# Semi-Supervised Conditional Random Fields for Improved Sequence Segmentation and Labeling

**Feng Jiao**
University of Waterloo

**Shaojun Wang  Chi-Hoon Lee**
**Russell Greiner  Dale Schuurmans**
University of Alberta

## Abstract

We present a new semi-supervised training procedure for conditional random fields (CRFs) that can be used to train sequence segmentors and labelers from a combination of labeled and unlabeled training data. Our approach is based on extending the minimum entropy regularization framework to the structured prediction case, yielding a training objective that combines unlabeled conditional entropy with labeled conditional likelihood. Although the training objective is no longer concave, it can still be used to improve an initial model (e.g. obtained from supervised training) by iterative ascent. We apply our new training algorithm to the problem of identifying gene and protein mentions in biological texts, and show that incorporating unlabeled data improves the performance of the supervised CRF in this case.

## 1 Introduction

Semi-supervised learning is often touted as one of the most natural forms of training for language processing tasks, since unlabeled data is so plentiful whereas labeled data is usually quite limited or expensive to obtain. The attractiveness of semi-supervised learning for language tasks is further heightened by the fact that the models learned are large and complex, and generally even thousands of labeled examples can only sparsely cover the parameter space. Moreover, in complex *structured* prediction tasks, such as parsing or sequence modeling (part-of-speech tagging, word segmentation, named entity recognition, and so on), it is considerably more difficult to obtain labeled training data than for classification tasks (such as document classification), since hand-labeling individual words and word boundaries is much harder than assigning text-level class labels.

Many approaches have been proposed for semi-supervised learning in the past, including: generative models (Castelli and Cover 1996; Cohen and Cozman 2006; Nigam et al. 2000), self-learning (Celeux and Govaert 1992; Yarowsky 1995), co-training (Blum and Mitchell 1998), information-theoretic regularization (Corduneanu and Jaakkola 2006; Grandvalet and Bengio 2004), and graph-based transductive methods (Zhou et al. 2004; Zhou et al. 2005; Zhu et al. 2003). Unfortunately, these techniques have been developed primarily for single class label classification problems, or class label classification with a structured input (Zhou et al. 2004; Zhou et al. 2005; Zhu et al. 2003). Although still highly desirable, semi-supervised learning for structured classification problems like sequence segmentation and labeling have not been as widely studied as in the other semi-supervised settings mentioned above, with the sole exception of generative models.

With generative models, it is natural to include unlabeled data using an expectation-maximization approach (Nigam et al. 2000). However, generative models generally do not achieve the same accuracy as discriminatively trained models, and therefore it is preferable to focus on discriminative approaches. Unfortunately, it is far from obvious how unlabeled training data can be naturally incorporated into a discriminative training criterion. For example, unlabeled data simply cancels from the objective if one attempts to use a traditional conditional likelihood criterion. Nevertheless, recent progress has been made on incorporating unlabeled data in discriminative training procedures. For example, dependencies can be introduced between the labels of nearby instances and thereby have an effect on training (Zhu et al. 2003; Li and McCallum 2005; Altun et al. 2005). These models are trained to encourage nearby data points to have the same class label, and they can obtain impressive accuracy using a very small amount of labeled data. However, since they model pairwise similarities among data points, most of these approaches require joint inference over the whole data set at test time, which is not practical for large data sets.

In this paper, we propose a new semi-supervised training method for conditional random fields (CRFs) that incorporates both labeled and unlabeled sequence data to estimate a discriminative

structured predictor. CRFs are a flexible and powerful model for structured predictors based on undirected graphical models that have been globally conditioned on a set of input covariates (Lafferty et al. 2001). CRFs have proved to be particularly useful for sequence segmentation and labeling tasks, since, as conditional models of the labels given inputs, they relax the independence assumptions made by traditional generative models like hidden Markov models. As such, CRFs provide additional flexibility for using arbitrary overlapping features of the input sequence to define a structured conditional model over the output sequence, while maintaining two advantages: first, efficient dynamic program can be used for inference in both classification and training, and second, the training objective is concave in the model parameters, which permits global optimization.

To obtain a new semi-supervised training algorithm for CRFs, we extend the minimum entropy regularization framework of Grandvalet and Bengio (2004) to structured predictors. The resulting objective combines the likelihood of the CRF on labeled training data with its conditional entropy on unlabeled training data. Unfortunately, the maximization objective is no longer concave, but we can still use it to effectively improve an initial supervised model. To develop an effective training procedure, we first show how the derivative of the new objective can be computed from the covariance matrix of the features on the unlabeled data (combined with the labeled conditional likelihood). This relationship facilitates the development of an efficient dynamic programming for computing the gradient, and thereby allows us to perform efficient iterative ascent for training. We apply our new training technique to the problem of sequence labeling and segmentation, and demonstrate it specifically on the problem of identifying gene and protein mentions in biological texts. Our results show the advantage of semi-supervised learning over the standard supervised algorithm.

## 2 Semi-supervised CRF training

In what follows, we use the same notation as (Lafferty et al. 2001). Let $\mathbf{X}$ be a random variable over data sequences to be labeled, and $\mathbf{Y}$ be a random variable over corresponding label sequences. All components, $\mathbf{Y_i}$, of $\mathbf{Y}$ are assumed to range over a finite label alphabet $\mathcal{Y}$. For example, $\mathbf{X}$ might range over sentences and $\mathbf{Y}$ over part-of-speech

taggings of those sentences; hence $\mathcal{Y}$ would be the set of possible part-of-speech tags in this case.

Assume we have a set of labeled examples, $\mathcal{D}^l = \left( (\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), \cdots, (\mathbf{x}^{(N)}, \mathbf{y}^{(N)}) \right)$, and unlabeled examples, $\mathcal{D}^u = \left( \mathbf{x}^{(N+1)}, \cdots, \mathbf{x}^{(M)} \right)$. We would like to build a CRF model

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_\theta(\mathbf{x})} \exp\left( \sum_{k=1}^{K} \theta_k f_k(\mathbf{x}, \mathbf{y}) \right)$$

$$= \frac{1}{Z_\theta(\mathbf{x})} \exp\left( \langle \theta, f(\mathbf{x}, \mathbf{y}) \rangle \right)$$

over sequential input and output data $\mathbf{x}, \mathbf{y}$, where $\theta = (\theta_1, \cdots, \theta_K)^\top$, $f(\mathbf{x}, \mathbf{y}) = (f_1(\mathbf{x}, \mathbf{y}), \cdots, f_K(\mathbf{x}, \mathbf{y}))^\top$ and

$$Z_\theta(\mathbf{x}) = \sum_{\mathbf{y}} \exp\left( \langle \theta, f(\mathbf{x}, \mathbf{y}) \rangle \right)$$

Our goal is to learn such a model from the combined set of labeled and unlabeled examples, $\mathcal{D}^l \cup \mathcal{D}^u$.

The standard supervised CRF training procedure is based upon maximizing the log conditional likelihood of the labeled examples in $\mathcal{D}^l$

$$CL(\theta) = \sum_{i=1}^{N} \log p_\theta(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}) - U(\theta) \quad (1)$$

where $U(\theta)$ is any standard regularizer on $\theta$, e.g. $U(\theta) = \|\theta\|^2/2$. Regularization can be used to limit over-fitting on rare features and avoid degeneracy in the case of correlated features. Obviously, (1) ignores the unlabeled examples in $\mathcal{D}^u$.

To make full use of the available training data, we propose a semi-supervised learning algorithm that exploits a form of *entropy regularization* on the unlabeled data. Specifically, for a semi-supervised CRF, we propose to maximize the following objective

$$RL(\theta) = \sum_{i=1}^{N} \log p_\theta(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}) - U(\theta) \quad (2)$$

$$+ \gamma \sum_{i=N+1}^{M} \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) \log p_\theta(\mathbf{y}|\mathbf{x}^{(i)})$$

where the first term is the penalized log conditional likelihood of the labeled data under the CRF, (1), and the second line is the negative conditional entropy of the CRF on the unlabeled data. Here, $\gamma$ is a tradeoff parameter that controls the influence of the unlabeled data.

This approach resembles that taken by (Grandvalet and Bengio 2004) for single variable classification, but here applied to structured CRF training. The motivation is that minimizing conditional entropy over unlabeled data encourages the algorithm to find putative labelings for the unlabeled data that are mutually reinforcing with the supervised labels; that is, greater certainty on the putative labelings coincides with greater conditional likelihood on the supervised labels, and vice versa. For a single classification variable this criterion has been shown to effectively partition unlabeled data into clusters (Grandvalet and Bengio 2004; Roberts et al. 2000).

To motivate the approach in more detail, consider the overlap between the probability distribution over a label sequence $y$ and the empirical distribution of $\tilde{p}(\mathbf{x})$ on the unlabeled data $\mathcal{D}^u$. The overlap can be measured by the Kullback-Leibler divergence $D(p_\theta(\mathbf{y}|\mathbf{x})\tilde{p}(\mathbf{x})\|\tilde{p}(\mathbf{x}))$. It is well known that Kullback-Leibler divergence (Cover and Thomas 1991) is positive and increases as the overlap between the two distributions decreases. In other words, maximizing Kullback-Leibler divergence implies that the overlap between two distributions is minimized. The total overlap over all possible label sequences can be defined as

$$\sum_{\mathbf{y}} D(p_\theta(\mathbf{y}|\mathbf{x})\tilde{p}(\mathbf{x})\|\tilde{p}(\mathbf{x}))$$

$$= \sum_{\mathbf{y}} \sum_{\mathbf{x}\in\mathcal{D}^u} p_\theta(\mathbf{y}|\mathbf{x})\tilde{p}(\mathbf{x}) \log \frac{p_\theta(\mathbf{y}|\mathbf{x})\tilde{p}(\mathbf{x})}{\tilde{p}(\mathbf{x})}$$

$$= \sum_{\mathbf{x}\in\mathcal{D}^u} \tilde{p}(\mathbf{x}) \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}) \log p_\theta(\mathbf{y}|\mathbf{x})$$

which motivates the negative entropy term in (2).

The combined training objective (2) exploits unlabeled data to improve the CRF model, as we will see. However, one drawback with this approach is that the entropy regularization term is not concave. To see why, note that the entropy regularizer can be seen as a composition, $h(\theta) = f(g(\theta))$, where $f : \Re^{|\mathcal{Y}|} \to \Re$, $f(g) = \sum_{\mathbf{y}} g_{\mathbf{y}} \log g_{\mathbf{y}}$ and $g_{\mathbf{y}} : \Re^K \to \Re$, $g_{\mathbf{y}}(\theta) = \frac{1}{Z_\theta(\mathbf{x})} \exp\left(\sum_{k=1}^K \theta_k f_k(\mathbf{x}, \mathbf{y})\right)$. For scalar $\theta$, the second derivative of a composition, $h = f \circ g$, is given by (Boyd and Vandenberghe 2004)

$$h''(\theta) = g'(\theta)^\top \nabla^2 f(g(\theta))g'(\theta) + \nabla f(g(\theta))^\top g''(\theta)$$

Although $f$ and $g_y$ are concave here, since $f$ is not nondecreasing, $h$ is not necessarily concave. So in general there are local maxima in (2).

## 3 An efficient training procedure

As (2) is not concave, many of the standard global maximization techniques do not apply. However, one can still use unlabeled data to improve a supervised CRF via iterative ascent. To derive an efficient iterative ascent procedure, we need to compute gradient of (2) with respect to the parameters $\theta$. Taking derivative of the objective function (2) with respect to $\theta$ yields Appendix A for the derivation)

$$\frac{\partial}{\partial\theta} RL(\theta) \tag{3}$$

$$= \sum_{i=1}^N \left( f(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) - \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) f(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) \right)$$

$$- \frac{\partial}{\partial\theta} U(\theta) + \gamma \sum_{i=N+1}^M \text{cov}_{p_\theta(\mathbf{y}|\mathbf{x}^{(i)})} \left[ f(\mathbf{x}^{(i)}, \mathbf{y}) \right] \theta$$

The first three items on the right hand side are just the standard gradient of the CRF objective, $\partial CL(\theta)/\partial\theta$ (Lafferty et al. 2001), and the final item is the gradient of the entropy regularizer (the derivation of which is given in Appendix A.

Here, $\text{cov}_{p_\theta(\mathbf{y}|\mathbf{x}^{(i)})}\left[f(\mathbf{x}^{(i)}, \mathbf{y})\right]$ is the conditional covariance matrix of the features, $f_j(\mathbf{x}, \mathbf{y})$, given sample sequence $\mathbf{x}^{(i)}$. In particular, the $(j, k)$th element of this matrix is given by

$$\text{cov}_{p_\theta(\mathbf{y}|\mathbf{x})}\left[f_j(\mathbf{x}, \mathbf{y})f_k(\mathbf{x}, \mathbf{y})\right]$$

$$= \text{E}_{p_\theta(\mathbf{y}|\mathbf{x})}\left(f_j(\mathbf{x}, \mathbf{y})f_k(\mathbf{x}, \mathbf{y})\right)$$

$$- \text{E}_{p_\theta(\mathbf{y}|\mathbf{x})}\left(f_j(\mathbf{x}, \mathbf{y})\right)\text{E}_{p_\theta(\mathbf{y}|\mathbf{x})}\left(f_k(\mathbf{x}, \mathbf{y})\right)$$

$$= \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x})\left(f_j(\mathbf{x}, \mathbf{y})f_k(\mathbf{x}, \mathbf{y})\right) \tag{4}$$

$$- \left(\sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x})f_j(\mathbf{x}, \mathbf{y})\right)\left(\sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x})f_k(\mathbf{x}, \mathbf{y})\right)$$

To efficiently calculate the gradient, we need to be able to efficiently compute the expectations with respect to $\mathbf{y}$ in (3) and (4). However, this can pose a challenge in general, because there are exponentially many values for $\mathbf{y}$. Techniques for computing the *linear* feature expectations in (3) are already well known if $\mathbf{y}$ is sufficiently structured (e.g. $\mathbf{y}$ forms a Markov chain) (Lafferty et al. 2001). However, we now have to develop efficient techniques for computing the *quadratic* feature expectations in (4).

For the quadratic feature expectations, first note that the diagonal terms, $j = l$, are straightforward, since each feature is an indicator, we have

that $f_j(\mathbf{x}, \mathbf{y})^2 = f_j(\mathbf{x}, \mathbf{y})$, and therefore the diagonal terms in the conditional covariance are just linear feature expectations $\mathrm{E}_{p_\theta(\mathbf{y}|\mathbf{x})}\left(f_j(\mathbf{x}, \mathbf{y})^2\right) = \mathrm{E}_{p_\theta(\mathbf{y}|\mathbf{x})}\left(f_j(\mathbf{x}, \mathbf{y})\right)$ as before.

For the off diagonal terms, $j \neq l$, however, we need to develop a new algorithm. Fortunately, for structured label sequences, $\mathbf{Y}$, one can devise an efficient algorithm for calculating the quadratic expectations based on nested dynamic programming. To illustrate the idea, we assume that the dependencies of $\mathbf{Y}$, conditioned on $\mathbf{X}$, form a *Markov chain*.

Define one feature for each state pair $(y', y)$, and one feature for each state-observation pair $(y, x)$, which we express with indicator functions $f_{y',y}(\langle u, v \rangle, \mathbf{y}|_{\langle u,v \rangle}, \mathbf{x}) = \delta(\mathbf{y}_u, y')\delta(\mathbf{y}_v, y)$ and $g_{y,x}(v, \mathbf{y}|_v, \mathbf{x}) = \delta(\mathbf{y}_v, y)\delta(\mathbf{x}_v, x)$ respectively. Following (Lafferty et al. 2001), we also add special start and stop states, $\mathbf{Y}_0 = \text{start}$ and $\mathbf{Y}_{n+1} = \text{stop}$. The conditional probability of a label sequence can now be expressed concisely in a matrix form. For each position $j$ in the observation sequence $\mathbf{x}$, define the $|\mathcal{Y}| \times |\mathcal{Y}|$ matrix random variable $M_j(\mathbf{x}) = [M_j(y', y|\mathbf{x})]$ by

$$M_j(y', y|\mathbf{x}) = \exp(\Lambda_j(y', y|\mathbf{x})) \quad \text{where}$$

$$\Lambda_j(y', y|\mathbf{x}) = \sum_k \lambda_k f_k\left(e_j, \mathbf{y}|_{e_j} = (y', y), \mathbf{x}\right)$$
$$+ \sum_k \mu_k g_k\left(v_j, \mathbf{y}|_{v_j} = y, \mathbf{x}\right)$$

Here $e_j$ is the edge with labels $(\mathbf{Y}_{j-1}, \mathbf{Y}_j)$ and $v_j$ is the vertex with label $\mathbf{Y}_j$.

For each index $j = 0, \cdots, n+1$ define the forward vectors $\alpha_j(\mathbf{x})$ with base case

$$\alpha_0(y|\mathbf{x}) = \begin{cases} 1 & \text{if } y = \text{start} \\ 0 & \text{otherwise} \end{cases}$$

and recurrence

$$\alpha_j(\mathbf{x}) = \alpha_{j-1}(\mathbf{x}) M_j(\mathbf{x})$$

Similarly, the backward vectors $\beta_j(\mathbf{x})$ are given by

$$\beta_{n+1}(y|\mathbf{x}) = \begin{cases} 1 & \text{if } y = \text{stop} \\ 0 & \text{otherwise} \end{cases}$$
$$\beta_j(\mathbf{x}) = M_{j+1}(\mathbf{x})\beta_{j+1}(\mathbf{x})$$

With these definitions, the expectation of the product of each pair of feature functions, $(f_j(\mathbf{x}, \mathbf{y}), f_k(\mathbf{x}, \mathbf{y}))$, $(f_j(\mathbf{x}, \mathbf{y}), g_k(\mathbf{x}, \mathbf{y}))$,

and $(g_j(\mathbf{x}, \mathbf{y}), g_k(\mathbf{x}, \mathbf{y}))$, for $j, k = 1, \cdots, K$, $j \neq k$, can be recursively calculated.

First define the summary matrix

$$M_{s+1,t-1}(y, y'|\mathbf{x}) = \left(\prod_{l=s+1}^{t-1} M_l(\mathbf{x})\right)_{y,y'}$$

Then the quadratic feature expectations can be computed by the following recursion, where the two double sums in each expectation correspond to the two cases depending on which feature occurs first ($e_s$ occuring before $e_t$).

$$\mathrm{E}_{p_\theta(\mathbf{y}|\mathbf{x})}\left(f_j(\mathbf{x}, \mathbf{y}) f_k(\mathbf{x}, \mathbf{y})\right)$$
$$= \sum_{\mathbf{x} \in \mathcal{D}^u} \sum_{s,t=1,s<t}^{n+1} \sum_{y',y} f_j\left(e_s, \mathbf{y}|_{e_s} = (y', y), \mathbf{x}\right)$$
$$\sum_{y'',y'''} f_k\left(e_t, \mathbf{y}|_{e_t} = (y'', y'''), \mathbf{x}\right)$$
$$\alpha_{s-1}(y'|\mathbf{x}) M_s(y', y|\mathbf{x}) M_{s+1,t-1}(y, y''|\mathbf{x})$$
$$M_t(y'', y'''|\mathbf{x})\beta_t(y'''|\mathbf{x})/Z_\theta(\mathbf{x})$$
$$+ \sum_{\mathbf{x} \in \mathcal{D}^u} \sum_{s,t=1,t<s}^{n+1} \sum_{y',y} f_j\left(e_t, \mathbf{y}|_{e_t} = (y', y), \mathbf{x}\right)$$
$$\sum_{y'',y'''} f_k\left(e_s, \mathbf{y}|_{e_s} = (y'', y'''), \mathbf{x}\right)$$
$$\alpha_{t-1}(y'''|\mathbf{x}) M_t(y''', y''|\mathbf{x}) M_{t+1,s-1}(y'', y'|\mathbf{x})$$
$$M_s(y', y|\mathbf{x})\beta_t(y|\mathbf{x})/Z_\theta(\mathbf{x})$$

$$\mathrm{E}_{p_\theta(\mathbf{y}|\mathbf{x})}\left(f_j(\mathbf{x}, \mathbf{y}) g_k(\mathbf{x}, \mathbf{y})\right)$$
$$= \sum_{\mathbf{x} \in \mathcal{D}^u} \sum_{s,t=1,s\leq t}^{n+1} \sum_{y',y} f_j\left(e_s, \mathbf{y}|_{e_s} = (y', y), \mathbf{x}\right)$$
$$\sum_{y''} g_k\left(v_t, \mathbf{y}|_{v_t} = y'', \mathbf{x}\right) \alpha_{s-1}(y'|\mathbf{x}) M_s(y', y|\mathbf{x})$$
$$M_{s+1,t-1}(y, y''|x)\beta_t(y''|\mathbf{x})/Z_\theta(\mathbf{x})$$
$$+ \sum_{x \in \mathcal{D}^u} \sum_{s,t=1,t<s}^{n+1} \sum_{y',y} f_j\left(e_t, \mathbf{y}|_{e_t} = (y', y), \mathbf{x}\right)$$
$$\sum_{y''} g_k\left(v_s, \mathbf{y}|_{v_s} = y'', \mathbf{x}\right) \alpha_{t-1}(y''|\mathbf{x})$$
$$M_{t+1,s-1}(y'', y'|x) M_s(y', y|\mathbf{x})\beta_t(y|\mathbf{x})/Z_\theta(\mathbf{x})$$

$$\mathrm{E}_{p_\theta(\mathbf{y}|\mathbf{x})}\left(g_j(\mathbf{x}, \mathbf{y}) g_k(\mathbf{x}, \mathbf{y})\right)$$
$$= \sum_{\mathbf{x} \in \mathcal{D}^u} \sum_{s,t=1,s<t}^{n+1}$$
$$\sum_{y'} g_j\left(v_s, \mathbf{y}|_{v_s} = y', \mathbf{x}\right) \sum_y g_k\left(v_t, \mathbf{y}|_{v_t} = y, \mathbf{x}\right)$$

$$\frac{\alpha_{s-1}(y'|\mathbf{x})M_{s+1,t-1}(y',y|\mathbf{x})\beta_t(y|\mathbf{x})}{Z_\theta(\mathbf{x})}$$

$$+ \sum_{\mathbf{x}\in\mathcal{D}^u}\sum_{s,t=1,t<s}^{n+1}$$

$$\sum_{y'}g_j\left(v_t,\mathbf{y}|_{v_t}=y',\mathbf{x}\right)\sum_{y'}g_k\left(v_s,\mathbf{y}|_{v_s}=y,\mathbf{x}\right)$$

$$\frac{\alpha_{t-1}(y|\mathbf{x})M_{t+1,s-1}(y,y'|\mathbf{x})\beta_t(y'|\mathbf{x})}{Z_\theta(\mathbf{x})}$$

The computation of these expectations can be organized in a trellis, as illustrated in Figure 1.

Once we obtain the gradient of the objective function (2), we use limited-memory L-BFGS, a quasi-Newton optimization algorithm (McCallum 2002; Nocedal and Wright 2000), to find the local maxima with the initial value being set to be the optimal solution of the supervised CRF on labeled data.

## 4 Time and space complexity

The time and space complexity of the semi-supervised CRF training procedure is greater than that of standard supervised CRF training, but nevertheless remains a small degree polynomial in the size of the training data. Let

| | | |
|---|---|---|
| $m_l$ | = | size of the labeled set |
| $m_u$ | = | size of the unlabeled set |
| $n_l$ | = | labeled sequence length |
| $n_u$ | = | unlabeled sequence length |
| $n_t$ | = | test sequence length |
| $s$ | = | number of states |
| $c$ | = | number of training iterations. |

Then the time required to classify a test sequence is $O(n_t s^2)$, independent of training method, since the Viterbi decoder needs to access each path.

For training, supervised CRF training requires $O(cm_l n_l s^2)$ time, whereas semi-supervised CRF training requires $O(cm_l n_l s^2 + cm_u n_u^2 s^3)$ time. The additional cost for semi-supervised training arises from the extra nested loop required to calculated the quadratic feature expectations, which introduces in an additional $n_u s$ factor.

However, the space requirements of the two training methods are the same. That is, even though the covariance matrix has size $O(K^2)$, there is never any need to store the entire matrix in memory. Rather, since we only need to compute the product of the covariance with $\theta$, the calculation can be performed iteratively without using extra space beyond that already required by supervised CRF training.
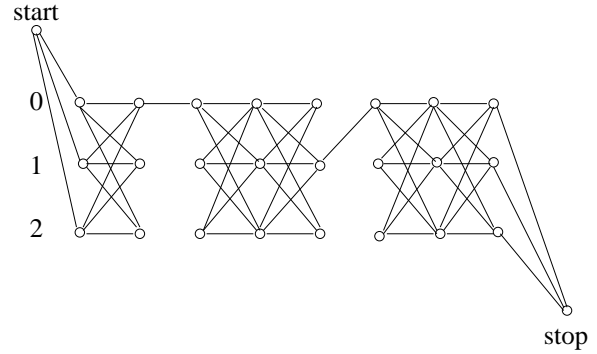


Figure 1: Trellis for computing the expectation of a feature product over a pair of feature functions, $f_{00}$ vs $f_{10}$, where the feature $f_{00}$ occurs first. This leads to one double sum.

## 5 Identifying gene and protein mentions

We have developed our new semi-supervised training procedure to address the problem of information extraction from biomedical text, which has received significant attention in the past few years. We have specifically focused on the problem of identifying explicit mentions of gene and protein names (McDonald and Pereira 2005). Recently, McDonald and Pereira (2005) have obtained interesting results on this problem by using a standard supervised CRF approach. However, our contention is that stronger results could be obtained in this domain by exploiting a large corpus of unannotated biomedical text to improve the quality of the predictions, which we now show.

Given a biomedical text, the task of identifying gene mentions can be interpreted as a tagging task, where each word in the text can be labeled with a tag that indicates whether it is the beginning of gene mention (B), the continuation of a gene mention (I), or outside of any gene mention (O). To compare the performance of different taggers learned by different mechanisms, one can measure the precision, recall and F-measure, given by

$$precision = \frac{\#\ correct\ predictions}{\#\ predicted\ gene\ mentions}$$

$$recall = \frac{\#\ correct\ predictions}{\#\ true\ gene\ mentions}$$

$$F\text{-}measure = \frac{2\times precision\times recall}{precision+recall}$$

In our evaluation, we compared the proposed semi-supervised learning approach to the state of the art supervised CRF of McDonald and Pereira (2005), and also to self-training (Celeux and Govaert 1992; Yarowsky 1995), using the same feature set as (McDonald and Pereira 2005). The CRF training procedures, supervised and semi-

supervised, were run with the same regularization function, $U(\theta) = \|\theta\|^2/2$, used in (McDonald and Pereira 2005).

First we evaluated the performance of the semi-supervised CRF in detail, by varying the ratio between the amount of labeled and unlabeled data, and also varying the tradeoff parameter $\gamma$. We choose a labeled training set $A$ consisting of 5448 words, and considered alternative unlabeled training sets, $B$ (5210 words), $C$ (10,208 words), and $D$ (25,145 words), consisting of the same, 2 times and 5 times as many sentences as $A$ respectively. All of these sets were disjoint and selected randomly from the full corpus, the smaller one in (McDonald et al. 2005), consisting of 184,903 words in total. To determine sensitivity to the parameter $\gamma$ we examined a range of discrete values $0, 0.1, 0.5, 1, 5, 10, 20, 50$.

In our first experiment, we train the CRF models using labeled set $A$ and unlabeled sets $B$, $C$ and $D$ respectively. Then test the performance on the sets $B$, $C$ and $D$ respectively The results of our evaluation are shown in Table 1. The performance of the supervised CRF algorithm, trained only on the labeled set $A$, is given on the first row in Table 1 (corresponding to $\gamma = 0$). By comparison, the results obtained by the semi-supervised CRFs on the held-out sets $B$, $C$ and $D$ are given in Table 1 by increasing the value of $\gamma$.

The results of this experiment demonstrate quite clearly that in most cases the semi-supervised CRF obtains higher precision, recall and F-measure than the fully supervised CRF, yielding a 20% improvement in the best case.

In our second experiment, again we train the CRF models using labeled set $A$ and unlabeled sets $B$, $C$ and $D$ respectively with increasing values of $\gamma$, but we test the performance on the held-out set $E$ which is the full corpus minus the labeled set $A$ and unlabeled sets $B$, $C$ and $D$. The results of our evaluation are shown in Table 2 and Figure 2. The blue line in Figure 2 is the result of the supervised CRF algorithm, trained only on the labeled set $A$. In particular, by using the supervised CRF model, the system predicted 3334 out of 7472 gene mentions, of which 2435 were correct, resulting in a precision of 0.73, recall of 0.33 and F-measure of 0.45. The other curves are those of the semi-supervised CRFs.

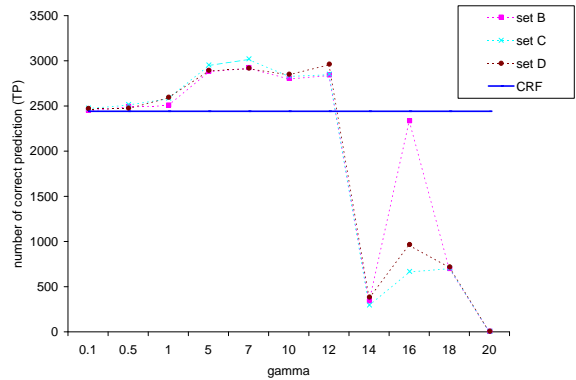The results of this experiment demonstrate quite clearly that the semi-supervised CRFs simultane-



Figure 2: Performance of the supervised and semi-supervised CRFs. The sets $B$, $C$ and $D$ refer to the unlabeled training set used by the semi-supervised algorithm.

ously increase both the number of predicted gene mentions and the number of correct predictions, thus the precision remains almost the same as the supervised CRF, and the recall increases significantly.

Both experiments as illustrated in Figure 2 and Tables 1 and 2 show that clearly better results are obtained by incorporating additional unlabeled training data, even when evaluating on disjoint testing data (Figure 2). The performance of the semi-supervised CRF is not overly sensitive to the tradeoff parameter $\gamma$, except that $\gamma$ cannot be set too large.

## 5.1 Comparison to self-training

For completeness, we also compared our results to the self-learning algorithm, which has commonly been referred to as bootstrapping in natural language processing and originally popularized by the work of Yarowsky in word sense disambiguation (Abney 2004; Yarowsky 1995). In fact, similar ideas have been developed in pattern recognition under the name of the decision-directed algorithm (Duda and Hart 1973), and also traced back to 1970s in the EM literature (Celeux and Govaert 1992). The basic algorithm works as follows:

1. Given $\mathcal{D}^l$ and $\mathcal{D}^u$, begin with a seed set of labeled examples, $\mathcal{D}^{(0)}$, chosen from $\mathcal{D}^l$.

2. For $m = 0, 1, \cdots$

    (a) Train the supervised CRF on labeled examples $\mathcal{D}^{(m)}$, obtaining $\theta^{(m)}$.

    (b) For each sequence $\mathbf{x}^{(i)} \in \mathcal{D}^u$, find $\mathbf{y}^{(i)}_{(m)} = \arg\max_{\mathbf{y}} p_{\theta^{(m)}}(\mathbf{y}|\mathbf{x}^{(i)})$ via Viterbi decoding or other inference algorithm, and add the pair $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}_{(m)})$ to the set of labeled examples (replacing any previous label for $\mathbf{x}^{(i)}$ if present).

Table 1: Performance of the semi-supervised CRFs obtained on the held-out sets $B$, $C$ and $D$

| $\gamma$ | Test Set B, Trained on A and B | | | Test Set C, Trained on A and C | | | Test Set D, Trained on A and D | | |
|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | F-Measure | Precision | Recall | F-Measure | Precision | Recall | F-Measure |
| 0 | 0.80 | 0.36 | 0.50 | 0.77 | 0.29 | 0.43 | 0.74 | 0.30 | 0.43 |
| 0.1 | 0.82 | 0.4 | 0.54 | 0.79 | 0.32 | 0.46 | 0.74 | 0.31 | 0.44 |
| 0.5 | 0.82 | 0.4 | 0.54 | 0.79 | 0.33 | 0.46 | 0.74 | 0.31 | 0.44 |
| 1 | 0.82 | 0.4 | 0.54 | 0.77 | 0.34 | 0.47 | 0.73 | 0.33 | 0.45 |
| 5 | 0.84 | 0.45 | 0.59 | 0.78 | 0.38 | 0.51 | 0.72 | 0.36 | 0.48 |
| 10 | 0.78 | 0.46 | 0.58 | 0.66 | 0.38 | 0.48 | 0.66 | 0.38 | 0.47 |

Table 2: Performance of the semi-supervised CRFs trained by using unlabeled sets $B$, $C$ and $D$

| $\gamma$ | Test Set E, Trained on A and B | | Test Set E, Trained on A and C | | Test Set E, Trained on A and D | |
|---|---|---|---|---|---|---|
| | # predicted | # correct prediction | # predicted | # correct prediction | # predicted | # correct prediction |
| 0.1 | 3345 | 2446 | 3376 | 2470 | 3366 | 2466 |
| 0.5 | 3413 | 2489 | 3450 | 2510 | 3376 | 2469 |
| 1 | 3446 | 2503 | 3588 | 2580 | 3607 | 2590 |
| 5 | 4089 | 2878 | 4206 | 2947 | 4165 | 2888 |
| 10 | 4450 | 2799 | 4762 | 2827 | 4778 | 2845 |

(c) If for each $\mathbf{x}^{(i)} \in \mathcal{D}^u$, $\mathbf{y}_{(m)}^{(i)} = \mathbf{y}_{(m-1)}^{(i)}$, stop; otherwise $m = m + 1$, iterate.

We implemented this self training approach and tried it in our experiments. Unfortunately, we were not able to obtain any improvements over the standard supervised CRF with self-learning, using the sets $\mathcal{D}^l = A$, and $\mathcal{D}^u \in \{B, C, D\}$. The semi-supervised CRF remains the best of the approaches we have tried on this problem.

## 6 Conclusions and further directions

We have presented a new semi-supervised training algorithm for CRFs, based on extending minimum conditional entropy regularization to the structured prediction case. Our approach is motivated by the information-theoretic argument (Grandvalet and Bengio 2004; Roberts et al. 2000) that unlabeled examples can provide the most benefit when classes have small overlap. An iterative ascent optimization procedure was developed for this new criterion, which exploits a nested dynamic programming approach to efficiently compute the covariance matrix of the features.

We applied our new approach to the problem of identifying gene name occurrences in biological text, exploiting the availability of auxiliary unlabeled data to improve the performance of the state of the art supervised CRF approach in this domain. Our semi-supervised CRF approach shares all of the benefits of the standard CRF training, including the ability to exploit arbitrary features of the inputs, while obtaining improved accuracy through the use of unlabeled data. The main drawback is that training time is increased because of the extra nested loop needed to calculate feature covariances. Nevertheless, the algorithm is sufficiently efficient to be trained on unlabeled data sets that yield a notable improvement in classification accuracy over standard supervised training. To further accelerate the training process of our semi-supervised CRFs, we may apply stochastic gradient optimization method with adaptive gain adjustment as proposed by Vishwanathan et al. (2006).

## References

S. Abney. (2004). Understanding the Yarowsky algorithm. *Computational Linguistics*, 30(3):365-395.

Y. Altun, D. McAllester and M. Belkin. (2005). Maximum margin semi-supervised learning for structured variables. *Advances in Neural Information Processing Systems 18*.

A. Blum and T. Mitchell. (1998). Combining labeled and unlabeled data with co-training. *Proceedings of the Workshop on Computational Learning Theory*, 92-100.

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

V. Castelli and T. Cover. (1996). The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameter. *IEEE Trans. on Information Theory*, 42(6):2102-2117.

G. Celeux and G. Govaert. (1992). A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics and Data Analysis*, 14:315-332.

I. Cohen and F. Cozman. (2006). Risks of semi-supervised learning. *Semi-Supervised Learning*, O. Chapelle, B. Scholköpf and A. Zien, (Editors), 55-70, MIT Press.

A. Corduneanu and T. Jaakkola. (2006). Data dependent regularization. *Semi-Supervised Learning*, O. Chapelle, B. Scholköpf and A. Zien, (Editors), 163-182, MIT Press.

T. Cover and J. Thomas, (1991). *Elements of Information Theory*, John Wiley & Sons.

R. Duda and P. Hart. (1973). *Pattern Classification and Scene Analysis*, John Wiley & Sons.

Y. Grandvalet and Y. Bengio. (2004). Semi-supervised learning by entropy minimization, *Advances in Neural Information Processing Systems*, 17:529-536.

J. Lafferty, A. McCallum and F. Pereira. (2001). Conditional random fields: probabilistic models for segmenting and labeling sequence data. *Proceedings of the 18th International Conference on Machine Learning*, 282-289.

W. Li and A. McCallum. (2005). Semi-supervised sequence modeling with syntactic topic models. *Proceedings of Twentieth National Conference on Artificial Intelligence*, 813-818.

A. McCallum. (2002). MALLET: A machine learning for language toolkit. [http://mallet.cs.umass.edu]

R. McDonald, K. Lerman and Y. Jin. (2005). Conditional random field biomedical entity tagger. [http://www.seas.upenn.edu/~sryantm/software/BioTagger/]

R. McDonald and F. Pereira. (2005). Identifying gene and protein mentions in text using conditional random fields. *BMC Bioinformatics 2005*, 6(Suppl 1):S6.

K. Nigam, A. McCallum, S. Thrun and T. Mitchell. (2000). Text classification from labeled and unlabeled documents using EM. *Machine learning.* 39(2/3):135-167.

J. Nocedal and S. Wright. (2000). *Numerical Optimization*, Springer.

S. Roberts, R. Everson and I. Rezek. (2000). Maximum certainty data partitioning. *Pattern Recognition*, 33(5):833-839.

S. Vishwanathan, N. Schraudolph, M. Schmidt and K. Murphy. (2006). Accelerated training of conditional random fields with stochastic meta-descent. *Proceedings of the 23th International Conference on Machine Learning.*

D. Yarowsky. (1995). Unsupervised word sense disambiguation rivaling supervised methods. *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, 189-196.

D. Zhou, O. Bousquet, T. Navin Lal, J. Weston and B. Schölkopf. (2004). Learning with local and global consistency. *Advances in Neural Information Processing Systems*, 16:321-328.

D. Zhou, J. Huang and B. Schölkopf. (2005). Learning from labeled and unlabeled data on a directed graph. *Proceedings of the 22nd International Conference on Machine Learning*, 1041-1048.

X. Zhu, Z. Ghahramani and J. Lafferty. (2003). Semi-supervised learning using Gaussian fields and harmonic functions. *Proceedings of the 20th International Conference on Machine Learning*, 912-919.

## A  Deriving the gradient of the entropy

We wish to show that

$$\frac{\partial}{\partial \theta} \left( \sum_{i=N+1}^{M} \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) \log p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) \right)$$
$$= \sum_{i=N+1}^{M} \mathrm{cov}_{p_\theta(\mathbf{y}|\mathbf{x}^{(i)})} \left[ f(\mathbf{x}^{(i)}, \mathbf{y}) \right] \theta \quad (5)$$

First, note that some simple calculation yields

$$\frac{\partial \log Z_\theta(\mathbf{x}^{(i)})}{\partial \theta_j} = \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) f_j(\mathbf{x}^{(i)}, \mathbf{y})$$

and

$$\frac{\partial p_\theta(\mathbf{y}|\mathbf{x}^{(i)})}{\partial \theta_j} = \frac{\partial}{\partial \theta_j} \left( \frac{\exp\left( \langle \theta, f(\mathbf{x}^{(i)}, \mathbf{y}) \rangle \right)}{Z_\theta(\mathbf{x}^{(i)})} \right)$$
$$= p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) f_j(\mathbf{x}^{(i)}, \mathbf{y})$$
$$- p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) f_j(\mathbf{x}^{(i)}, \mathbf{y})$$

Therefore

$$\frac{\partial}{\partial \theta_j} \left( \sum_{i=N+1}^{M} \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) \log p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) \right)$$
$$= \sum_{i=N+1}^{M} \frac{\partial}{\partial \theta_j} \left( \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) \langle \theta, f(\mathbf{x}^{(i)}, \mathbf{y}) \rangle \right.$$
$$\left. - \log Z_\theta(\mathbf{x}^{(i)}) \right)$$
$$= \sum_{i=N+1}^{M} \left( \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) f_j(\mathbf{x}^{(i)}, \mathbf{y}) \right.$$
$$+ \sum_{\mathbf{y}} \frac{\partial p_\theta(\mathbf{y}|\mathbf{x}^{(i)})}{\partial \theta_j} \langle \theta, f(\mathbf{x}^{(i)}, \mathbf{y}) \rangle$$
$$\left. - \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) f_j(\mathbf{x}^{(i)}, \mathbf{y}) \right)$$
$$= \sum_{i=N+1}^{M} \left( \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) f_j(\mathbf{x}^{(i)}, \mathbf{y}) \langle \theta, f(\mathbf{x}^{(i)}, \mathbf{y}) \rangle \right.$$
$$- [\sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) \langle \theta, f(\mathbf{x}^{(i)}, \mathbf{y}) \rangle]$$
$$\left. [\sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) f_j(\mathbf{x}^{(i)}, \mathbf{y})] \right)$$
$$= \sum_{i=N+1}^{M} \left( \sum_{k} \theta_k \left[ \sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) f_j(\mathbf{x}^{(i)}, \mathbf{y}) f_k(\mathbf{x}^{(i)}, \mathbf{y}) \right. \right.$$
$$- [\sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) f_k(\mathbf{x}^{(i)}, \mathbf{y})]$$
$$\left. \left. [\sum_{\mathbf{y}} p_\theta(\mathbf{y}|\mathbf{x}^{(i)}) f_j(\mathbf{x}^{(i)}, \mathbf{y})] \right] \right)$$

In the vector form, this can be written as (5)