elementary differentiable operations. For each elementary operation, given that we know the input and result values, and the partial derivative of the loss with respect to the result, we can compute the partial derivative of the loss with respect to the inputs to the step. Differentiating the whole complicated computation can be carried out in backward pass in this step-by-step manner as long as we record intermediate results during the computation of the function (the forward pass). At the end, we accumulate the partials of the loss with respect to each parameter θ_i.

ERMA is similar to back-propagation used in recurrent neural networks, which involve cyclic updates like those in belief propagation (Williams and Zipser, 1989). It considers an "unrolled" version of the forward pass, in which "snapshots" of a variable at times t and $t+1$ are treated as distinct variables, with one perhaps influencing the other. The forward pass computes $\ell(\delta_\theta(\mathbf{x}^i), \mathbf{y}^i)$ by performing approximate inference, then decoding, then evaluation. These steps convert $(\mathbf{x}^i, \theta) \rightarrow$ marginals $\rightarrow$ decision $\rightarrow$ loss. The backward pass rewinds the entire computation, differentiating each phase in term. The total time required by this algorithm is roughly twice the time of the forward pass, so its complexity is comparable to approximate inference.

In this paper, we do not advocate any particular test-time inference or decoding procedures. It is reasonable to experiment with several choices that may produce faster or more accurate systems. We simply recommend doing ERMA training to match each selected test-time condition. Stoyanov et al. (2011) specifically showed how to train a system that will use *sum-product* BP for inference at test time (unlike margin-based methods). This may be advantageous for some tasks because it marginalizes over latent variables. However, it is popular and sometimes faster to do 1-best decoding, so we also include experiments where the test-time system returns a 1-best value of $\mathbf{y}$ (or an approximation to this if the CRF is loopy), based on *max-product* BP inference. Although 1-best systems are not differentiable functions, we can approach their behavior during ERM training by annealing the training objective (Smith and Eisner, 2006). In the annealed case we evaluate (4) and its gradient under sum-product BP, except that we perform inference under $p_{(\theta/T)}$ instead of p_θ. We gradually reduce the temperature $T \in \mathbb{R}$ from 1 to 0 as training proceeds, which turns sum-product inference into max-product by moving all the probability mass toward the highest-scoring assignment.

4 Modeling Natural Language with CRFs

This section describes three NLP problems that can be naturally modeled with approximate CRFs. The first problem, modeling congressional votes, has not been previously modeled with a CRF. We show that by switching to the principled CRF framework we can learn models that are much more accurate when evaluated on test data, though using the same (or less expressive) features as previous work. The other two problems, information extraction from semi-structured text and collective multi-label classification, have been modeled with loopy CRFs before. For all three models, we show that ERMA training results in better test set performance.[3]

4.1 Modeling Congressional Votes

The Congressional Vote (ConVote) corpus was created by Thomas et al. (2006) to study whether votes of U.S. congressional representatives can be predicted from the speeches they gave when debating a bill. The corpus consists of transcripts of congressional floor debates split into speech segments. Each speech segment is labeled with the representative who is speaking and the recorded vote of that representative on the bill. We aim to predict a high percentage of the recorded votes correctly.

Speakers often reference one another (e.g., “I thank the gentleman from Utah”), to indicate agreement or disagreement. The ConVote corpus manually annotates each phrase such as “the gentleman from Utah” with the representative that it denotes.

Thomas et al. (2006) show that classification using the agreement/disagreement information in the local context of such references, *together with* the rest of the language in the speeches, can lead to significant improvement over using either of these two sources of information in isolation. The original approach of Thomas et al. (2006) is based on training two Support Vector Machine (SVM) classifiers—one for classifying speeches as supporting/opposing the legislation and another for classifying references as agreement/disagreement. Both classifiers rely on bag-of-word (unigram) features of the document and the context surrounding the link respectively. The scores produced by the two SVMs are used to weight a global graph whose vertices are the representatives; then the min-cut algorithm is applied to partition the vertices into “yea” and “nay” voters.

While the approach of Thomas et al. (2006) leads to significant improvement over using the first SVM alone, it does not admit a probabilistic interpretation and the two classifiers are not trained jointly. We also remark that the min-cut technique would not generalize beyond binary random variables (yea/nay).

We observe that congressional votes together with references between speakers can be naturally modeled with a CRF. Figure 1 depicts the CRF constructed for one of the debates in the development part of the ConVote corpus. It contains a random variable for each representative’s vote. In addition, each speech is an observed input random variable: it is connected by a factor to its speaker’s vote and encourages it to be “yea” or “nay” according to features of the text of the speech. Finally, each reference in each speech is an observed input random variable connected by a factor to two votes—those of the speaker and the referent—which it encourages to agree or disagree according to features of the text surrounding the reference. Just as in (Thomas et al., 2006), the score of a global assignment to all votes is defined by considering both kinds of factors. However, unlike min-cut, CRF inference finds a probability distribution over assignments, not just a single best assignment. This fact allows us to train the two kinds of factors jointly (on the set of training debates where the votes are known) to predict the correct votes accurately (as defined by accuracy).

As Figure 1 shows, the reference factors introduce arbitrary loops, making exact inference intractable and thus motivating ERMA. Our experiments de-

[3]We also experimented with a fourth application, joint POS tagging and shallow parsing (Sutton et al., 2007) and we observed the same overall trend as the results in this paper (i.e., minimum risk training improved performance significantly). We do not include those experiments, however, because we were unable to make our baseline results replicate (Sutton et al., 2007).

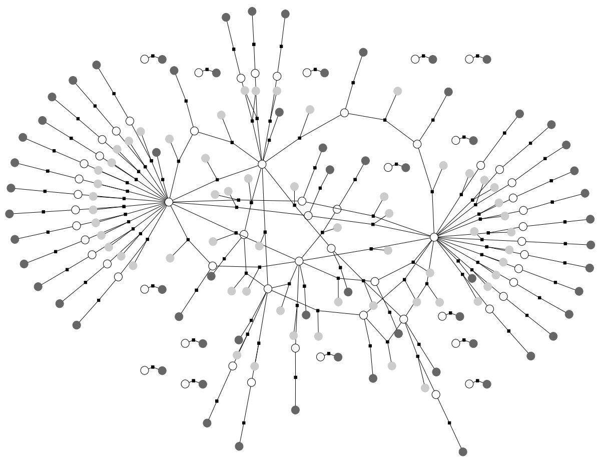

Figure 1: An example of a debate structure from the ConVote corpus. Each black square node represents a factor and is connected to the variables in that factor, shown as round nodes. Unshaded variables correspond to the representatives' votes and depict the output variables that we learn to jointly predict. Shaded variables correspond to the observed input data— the text of all speeches of a representative (in dark gray) or all local contexts of references between two representatives (in light gray).

scribed in section 5.2 show that switching to a CRF model (keeping the same features) leads to a sizable improvement over the previous state of the art—and that ERMA further significantly improves performance, particularly when it properly trains with the same inference algorithm (max-product vs. sum-product) to be used at test time.

Baseline. As an exact baseline, we compare against the results of Thomas et al. (2006). Their test-time Min-Cut algorithm is exact in this case: binary variables and a two-way classification.

4.2 Information Extraction from Semi-Structured Text

We utilize the CMU seminar announcement corpus of Freitag (2000) consisting of emails with seminar announcements. The task is to extract four fields that describe each seminar: *speaker, location, start time* and *end time*. The corpus annotates the document with all mentions of these four fields.

Sequential CRFs have been used successfully for semi-structured information extraction (Sutton and McCallum, 2005; Finkel et al., 2005). However, they cannot model non-local dependencies in the data. For example, in the seminar announcements corpus, if "Sutner" is mentioned once in an email in a context that identifies him as a speaker, it is likely that other occurrences of "Sutner" in the same email should be marked as *speaker*. Hence Finkel et al. (2005) and Sutton and McCallum (2005) propose adding non-local edges to a sequential CRF to represent soft consistency constraints. The model, called a "skip-chain CRF" and shown in Figure 2, contains a factor linking each pair of capitalized words with the same lexical form. The skip-chain CRF model exhibits better empirical performance than its sequential counterpart (Sutton and McCallum, 2005; Finkel et al., 2005).

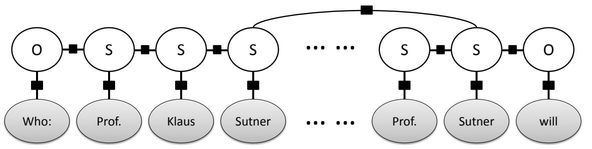

Figure 2: Skip-chain CRF for semi-structured information extraction.

The non-local skip links make exact inference intractable. To train the full model, Finkel et al. (2005) estimate the parameters of a sequential CRF and then manually select values for the weights of the non-local edges. At test time, they use Gibbs sampling to perform inference. Sutton and McCallum (2005) use max-product loopy belief propagation for test-time inference, and compare a training procedure that uses a piecewise approximation of the partition function against using sum-product loopy belief propagation to compute output variable marginals. They find that the two training regimens perform similarly on the overall task. All of these training procedures try to approximately maximize conditional likelihood, whereas we will aim to minimize the empirical loss of the approximate inference and decoding procedures.

Baseline. As an exact (non-loopy) baseline, we train a model without the skip chains. We give two baseline numbers in Table 1—for training the exact CRF with MLE and with ERM. The ERM setting resulted in a statistically significant improvement even in the exact case, thanks to the use of the loss function at training time.

4.3 Multi-Label Classification

Multi-label classification is the problem of assigning multiple labels to a document. For example, a news article can be about both "Libya" and "civil war." The most straightforward approach to multi-label classification employs a binary classifier for each class separately. However, previous work has shown that incorporating information about label dependencies can lead to improvement in performance (Elisseeff and Weston, 2001; Ghamrawi and McCallum, 2005; Finley and Joachims, 2008).

For this task we follow Ghamrawi and McCallum (2005) and Finley and Joachims (2008) and model the label interactions by constructing a fully connected CRF between the output labels. That is, for every document, we construct a CRF that contains a binary random variable for each label (indicating that the corresponding label is on/off for the document) and one binary edge for *every* unique pair of labels. This architecture can represent dependencies between labels, but leads to a setting in which the output variables form one massive clique. The resulting intractability of inference (and decoding) motivates the use of ERMA training.

Baseline. We train a model without any of the pairwise edges (i.e., a separate logistic regression model for each class). We report the single best baseline number, since MLE and ERM training resulted in statistically indistinguishable results.

5 Experiments

5.1 Learning Methodology

For all experiments we split the data into train/development/test sets using the standard splits when available. We tune optimization algorithm parameters (initial learning rate, batch size and meta-parameters λ and μ for stochastic meta descent) on the training set based on training objective convergence rates. We tune the regularization parameter β (below) on development data when available, otherwise we use a default value of 0.1—performance was generally robust for small changes in the value of β. All statistical significance testing is performed using paired permutation tests (Good, 2000).

Gradient-based Optimization. Gradient information from the back-propagation procedure can be used in a local optimization method to minimize empirical loss. In this paper we use stochastic meta descent (SMD) (Schraudolph, 1999). SMD is a second-order method that requires vector-Hessian products. For computing those, we do not need to maintain the full Hessian matrix. Instead, we apply more automatic differentiation magic—this time in the forward mode. Computing the vector-Hessian product and utilizing it in SMD does not add to the asymptotic runtime, it requires about twice as many arithmetic operations, and leads to much faster convergence of the learner in our experience. See Stoyanov et al. (2011) for details.

Since the empirical risk objective could overfit the training data, we add an L_2 regularizer $\beta \sum_j \theta_j^2$ that prefers parameter values close to 0. This improves generalization, like the margin constraints in margin-based methods.

Training Procedure Stoyanov et al. (2011) observed that the minimum-risk objective tends to be highly non-convex in practice. The usual approximate log likelihood training objective appeared to be smoother over the parameter space, but exhibited global maxima at parameter values that were relatively good, but sub-optimal for other loss functions. Mean-squared error (MSE) also gave a smoother objective than other loss functions. These observations motivated Stoyanov et al. (2011) to use a continuation method. They optimized approximate log-likelihood for a few iterations to get to a good part of the parameter space, then switched to using the hybrid loss function $\lambda\ell(y, y') + (1-\lambda)\ell_{\mathrm{MSE}}(y, y')$. The coefficient λ changed gradually from 0 to 1 during training, which morphs from optimizing a smoother loss to optimizing the desired bumpy test loss. We follow the same procedure.

Experiments in this paper use two evaluation metrics: percentage accuracy and F-measure. For both of these losses we decode by selecting the most probable value under the marginal distribution of each random variable. This is an exact MBR decode for accuracy but an approximate one for the F-measure; our ERMA training will try to compensate for this approximate decoder. This decoding procedure is not differentiable due to the use of the argmax function. To make the decoder differentiable, we replace argmax with a stochastic (soft-

	Problem	Congressional Vote		Semi-structured IE		Multi-label class.	
	Loss function	*Accuracy*		*Token-wise F-score*		*F-score*	
	Non-loopy Baseline	71.2		86.2 (87.1)		81.6	
	Loopy CRF models	INFERENCE:					
TRAINING:		maxprod	sumprod	maxprod	sumprod	maxprod	sumprod
	MLE	78.2	78.2	89.0	89.5	84.2	84.0
	Softmax-margin	79.0	79.0	90.1	90.2	84.3	83.8
	Min-risk (maxprod)	**85.1**	80.1	**90.9**	**90.7**	**84.5**	**84.4**
	Min-risk (sumprod)	83.6	**84.5**	90.3	**90.9**	**84.7**	**84.6**

Table 1: Results. The top of the table lists the loss function used for each problem and the score for the best exact baseline. The bottom lists results for the full models used with loopy BP. Models are tested with either sum-product BP (*sumprod*) or max-product BP (*maxprod*) and trained with MLE or the minimum risk criterion. Min-risk training runs are either annealed (*maxprod*), which matches max-product test, or not (*sumprod*), which matches sum-product test; grey cells in the table indicate matched training and test settings. In each column, we boldface the best result as well as all results that are not significantly worse (paired permutation test, $p < 0.05$).

max) version during training, averaging loss over all possible values v in proportion to their exponentiated probability $p(y_i = v \mid \mathbf{x})^{1/T_{\text{decode}}}$. This decoder loses smoothness and approaches an argmax decoder as T_{decode} decreases toward 0. For simplicity, our experiments just use a single fixed value of 0.1 for T_{decode}. Annealing the decoder slowly did not lead to significant differences in early experiments on development data.

5.2 Results

Table 1 lists results of our evaluation. For all three of our problems, using approximate CRFs results in statistically significant improvement over the exact baselines, for any of the training procedures. But among the training procedures for approximate CRFs, our ERMA procedure—minimizing empirical risk with the training setting matched to the test setting—improves over the two baselines, namely MLE and softmax-margin. MLE and softmax-margin training were statistically indistinguishable in our experiments with the exception of semi-structured IE. ERMA's improvements over them are statistically significant at the $p < .05$ level for the Congressional Vote and Semi-Structured IE problems and at the $p < .1$ level for the Multi-label classification problem (comparing each matched min-risk setting shown in a gray cell in Table 1 vs. MLE).

When minimizing risk, we also observe that matching training and test-time procedures can result in improved performance in one of the three problems, Congressional Vote. For this problem, the matched training condition performs better than the alternatives (accuracy of 85.1 vs. 83.6 for the annealed max-product testing and 84.5 vs 80.1 for the sum-product setting), significant at $p < .01$). We observe the same effect for semi-structured IE when testing using max-product inference. For the other remaining three problem setting training with either minimal risk training regiment.

Finally, we hypothesized that sum-product inference may produce more accurate results in certain cases as it allows more information about different parts of the model to be exchanged. However, our results show that for these three problems, sum-product and max-product inference yield statistically indistinguishable results. This may be because the particular CRFs we used included no latent variables (in constrast to the synthetic CRFs in Stoyanov et al. (2011)). As expected, we found that max-product BP converges in fewer iterations—sum-product BP required as many as twice the number of iterations for some of the runs.

Results in this paper represent a new state-of-the-art for the first two of the problems, Congressional Vote and Semi-structured IE. For Multi-Label classification, comparing against the SVM-based method of Finley and Joachims (2008) goes beyond the scope of the current paper.

6 Related Work

Minimum-risk training has been used in speech recognition (Bahl et al., 1988), machine translation (Och, 2003), and energy-based models generally (LeCun et al., 2006). In graphical models, methods have been proposed to directly minimize loss in tree-shaped or linear chain MRFs and CRFs (Kakade et al., 2002; Suzuki et al., 2006; Gross et al., 2007).

All of the above focus on exact inference. Our approach can be seen as generalizing these methods to arbitrary graph structures, arbitrary loss functions and approximate inference.

Lacoste-Julien et al. (2011) also consider the effects of approximate inference on loss. However, they assume the parameters are given, and modify the approximate inference algorithm for a given test instance to consider the loss function.

Using empirical risk minimization to *train* graphical models was independently proposed by Domke (2010; 2011). Just as in our own paper (Stoyanov et al., 2011), Domke took a decision-theoretic stance and proposed ERM as a way of calibrating the graphical model for use with approximate inference, or for use with data that do not quite match the modeling assumptions.[4]

In particular, (Domke, 2011) is similar to (Stoyanov et al., 2011) in using ERMA to train model parameters to be used with "truncated" inference that will be run for only a fixed number of iterations. For a common pixel-labeling benchmark in computer vision, Domke (2011) shows that this procedure improves training time by orders of magnitude, and slightly improves accuracy if the same number of message-passing iterations is used at test time.

Stoyanov and Eisner (2011) extend the ERMA objective function by adding an explicit runtime term. This allows them to tune model parameters and stopping criteria to learn models that obtain a given speed-accuracy tradeoff. Their approach improves this hybrid objective over a range of coefficients when compared to the traditional way of inducing sparse structures through L_1 regularization. Eisner and Daumé III (2011) propose the same linear combination of speed and accuracy as a reinforcement learning objective. In general, our proposed ERMA setting of trying to directly minimize the loss (or maximize the reward) of a controller is familiar in reinforcement learning, e.g., in model-free methods such as policy gradient.

We have been concerned with the fact that ERMA training objectives may suffer from local optima and non-differentiability. Stoyanov et al. (2011) studied several such settings, graphed the difficult objective, and identified some practical workarounds that are used in the present paper. Although these methods have enabled us to get strong results by reducing the empirical risk, we suspect that ERMA training objectives will benefit from more sophisticated optimization methods.

[4]However, he is less focused than we are on matching training conditions to test conditions (by including the decoder and task loss in the ERMA objective).

7 Conclusions

Motivated by the recently proposed method of Stoyanov et al. (2011) for minimum-risk training of CRF-based systems, we revisited three NLP domains that can naturally be modeled with approximate CRF-based systems. These include applications that have not been modeled with CRFs before (the ConVote corpus), as well as applications that have been modeled with loopy CRFs trained to minimize the approximate log-likelihood (semi-structured information extraction and collective multi-label classification). We show that (i) the NLP models are improved by moving to richer CRFs that require approximate inference, and (ii) empirical performance is always significantly improved by training to reduce the loss that would be achieved by approximate inference, even compared to another state-of-the-art training method (softmax-margin) that also considers loss and uses approximate inference. The general software package that implements the algorithms in this paper is available at `http://www.clsp.jhu.edu/~ves/software.html`.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant #0937060 to the Computing Research Association for the CIFellows Project.

References

L. Bahl, P. Brown, P. de Souza, and R. Mercer. 1988. A new algorithm for the estimation of hidden Markov model parameters. In *Proceedings of ICASSP*, pages 493–496.

E. Benson, A. Haghighi, and R. Barzilay. 2011. Event discovery in social media feeds. In *Proceedings of ACL-HLT*, pages 389–398.

Y. Choi, C. Cardie, E. Riloff, and S. Patwardhan. 2005. Identifying sources of opinions with conditional random fields and extraction patterns. In *Proceedings of HLT/EMNLP*, pages 355–362.

J. Domke. 2008. Learning convex inference of marginals. In *Proceedings of UAI*.

J. Domke. 2010. Implicit differentiation by perturbation. In *Advances in Neural Information Processing Systems*, pages 523–531.

J. Domke. 2011. Parameter learning with truncated message-passing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

J. Domke. 2012. Generic methods for optimization-based modeling. In *Proceedings of AISTATS*.

M. Dreyer and J. Eisner. 2009. Graphical models over multiple strings. In *Proceedings of EMNLP*, pages 101–110.

J. Eisner and Hal Daumé III. 2011. Learning speed-accuracy tradeoffs in nondeterministic inference algorithms. In *COST: NIPS 2011 Workshop on Computational Trade-offs in Statistical Learning*, Sierra Nevada, Spain, December.

A. Elisseeff and J. Weston. 2001. Kernel methods for multi-labelled classification and categorical regression problems. In *Advances in Neural Information Processing Systems*, pages 681–687.

J.R. Finkel, T. Grenager, and C. Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of ACL*, pages 363–370.

T. Finley and T. Joachims. 2008. Training structural SVMs when exact inference is intractable. In *Proceedings of ICML*, pages 304–311.

D. Freitag. 2000. Machine learning for information extraction in informal domains. *Machine learning*, 39(2).

N. Ghamrawi and A. McCallum. 2005. Collective multi-label classification. In *Proceedings of CIKM*, pages 195–200.

K. Gimpel and N.A. Smith. 2010. Softmax-margin CRFs: Training log-linear models with cost functions. In *Proceedings of ACL*, pages 733–736.

P. I. Good. 2000. *Permutation Tests*. Springer.

A. Griewank and G. Corliss, editors. 1991. *Automatic Differentiation of Algorithms*. SIAM, Philadelphia.

S. Gross, O. Russakovsky, C. Do, and S. Batzoglou. 2007. Training conditional random fields for maximum labelwise accuracy. *Advances in Neural Information Processing Systems*, 19:529.

H. Ji and R. Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of ACL-HLT*, pages 1148–1158.

S. Kakade, Y.W. Teh, and S. Roweis. 2002. An alternate objective function for Markovian fields. In *Proceedings of ICML*, pages 275–282.

D. Koller and N. Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press.

A. Kulesza and F. Pereira. 2008. Structured learning with approximate inference. In *Advances in Neural Information Processing Systems*, pages 785–792.

S. Lacoste-Julien, F. Huszr, and Z. Ghahramani. 2011. Approximate inference for the loss-calibrated Bayesian. In *Proceedings of AISTATS*.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289.

Y. LeCun, S. Chopra, R. Hadsell, M.A. Ranzato, and F.-J. Huang. 2006. A tutorial on energy-based learning. In G. Bakir, T. Hofman, B. Schlkopf, A. Smola, and B. Taskar, editors, *Predicting Structured Data*. MIT Press.

Z. Li and J. Eisner. 2009. First- and second-order expectation semirings with applications to minimum-risk training on translation forests. In *Proceedings of EMNLP*, pages 40–51.

K. P. Murphy, Y. Weiss, and M. I. Jordan. 1999. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of UAI*.

F. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL*, pages 160–167.

F. Peng and A. McCallum. 2006. Information extraction from research papers using conditional random fields. *Information Processing & Management*, 42(4):963–979.

N.N. Schraudolph. 1999. Local gain adaptation in stochastic gradient descent. In *Proceedings of ANN*, pages 569–574.

F. Sha and F. Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of ACL/HLT*, pages 134–141.

D.A. Smith and J. Eisner. 2006. Minimum risk annealing for training log-linear models. In *Proceedings of the COLING/ACL*, pages 787–794.

D. Smith and J. Eisner. 2008. Dependency parsing by belief propagation. In *Proceedings of EMNLP*, pages 145–156.

V. Stoyanov and J. Eisner. 2011. Learning cost-aware, loss-aware approximate inference policies for probabilistic graphical models. In *COST: NIPS 2011 Workshop on Computational Trade-offs in Statistical Learning*, Sierra Nevada, Spain, December.

V. Stoyanov, A. Ropson, and J. Eisner. 2011. Empirical risk minimization of graphical model parameters given approximate inference, decoding, and model structure. In *Proceedings of AISTATS*.

C. Sutton and A. McCallum. 2005. Piecewise training of undirected models. In *Proceedings of UAI*, pages 568–575.

C. Sutton, A. McCallum, and K. Rohanimanesh. 2007. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. *The Journal of Machine Learning Research*, 8:693–723.

J. Suzuki, E. McDermott, and H. Isozaki. 2006. Training conditional random fields with multivariate evaluation measures. In *Proceedings of COLING/ACL*, pages 217–224.

B. Taskar, C. Guestrin, and D. Koller. 2003. Max-margin Markov networks. *Proceedings of NIPS*, pages 25–32.

M. Thomas, B. Pang, and L. Lee. 2006. Get out the vote: Determining support or opposition from congressional floor-debate transcripts. In *Proceedings of EMNLP*, pages 327–335.

S. Vishwanathan, N. Schraudolph, M. Schmidt, and K. Murphy. 2006. Accelerated training of conditional random fields with stochastic gradient methods. In *Proceedings of ICML*, pages 969–976.

M. Wainwright. 2006. Estimating the "wrong" graphical model: Benefits in the computation-limited setting. *Journal of Machine Learning Research*, 7:1829–1859, September.

R.J. Williams and D. Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280.