Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data

John Lafferty[†*] LAFFERTY@CS.CMU.EDU
Andrew McCallum[*†] MCCALLUM@WHIZBANG.COM
Fernando Pereira[*‡] FPEREIRA@WHIZBANG.COM
[*]WhizBang! Labs–Research, 4616 Henry Street, Pittsburgh, PA 15213 USA
[†]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213 USA
[‡]Department of Computer and Information Science, University of Pennsylvania, Philadelpha, PA 19104 USA

Abstract

We present *conditional random fields*, a framework for building probabilistic models to segment and label sequence data. Conditional random fields offer several advantages over hidden Markov models and stochastic grammars for such tasks, including the ability to relax strong independence assumptions made in those models. Conditional random fields also avoid a fundamental limitation of maximum entropy Markov models (MEMMs) and other discriminative Markov models based on directed graphical models, which can be biased towards states with few successor states. We present iterative parameter estimation algorithms for conditional random fields and compare the performance of the resulting models to HMMs and MEMMs on synthetic data.

1. Introduction

The need to segment and label sequences arises in many different problems in several scientific fields. Hidden Markov models (HMMs) and stochastic grammars are well understood and widely used probabilistic models for such problems. In computational biology, HMMs and stochastic grammars have been successfully used to align biological sequences, find sequences homologous to a known evolutionary family, and analyze RNA secondary structure (Durbin et al., 1998). In computational linguistics and computer science, HMMs and stochastic grammars have been applied to a wide variety of problems in text and speech processing, including topic segmentation, part-of-speech tagging, information extraction, and syntactic disambiguation (Manning & Schütze, 1999).

HMMs and stochastic grammars are generative models, assigning a joint probability to paired observation and label sequences; the parameters are typically trained to maximize the joint likelihood of training examples. To define a joint probability over observation and label sequences, a generative model needs to enumerate all possible observation sequences, typically requiring a representation in which observations are task-appropriate atomic entities, such as words or nucleotides. In particular, it is not practical to represent multiple interacting features of the observations or long-range correlations in the observations, as that would require the generative model to carry out intractable inferences over a general graphical model.

This difficulty is one of the main motivations for looking at conditional models as an alternative. A conditional model specifies the probabilities of possible label sequences given an observation sequence. Therefore, it does not expend modeling effort on the observations, which at test time are fixed anyway. Furthermore, the conditional probability of the label sequence can depend on arbitrary, non-independent features of the observation sequence without forcing the model to account for those feature dependencies. The chosen features may represent attributes at different levels of granularity (eg. words, characters) of the same observations, or aggregate properties of the observation sequence, and the number of observations may be enumerable. The probability of a transition between labels may depend not only on the current observation, but also on past and future observations, if available. In contrast, generative models must make very strict independence assumptions on the observations, for instance conditional independence given the labels, to achieve tractability.

Maximum entropy Markov models (MEMMs) are conditional probabilistic sequence models that attain all of the above advantages (McCallum et al., 2000). In MEMMs, each source state[1] has an exponential model that takes the vector observation features as input, and outputs a distribution over possible next states. These exponential mod-

[1]Output labels are associated with states; it is possible for several states to have the same label, but for simplicity in the rest of this paper we assume a one-to-one correspondence.

els are trained by an appropriate iterative scaling method in the maximum entropy framework. Previously published experimental results show MEMMs increasing recall and doubling precision relative to HMMs in a FAQ segmentation task.

MEMMs and other non-generative finite-state models based on next-state classifiers, such as discriminative Markov models (Bottou, 1991), share a weakness we call here the *label bias problem*: the transitions leaving a given state compete only against each other, rather than against all other transitions in the model. In probabilistic terms, transition scores are the conditional probabilities of possible next states given the current state and the observation sequence. This per-state normalization of transition scores implies a "conservation of score mass" (Bottou, 1991) whereby all the mass that arrives at a state must be distributed among the possible successor states. An observation can affect which destination states get the mass, but not how much total mass to pass on. This causes a bias toward states with fewer outgoing transitions. In the extreme case, a state with a single outgoing transition effectively ignores the observation. In those cases, unlike in HMMs, Viterbi decoding cannot downgrade a branch based on observations after the branch point, and models with state-transition structures that have sparsely-connected chains of states are not properly handled.

This paper introduces *conditional random fields* (CRFs), a sequence modeling framework that has all the advantages of MEMMs but also solves label bias problem in a principled way. The essential difference between CRFs and MEMMs is that the underlying graphical model structure of CRFs is undirected, while that of MEMMs is directed. A MEMM uses per-state exponential models for the conditional probabilities of next states given the current state, while a CRF has a single exponential model for the joint probability of a label sequence given the observation sequence. Since normalization is done globally rather than for each state individually, the weights of different features at different states can be traded off against each other.

We can also think of a CRF as a finite state model with unnormalized transition probabilities. However, unlike some other weighted finite-state approaches (LeCun et al., 1998), CRFs assign a well-defined probability distribution over possible labelings, trained by maximum likelihood or MAP estimation. Furthermore, the loss function is convex,[2] guaranteeing convergence to the global optimum. CRFs also generalize easily to analogues of stochastic context-free grammars that would be useful in such problems as RNA secondary structure prediction and natural language processing.

We present the model, describe two training procedures and sketch a proof of convergence. We also give experimental results on synthetic data showing that CRFs solve the classical version of the label bias problem, and, more significantly, that CRFs perform better than HMMs and MEMMs when the true data distribution has higher-order dependencies than the model, as is often the case in practice.

[2]In the case of fully-observable states we are discussing here; if several states have the same label, the usual local maxima of Baum-Welch arise.

2. The Label Bias Problem

Classical probabilistic automata (Paz, 1971), discriminative Markov models (Bottou, 1991), maximum entropy taggers (Ratnaparkhi, 1996), and MEMMs, as well as non-probabilistic sequence tagging and segmentation models with independently-trained next-state classifiers (Punyakanok & Roth, 2001) are all potential victims of the label bias problem. In each case, the observation-conditioned scores for transitions represent only the relative importance of the transitions *leaving a single state*, and do not reflect an estimate of the relative importance of observation-label pairings *across different states*. In other words the "score mass flow" in decoding is conserved: the total score mass, representing the strength of a partial labeling hypothesis, that arrives in a state must be distributed among its outgoing transitions. An observation can affect the relative distribution of this mass among outgoing transitions of a state, but the state must pass on exactly the same total mass it received. When a state has a restricted number of outgoing transitions, this can result in unintuitive and undesirable outcomes.

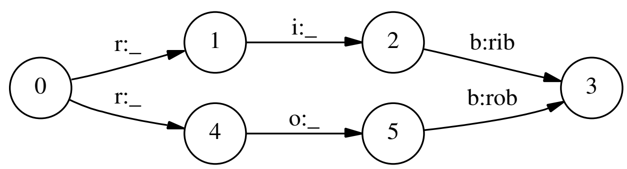

Figure 1. Label bias example, after (Bottou, 1991). For conciseness, we place observation-label pairs $o : l$ on transitions rather than states; the symbol '_' represents the null output label.

For example, Figure 1 represents a simple finite-state model designed to distinguish between the two words rib and rob. Suppose that the observation sequence is r i b. At the first time step, r matches both transitions from the start state, so the probability mass gets distributed roughly equally among those two transitions. Next we observe i. Both states 1 and 4 have only one outgoing transition. State 1 has seen this observation often in training; state 4 has almost never seen this observation; but like state 1, state 4 has no choice but to pass all its mass to its single outgoing transition, since it is not generating the observation, only conditioning on it. Thus, states with a single outgoing transition effectively ignore their observations. More generally, states with low-entropy next state distributions will take little notice of observations. Returning to the example, the top path and the bottom path should be about equally likely, independent of the observation sequence. If one of the two words is slightly more common in the training set, the transitions out of the start state will slightly prefer its corresponding transition, and that word's state sequence will always win. This behavior is demonstrated experimentally in Section 5.

The label bias problem was described in Léon Bottou's thesis (1991, page 221). Two solutions are proposed there. One is to change the state-transition structure of the model. In the above example we could collapse states 1 and 4, and delay the branching until we get a discriminating observation. This operation is a special case of determinization (Mohri, 1997), but determinization of weighted finite-state machines is not always possible, and even when possible, it may lead to combinatorial explosion. The other solution mentioned is to start with a fully-connected model and let the training procedure figure out a good structure. But that would preclude the use of prior structural knowledge that has proven so valuable in information extraction tasks (Freitag & McCallum, 2000).

Proper solutions require models that account for whole state sequences at once by letting some transitions "vote" more strongly than others. This implies that score mass will not be conserved, but instead individual transitions can "amplify" or "dampen" the mass they receive. In the above example, the transitions out of the start state would have low-magnitude amplification, and transitions out of states 1 and 4 would have high-magnitude amplification, and a proportionally higher contribution to the selection of the Viterbi path.[3]

In the related work section we discuss other heuristic model classes that account for state sequences globally rather than locally. To the best of our knowledge, CRFs are the only model class that does this in a purely probabilistic setting, with guaranteed global maximum likelihood convergence.

3. Conditional Random Fields

Suppose that $\mathbf{X}$ is data to be annotated with labels $\mathbf{Y}$. For example, $\mathbf{X}$ might be a natural language sentence and $\mathbf{Y}$ a labeling of its words with part-of-speech tags. The random variables $\mathbf{X}$ and $\mathbf{Y}$ are jointly distributed, but in a discriminative framework we construct a probabilistic model $p(\mathbf{Y} \mid \mathbf{X})$ to be estimated from N paired training instances $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N}$, and do not explictly model the marginal $p(\mathbf{X})$.

Definition. *Let $G = (V, E)$ be a graph such that $\mathbf{Y} = (\mathbf{Y}_v)_{v \in V}$, so that $\mathbf{Y}$ is indexed by the vertices of G. Then $(\mathbf{X}, \mathbf{Y})$ is a conditional random field in case, when conditioned on $\mathbf{X}$, the random variables $\mathbf{Y}_v$ obey the Markov property with respect to the graph: $p(\mathbf{Y}_v \mid \mathbf{X}, \mathbf{Y}_w, w \neq v) = p(\mathbf{Y}_v \mid \mathbf{X}, \mathbf{Y}_w, w \sim v)$, where $w \sim v$ means that w and v are neighbors in G.*

Thus, a CRF is a random field globally conditioned on the observation $\mathbf{X}$. Throughout the paper we tacitly assume that the graph G is fixed. The simplest and most important example for modeling sequences is when G is a simple chain graph or line: $G = (V = \{1, 2, \ldots m\}, E = \{(i, i+1)\})$. $\mathbf{X}$ may also have a natural graph structure; yet in general it is not necessary to assume that $\mathbf{X}$ and $\mathbf{Y}$ have the same graphical structure, or even that $\mathbf{X}$ has any graphical structure at all. However, in this paper we will be most concerned with sequences $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n)$ and $\mathbf{Y} = (\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_n)$.

If the graph $G = (V, E)$ of $\mathbf{Y}$ is a tree (of which a chain is the simplest example), its cliques are the edges and vertices. Therefore, by the fundamental theorem of random fields (Hammersley & Clifford, 1971), the joint distribution over the label sequence $\mathbf{Y}$ given $\mathbf{X}$ has the form:

$$p_\theta(\mathbf{y} \mid \mathbf{x}) \propto \qquad (1)$$
$$\exp\left(\sum_{e \in E, k} \lambda_k\, f_k(e, \mathbf{y}|_e, \mathbf{x}) + \sum_{v \in V, k} \mu_k\, g_k(v, \mathbf{y}|_v, \mathbf{x})\right)$$

where $\mathbf{y}|_S$ is the set of components of $\mathbf{y}$ associated with the vertices in subgraph S.

We assume that the *features* f_k and g_k are given and fixed. For example, a Boolean vertex feature g_k might be true of the word $\mathbf{X}_i$ is upper case and the tag $\mathbf{Y}_i$ is a proper name.

The *parameter estimation problem* is to estimate $\theta = (\lambda_1, \lambda_2, \ldots; \mu_1, \mu_2, \ldots)$. In Section 4 we describe an iterative scaling algorithm that maximizes the log-likelihood objective function $\mathcal{O}(\theta)$:

$$\begin{aligned} \mathcal{O}(\theta) &= \sum_{i=1}^{N} \log p_\theta(\mathbf{y}^{(i)} \mid \mathbf{x}^{(i)}) \\ &\propto \sum_{\mathbf{x},\mathbf{y}} \widetilde{p}(\mathbf{x}, \mathbf{y}) \, \log p_\theta(\mathbf{y} \mid \mathbf{x}) \end{aligned}$$

where $\{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{N}$ is training data with empirical distribution $\widetilde{p}(\mathbf{x}, \mathbf{y})$.

To recover the usual parameterization of an HMM, we would take edge features parameterized by label-label pairs (y', y), and vertex features parameterized by label-observation pairs (l, x), defining

$$\begin{aligned} f_{y',y}\,(<u,v>, \mathbf{y}|_{<u,v>}, \mathbf{x}) &= \delta(\mathbf{y}_u, y')\, \delta(\mathbf{y}_v, y) \\ g_{y,x}\,(v, \mathbf{y}|_v, \mathbf{x}) &= \delta(\mathbf{y}_v, y)\, \delta(\mathbf{x}_v, x) \end{aligned}$$

Thus, the parameters are the numbers $\lambda_{y',y}$ and $\mu_{y,x}$, and our parameter estimation algorithm would provide a discriminatively trained HMM (Saul & Jordan, 1996; MacKay, 1996).

[3] Weighted determinization and minimization techniques shift transitions weights while preserving overall path weight (Mohri, 2000); their connection to this discussion deserves further study.

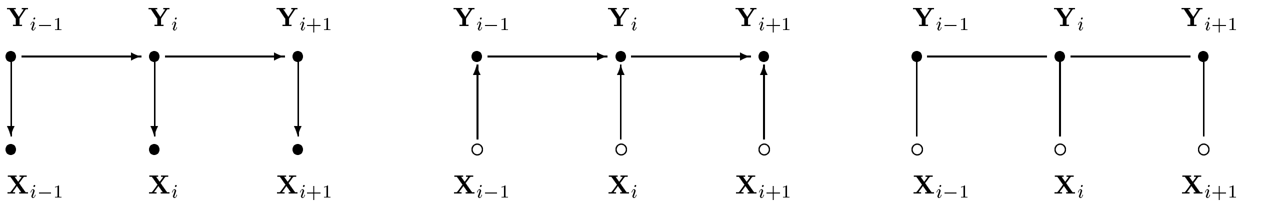

Figure 2. Graphical structures of simple HMMs (left), MEMMs (center), and the chain-graph case of CRFs (right) for sequences. An open circle indicates that the variable is not generated by the model.

Although it can be parameterized similarly to an HMM, the class of conditional random field models has many potential advantages over traditional HMMs. As a discriminative model, a CRF does not make strong independence assumptions about the random variables $\mathbf{X}_v$. The features f and g can depend on arbitrary parts of the observation sequence $\mathbf{x}$, and need not be "local." For efficiency, we need only require the labels $\mathbf{Y}_v$ have local interactions. Since the features do not need to specify the entire state or observation, $\mathbf{X}_v$, one can expect that the model can be estimated from less training data. Another attractive property is the convexity of the likelihood objective function; indeed, CRFs share all of the convexity properties of general maximum entropy models.

For the remainder of the paper we assume that the dependencies of $\mathbf{Y}$, conditioned on $\mathbf{X}$, form a chain. To simplify some expressions, we add special start and stop states $\mathbf{Y}_0 = \mathsf{start}$ and $\mathbf{Y}_{n+1} = \mathsf{stop}$. Thus, we will be using the graphical structure shown in Figure 2 . For a chain graph, the conditional probability of a label sequence can be expressed concisely in matrix form, which will be useful in describing the parameter estimation and inference algorithms in Section 4. Suppose that $\mathbf{Y}_i$ takes values from the alphabet of labels, $\mathcal{Y}$, and that $p_\theta(\mathbf{Y} \mid \mathbf{X})$ is a CRF given by (1). Then for each observation index i, we define a $|\mathcal{Y}| \times |\mathcal{Y}|$ matrix random variable $M_i(\mathbf{x}) = [M_i(y', y \mid \mathbf{x})]$ given by

$$\begin{aligned} M_i(y', y \mid \mathbf{x}) &= \exp\left(\Lambda_i(y', y \mid \mathbf{x})\right) \\ \Lambda_i(y', y \mid \mathbf{x}) &= \textstyle\sum_k \lambda_k\, f_k(e_i, \mathbf{Y}|_{e_i} = (y', y), \mathbf{x}) + \\ &\quad \textstyle\sum_k \mu_k\, g_k(v_i, \mathbf{Y}|_{v_i} = y, \mathbf{x}) \end{aligned}$$

where e_i is the edge with labels $(\mathbf{Y}_{i-1}, \mathbf{Y}_i)$ and v_i is the vertex with label $\mathbf{Y}_i$. Then the normalization (partition function) $Z_\theta(\mathbf{x})$ is the $(\mathsf{start}, \mathsf{stop})$ entry of the product of these matrices:

$$Z_\theta(\mathbf{x}) = \left(M_1(\mathbf{x})\, M_2(\mathbf{x}) \cdots M_{n+1}(\mathbf{x})\right)_{\mathsf{start},\mathsf{stop}}$$

Using this notation, the conditional probability of a sequence $\mathbf{y}$ is written as

$$p_\theta(\mathbf{y} \mid \mathbf{x}) = \frac{\prod_{i=1}^{n+1} M_i(\mathbf{y}_{i-1}, \mathbf{y}_i \mid \mathbf{x})}{\left(\prod_{i=1}^{n+1} M_i(\mathbf{x})\right)_{\mathsf{start},\mathsf{stop}}}$$

where $\mathbf{y}_0 = \mathsf{start}$ and $\mathbf{y}_{n+1} = \mathsf{stop}$.

4. Parameter Estimation for CRFs

We now describe two iterative scaling algorithms to compute values of λ_k and μ_k that maximize the log-likelihood of the training data. Both algorithms are based on the improved iterative scaling (IIS) algorithm of Della Pietra et al. (1997); the proof technique based on auxiliary functions can be extended to show convergence of the algorithms for CRFs.

Iterative scaling algorithms update the weights as $\lambda_k \leftarrow \lambda_k + \delta\lambda_k$ and $\mu_k \leftarrow \mu_k + \delta\mu_k$ for appropriately chosen $\delta\lambda_k$ and $\delta\mu_k$. Application of the IIS algorithm to our problem yields the following update equations. For edge features, $\delta\lambda_k$ is chosen to satisfy the equation

$$\begin{aligned} \widetilde{E}[f_k] &\stackrel{\text{def}}{=} \sum_{\mathbf{x},\mathbf{y}} \widetilde{p}(\mathbf{x}, \mathbf{y}) \sum_{i=1}^{n+1} f_k(e_i, \mathbf{y}|_{e_i}, \mathbf{x}) \\ &= \sum_{\mathbf{x},\mathbf{y}} \widetilde{p}(\mathbf{x})\, p(\mathbf{y} \mid \mathbf{x}) \sum_{i=1}^{n+1} f_k(e_i, \mathbf{y}|_{e_i}, \mathbf{x})\, e^{\delta\lambda_k T(\mathbf{x},\mathbf{y})} \end{aligned}$$

and for vertex features, $\delta\mu_k$ is chosen to satisfy

$$\begin{aligned} \widetilde{E}[g_k] &\stackrel{\text{def}}{=} \sum_{\mathbf{x},\mathbf{y}} \widetilde{p}(\mathbf{x}, \mathbf{y}) \sum_{i=1}^{n} g_k(e_i, \mathbf{y}|_{e_i}, \mathbf{x}) = \\ &\sum_{\mathbf{x},\mathbf{y}} \widetilde{p}(\mathbf{x})\, p(\mathbf{y} \mid \mathbf{x}) \sum_{i=1}^{n} g_k(v_i, \mathbf{y}|_{v_i}, \mathbf{x})\, e^{\delta\mu_k T(\mathbf{x},\mathbf{y})} \end{aligned}$$

where we define $T(\mathbf{x}, \mathbf{y})$ to be the *total feature count*

$$T(\mathbf{x}, \mathbf{y}) \stackrel{\text{def}}{=} \sum_{i,k} f_k(e_i, \mathbf{y}|_{e_i}, \mathbf{x}) + \sum_{i,k} g_k(v_i, \mathbf{y}|_{v_i}, \mathbf{x})$$

However, efficiently computing the exponential sum on the right-hand side of these equations is problematic, because $T(\mathbf{x}, \mathbf{y})$ is a global property of $(\mathbf{x}, \mathbf{y})$, and dynamic programming will sum over sequences with potentially varying T. To deal with this, the first algorithm, Algorithm S, uses a "slack feature." The second, Algorithm T, keeps track of partial T totals.

For Algorithm S, we define the *slack feature* by:

$$\begin{aligned} s(\mathbf{x}, \mathbf{y}) &\stackrel{\text{def}}{=} \\ &S - \sum_i \sum_k f_k(e_i, \mathbf{y}|_{e_i}, \mathbf{x}) - \sum_i \sum_k g_k(v_i, \mathbf{y}|_{v_i}, \mathbf{x}) \end{aligned}$$

where S is a constant chosen so that $s(\mathbf{x}^{(i)}, \mathbf{y}) \geq 0$ for all $\mathbf{y}$ and all observation vectors $\mathbf{x}^{(i)}$ in the training set. Feature s is "global," that is, it does not correspond to any particular edge or vertex.

For each index $i = 0, \ldots, n+1$ we now define the *forward vector* $\alpha_i(\mathbf{x})$ by

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

and, in our matrix notation,

$$\alpha_i(\mathbf{x}) \;=\; \alpha_{i-1}(\mathbf{x})\, M_i(\mathbf{x})$$

That is,

$$\alpha_i(y \,|\, \mathbf{x}) \;=\; \sum_{y'} \alpha_{i-1}(y' \,|\, \mathbf{x})\, M_i(y', y \,|\, \mathbf{x})$$

The *backward vector* $\beta_i(\mathbf{x})$ is defined by

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

and

$$\beta_i(\mathbf{x})^\top \;=\; M_{i+1}(\mathbf{x})\, \beta_{i+1}(\mathbf{x})$$

or

$$\beta_i(y \,|\, \mathbf{x}) \;=\; \sum_{y'} M_{i+1}(y, y' \,|\, \mathbf{x})\, \beta_{i+1}(y' \,|\, \mathbf{x})$$

With these definitions, we can write our update equations:

$$\delta\lambda_k \;=\; \frac{1}{S} \log \frac{\widetilde{E} f_k}{E f_k}\,, \qquad \delta\mu_k \;=\; \frac{1}{S} \log \frac{\widetilde{E} g_k}{E g_k}$$

where

$$E f_k \;=\; \sum_{\mathbf{x}} \widetilde{p}(\mathbf{x}) \sum_{i=1}^{n+1} \sum_{y', y} f_k(e_i, \mathbf{y}|_{e_i} = (y', y), \mathbf{x}) \;\times\; \frac{\alpha_{i-1}(y' \,|\, \mathbf{x})\, M_i(y', y \,|\, \mathbf{x})\, \beta_i(y \,|\, \mathbf{x})}{Z_\theta(\mathbf{x})}$$

$$E g_k \;=\; \sum_{\mathbf{x}} \widetilde{p}(\mathbf{x}) \sum_{i=1}^{n} \sum_{y} g_k(v_i, \mathbf{y}|_{v_i} = y, \mathbf{x}) \;\times\; \frac{\alpha_i(y \,|\, \mathbf{x})\, \beta_i(y \,|\, \mathbf{x})}{Z_\theta(\mathbf{x})}$$

The factors involving the forward and backward vectors in the above equations have the same meaning as for standard hidden Markov models. For example,

$$p_\theta(\mathbf{Y}_i = y \,|\, \mathbf{x}) \;=\; \frac{\alpha_i(y \,|\, \mathbf{x})\, \beta_i(y \,|\, \mathbf{x})}{Z_\theta(\mathbf{x})}$$

is the marginal probability of label $\mathbf{Y}_i = y$ given that the observation sequence is $\mathbf{x}$. This algorithm is closely related to the algorithm of Darroch and Ratcliff (1972), and MART algorithms used in image reconstruction.

The constant S in Algorithm S is can be quite large, since in practice it is proportional to the length of the longest training observation sequence. As a result, the algorithm may converge slowly, taking very small steps toward the maximum in each iteration. If the length of the observations $\mathbf{x}^{(i)}$ and the number of active features varies greatly, a faster-converging algorithm can be obtained by keeping track of feature totals for each observation sequence separately,

Let $T(\mathbf{x}) \stackrel{\text{def}}{=} \max_{\mathbf{y}} T(\mathbf{x}, \mathbf{y})$. Algorithm T accumulates feature expectations into counters indexed by $T(\mathbf{x})$. Using the forward-backward calculations given in the previous section, we compute

$$a_{k,t} \;=\; \sum_{\mathbf{x}, T(\mathbf{x}) = t} \widetilde{p}(\mathbf{x}) \sum_{i=1}^{n+1} \sum_{y', y} f_k(e_i, \mathbf{y}|_{e_i} = (y', y), \mathbf{x}) \;\times\; \frac{\alpha_{i-1}(y' \,|\, \mathbf{x})\, M_i(y', y \,|\, \mathbf{x})\, \beta_i(y \,|\, \mathbf{x})}{Z_\theta(\mathbf{x})}$$

$$b_{k,t} \;=\; \sum_{\mathbf{x}, T(\mathbf{x}) = t} \widetilde{p}(\mathbf{x}) \sum_{i=1}^{n} \sum_{y} g_k(v_i, \mathbf{y}|_{v_i} = y, \mathbf{x}) \;\times\; \frac{\alpha_i(y \,|\, \mathbf{x})\, \beta_i(y \,|\, \mathbf{x})}{Z_\theta(\mathbf{x})}$$

With these definitions, we choose as our parameter updates $\delta\lambda_k = \log \beta_k$ and $\delta\mu_k = \log \gamma_k$, where β_k and γ_k are the unique positive roots to the following polynomial equations

$$\sum_{t=0}^{T_{\max}} a_{k,t}\, \beta_k^t \;=\; \widetilde{E} f_k\,, \qquad \sum_{t=0}^{T_{\max}} b_{k,t}\, \gamma_k^t \;=\; \widetilde{E} g_k \tag{2}$$

These roots can be easily computed using Newton's method.

A single iteration of Algorithm S and Algorithm T has roughly the same time and space complexity of the well known Baum-Welch algorithm for HMMs. To prove convergence of our algorithms, we can derive an auxiliary function to bound the change in likelihood from below; this method is developed in detail by Della Pietra et al. (1997). The full proof is somewhat detailed; however, here we give an idea of how to derive the auxiliary function. To simplify notation, we assume only edge features f_k with parameters λ_k.

Given two parameter settings $\theta = (\lambda_1, \lambda_2, \ldots)$ and $\theta' = (\lambda_1 + \delta\lambda_1, \lambda_2 + \delta\lambda_2, \ldots)$, we bound from below the change in the objective function with an *auxiliary function* $\mathcal{A}(\theta', \theta)$ as follows

$$\mathcal{O}(\theta') - \mathcal{O}(\theta) \;=\; \sum_{\mathbf{x}, \mathbf{y}} \widetilde{p}(\mathbf{x}, \mathbf{y}) \log \frac{p_{\theta'}(\mathbf{y} \,|\, \mathbf{x})}{p_\theta(\mathbf{y} \,|\, \mathbf{x})}$$

$$
\begin{aligned}
&= (\theta' - \theta) \cdot E_{\widetilde{p}} f - \sum_{\mathbf{x}} \widetilde{p}(\mathbf{x}) \log \frac{Z_{\theta'}(\mathbf{x})}{Z_{\theta}(\mathbf{x})} \\
&\geq (\theta' - \theta) \cdot E_{\widetilde{p}} f - \sum_{\mathbf{x}} \widetilde{p}(\mathbf{x}) \frac{Z_{\theta'}(\mathbf{x})}{Z_{\theta}(\mathbf{x})} \\
&= \delta\lambda \cdot E_{\widetilde{p}} f - \sum_{\mathbf{x}} \widetilde{p}(\mathbf{x}) \sum_{\mathbf{y}} p_{\theta}(\mathbf{y} \mid \mathbf{x})\, e^{\delta\lambda \cdot f(\mathbf{x},\mathbf{y})} \\
&\geq \delta\lambda \cdot E_{\widetilde{p}} f - \sum_{\mathbf{x},\mathbf{y},k} \widetilde{p}(\mathbf{x})\, p_{\theta}(\mathbf{y} \mid \mathbf{x})\, \frac{f_k(\mathbf{x},\mathbf{y})}{T(\mathbf{x})} e^{\delta\lambda_k T(\mathbf{x})} \\
&\stackrel{\text{def}}{=} \mathcal{A}(\theta', \theta)
\end{aligned}
$$

where the inequalities follow from the convexity of $\log$ and $\exp$. Differentiating $\mathcal{A}$ with respect to $\delta\lambda_k$ and setting the result to zero yields equation (2).

5. Experiments on Synthetic Data

We now discuss two sets of experiments with synthetic data that highlight the differences between CRFs and MEMMs. The first experiments are a direct verification of the label bias problem discussed in Section 2. In the second set of experiments, we generate synthetic data using randomly chosen hidden Markov models, each of which is a mixture of a first-order and second-order model. Competing *first-order* models are then trained and compared on test data. As the data becomes more second-order, the test error rates of the trained models increase. This experiment corresponds to the common modeling practice of approximating complex local and long-range dependencies, as occur in natural data, by small-order Markov models. Our results clearly indicate that even when the models are parameterized in exactly the same way, CRFs are more robust to inaccurate modeling assumptions than MEMMs, and resolve the label bias problem, which affects the performance of MEMMs. Our experiments are *not* designed to demonstrate the advantages of CRFs over HMMs, including the ability to incorporate overlapping features of the local and global context given in the input; those advantages were clearly demonstrated for MEMMs in earlier work (McCallum et al., 2000).

5.1 Modeling label bias

We generate data from a simple HMM which encodes a noisy version of the finite-state network in Figure 1. Each state emits its designated symbol with probability $29/32$ and any of the other symbols with probability $1/32$. We train both an MEMM and a CRF with the same topologies on the data generated by the HMM. The observation features are simply the identity of the observation symbols. In a typical run using $2,000$ training and 500 test samples, trained to convergence of the iterative scaling algorithm, the CRF error is 4.6% while the MEMM error is 42%, showing that the MEMM fails to discriminate between the two branches.

5.2 Modeling mixed-order sources

The results we report on below used five labels, a-e ($|\mathcal{Y}| = 5$), and 26 observation values, A-Z ($|\mathcal{X}| = 26$); however, the results were qualitatively the same over a range of sizes for $\mathcal{Y}$ and $\mathcal{X}$. We generate data from a mixed-order HMM with state transition probabilities is given by

$$
\begin{aligned}
p_\alpha(\mathbf{y}_i \mid \mathbf{y}_{i-1}, \mathbf{y}_{i-2}) &= \\
&\alpha\, p_2(\mathbf{y}_i \mid \mathbf{y}_{i-1}, \mathbf{y}_{i-2}) + (1-\alpha)\, p_1(\mathbf{y}_i \mid \mathbf{y}_{i-1})
\end{aligned}
$$

and emission probabilities given by

$$
\begin{aligned}
p_\alpha(\mathbf{x}_i \mid \mathbf{y}_i, \mathbf{x}_{i-1}) &= \\
&\alpha\, p_2(\mathbf{x}_i \mid \mathbf{y}_i, \mathbf{x}_{i-1}) + (1-\alpha)\, p_1(\mathbf{x}_i \mid \mathbf{y}_i)
\end{aligned}
$$

Thus, for $\alpha = 0$ we have a standard first-order HMM. In order to limit the size of the (Bayes) error rate for the resulting models, the transition matrices p_α are constrained to be sparse. In particular, $p_\alpha(\cdot \mid y, y')$ can have at most two nonzero entries, for each y, y', and $p_\alpha(\cdot \mid y, x')$ can have at most three nonzero entries for each y, x'. For each randomly generated model, a sample of 1000 sequences of length 25 is generated for training and testing.

On each randomly generated training set, a CRF is trained using Algorithm S. (Note that since the length of the sequences and number of active features is constant, and Algorithms S and T are identical.) The algorithm is fairly slow to converge, typically taking approximately 500 iterations for the model to stabilize. On the 500 MHz Pentium PC used in our experiments, each iteration takes approximately 0.2 seconds. On the same data an MEMM is trained using iterative scaling, which does not require forward-backward calculations, and is thus more efficient. The MEMM training converges more quickly, stabilizing after approximately 100 iterations. For each model, The Viterbi algorithm is used to label a test set.

The results of several runs are presented in Figure 3. Each plot compares two classes of models, with each point indicating the error rate for a single test set. As α increases, the error rates generally increase, as the first-order models fail to fit the second-order data. The figure compares models parameterized as $\mu_{y',y,x}$, μ_y, and $\lambda_{y',y}$; results for models parameterized as μ_y, $\lambda_{y,x}$, and $\lambda_{y',y}$ are qualitatively the same. As shown in the first graph, the CRF generally outperforms the MEMM, often by a wide margin of 10%–20% relative error. (The points for very small error rate, with $\alpha < 0.01$, where the MEMM does better than the CRF, are suspected to be the result of an insufficient number of training iterations for the CRF.)

6. Further Aspects of CRFs

There are many aspects of CRFs that are attractive for applications and deserve further study. In this section we briefly mention just two.

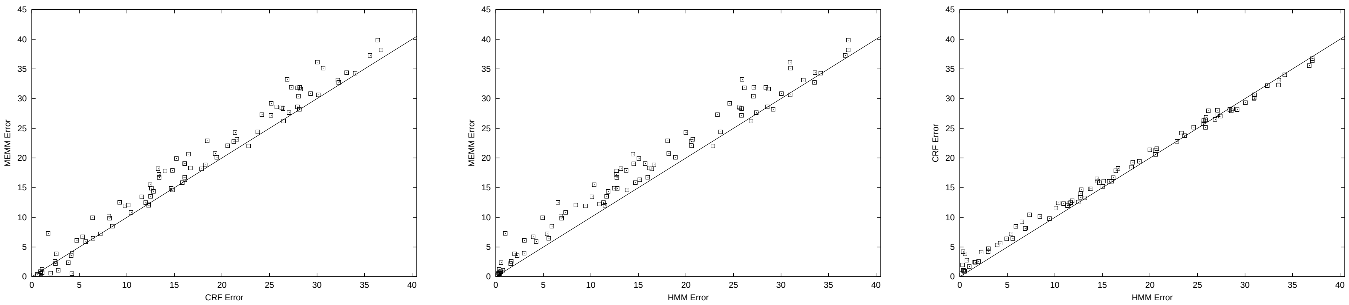

Figure 3. Plots of 2 × 2 error rates for HMMs, CRFs, and MEMMs on randomly generated synthetic data sets, as described in the text. As the data becomes "more second order," the error rates of the test models increases. As shown in left plot, the CRF typically significantly outperforms the MEMM. The center and right plots provide alternative views of this data. Although the data is generated by a mixed-order HMM, the first-order CRF achieves comparable error rate to the first-order HMM, and slightly improves on it when the data is mostly second order. The MEMM, on the other hand, is generally significantly worse. We note that these experiments are not designed to demonstrate the advantages of CRFs over HMMs, such as the ability to handle overlapping, non-local features of the input.

Conditional random fields can be trained using the exponential loss objective function used by the AdaBoost algorithm (Freund & Schapire, 1997). Typically, boosting is applied to classification problems with a small, fixed number of classes; applications of boosting to sequence labeling have treated each label as a separate classification problem (Abney et al., 1999). Given the empirical success of boosting, as well as the interest of the associated theory, an extension of the framework to sequence models may be useful. So far we have been addressing the optimization problem

$$\begin{aligned}\theta^\star &= \arg\min_\theta \sum_{i=1}^N \log \frac{1}{p_\theta(\mathbf{y}^{(i)} \mid \mathbf{x}^{(i)})} \\ &= \arg\min_\theta \sum_{i=1}^N \log \sum_{\mathbf{y}} e^{\,\theta\cdot h(\mathbf{x}^{(i)},\mathbf{y}) - \theta\cdot h(\mathbf{x}^{(i)},\mathbf{y}^{(i)})}\end{aligned}$$

where we define $\theta \cdot h(\mathbf{x},\mathbf{y}) = \sum_{i,k} \lambda_k\, f_k(e_i, \mathbf{y}|_{e_i}, \mathbf{x}) + \sum_{i,k} \mu_k\, g_k(v_i, \mathbf{y}|_{v_i}, \mathbf{x})$. In contrast, the optimization problem for the multi-class boosting algorithm AdaBoost.M2 is

$$\begin{aligned}\theta^\star &= \arg\min_\theta \sum_{i=1}^N \sum_{\mathbf{y}\neq\mathbf{y}^{(i)}} e^{\,\theta\cdot h(\mathbf{x}^{(i)},\mathbf{y}) - \theta\cdot h(\mathbf{x}^{(i)},\mathbf{y}^{(i)})} \\ &= \arg\min_\theta \sum_{i=1}^N \sum_{\mathbf{y}} \frac{e^{\,\theta\cdot h(\mathbf{x}^{(i)},\mathbf{y})}}{e^{\,\theta\cdot h(\mathbf{x}^{(i)},\mathbf{y}^{(i)})}} - N \\ &= \arg\min_\theta \sum_{i=1}^N \frac{1}{p_\theta(\mathbf{y}^{(i)} \mid \mathbf{x}^{(i)})}\end{aligned}$$

This optimization problem cannot be derived as a sampling likelihood for a statistical model. However, it is possible to apply the parallel update algorithm of Collins et al. (2000) to our task. This requires a forward-backward algorithm to compute efficiently certain feature expectations, along the lines of Algorithm T, except that each feature requires a separate set of forward and backward accumulators.

Another attractive aspect of CRFs is that one can implement efficient feature selection and feature induction algorithms for them. That is, rather than pre-specifying which features of $(\mathbf{X}, \mathbf{Y})$ to use in the model, the features can be learned automatically from data. In particular, the feature induction algorithms presented in Della Pietra et al. (1997) can be adapted to use dynamic programming techniques in the setting of conditional random fields.

7. Related Work and Conclusions

As far as we know, the present work is the first to combine the benefits of conditional models with the global normalization of random field models. Other applications of exponential models in sequence modeling have either attempted to build generative models (Rosenfeld, 1997), which involve a hard normalization problem, or adopted local conditional models (Berger et al., 1996; Ratnaparkhi, 1996; McCallum et al., 2000) that suffer potentially from label bias.

Non-probabilistic local decision models have also be widely used in segmentation and tagging (Brill, 1995; Roth, 1998; Abney et al., 1999). The computational complexity of global training forces these models to be trained to minimize the error of individual label decisions under the assumption that neighboring labels are correctly chosen. One would expect label bias to be also a problem for these models.

An alternative approach for applying discriminative models to sequence labeling is to use a permissive generative model, which can only model local dependencies, to produce a list of candidates, and then use a more global model to rerank those candidates. This is the standard approach in large-vocabulary speech recognition (Schwartz & Austin, 1993), and has also been proposed for parsing (Collins, 2000). However, these methods are sub-optimal in that they are subject to failing to find the correct output because it was pruned away in the first pass.

Closest to our proposal are gradient-descent methods that adjust the parameters of all of the local classifiers to minimize a smooth loss function (eg. quadratic loss) combining loss terms for each label. If state dependencies are local, this can be done efficiently with Viterbi training (LeCun et al., 1998). Such methods should alleviate label bias. However, Viterbi training ignores all the but lowest loss labeling for each training instance, and can be very sensitive to local minima. In contrast, our probabilistic setting allows all labelings to be included and weighted by their likelihood.

Conditional random fields offer a unique combination of properties: discriminatively-trained models for sequence segmentation and labeling; combination of arbitrary, overlapping and agglomerative observation features from both the past and future; efficient training and decoding based on dynamic programming; and parameter estimation guaranteed to find the global optimum. In future work, we plan to examine further the exponential loss setting, more general tree-structured random fields, and feature induction methods, and to demonstrate on real-world data that CRFs maintain MEMMs dominance over HMMs, while also allowing the use of complex state structures by avoiding the label bias problem.

References

Abney, S., Schapire, R. E., & Singer, Y. (1999). Boosting applied to tagging and PP attachment. *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

Berger, A. L., Della Pietra, S. A., & Della Pietra, V. J. . a. (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, *22*.

Bottou, L. (1991). *Une approche théorique de l'apprentissage connexionniste: Applications à la reconnaissance de la parole*. Doctoral dissertation, Université de Paris XI.

Brill, E. (1995). Transformation-based error-driven learning and natural language processing: a case study in part of speech tagging. *Computational Linguistics*, *21*, 543–565.

Collins, M. (2000). Discriminative reranking for natural language parsing. *Machine Learning: Proceedings of the Seventeenth International Conference (ICML 2000)*. Stanford, California.

Collins, M., Schapire, R., & Singer, Y. (2000). Logistic regression, AdaBoost, and Bregman distances. *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*.

Darroch, J. N., & Ratcliff, D. (1972). Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, *43*, 1470–1480.

Della Pietra, S., Della Pietra, V., & Lafferty, J. (1997). Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*, 380–393.

Durbin, R., Eddy, S., Krogh, A., & Mitchison, G. (1998). *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*. Cambridge University Press.

Freitag, D., & McCallum, A. (2000). Information extraction with HMM structures learned by stochastic optimization. *Proceedings of the Eighteenth Conference on Artificial Intelligence (AAAI-2000)*.

Freund, Y., & Schapire, R. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, *55*, 119–139.

Hammersley, J., & Clifford, P. (1971). Markov fields on finite graphs and lattices. Unpublished manuscript.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*, 2278–2324.

MacKay, D. J. (1996). Equivalence of linear Boltzmann chains and hidden Markov models. *Neural Computation*, *8*, 178–181.

Manning, C. D., & Schütze, H. (1999). *Foundations of statistical natural language processing*. Cambridge Massachusetts: MIT Press.

McCallum, A., Freitag, D., & Pereira, F. (2000). Maximum entropy Markov models for information extraction and segmentation. *Machine Learning: Proceedings of the Seventeenth International Conference (ICML 2000)* (pp. 591–598). Stanford, California.

Mohri, M. (1997). Finite-state transducers in language and speech processing. *Computational Linguistics*, *23*.

Mohri, M. (2000). Minimization algorithms for sequential transducers. *Theoretical Computer Science*, *234*, 177–201.

Paz, A. (1971). *Introduction to probabilistic automata*. Academic Press.

Punyakanok, V., & Roth, D. (2001). The use of classifiers in sequential inference. *Advances in Neural Information Processing Systems 13*. Forthcoming.

Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging. *Proceedings of the Empirical Methods in Natural Language Conference*.

Rosenfeld, R. (1997). A whole sentence maximum entropy language model. *Proceedings of the IEEE Workshop on Speech Recognition and Understanding*. Santa Barbara, California.

Roth, D. (1998). Learning to resolve natural language ambiguities: a unified approach. *Proceedings of the Fifteenth National Conference on Artificial Intelligence* (pp. 806–813). Menlo Park, California: AAAI Press.

Saul, L., & Jordan, M. (1996). Boltzmann chains and hidden Markov models. *Advances in Neural Information Processing Systems 7*. MIT Press.

Schwartz, R., & Austin, S. (1993). A comparison of several approximate algorithms for finding multiple (N-BEST) sentence hypotheses. *Proc. ICASSP*. Minneapolis, MN.