An Alternate Objective Function for Markovian Fields

Sham Kakade* SHAM@GATSBY.UCL.AC.UK
Yee Whye Teh[†] YWTEH@CS.TORONTO.EDU
Sam T. Roweis[†] ROWEIS@CS.TORONTO.EDU
* Gatsby Computational Neuroscience Unit, UCL, 17 Queen Square, WC1N 3AR, London, UK
[†] Department of Computer Science, University of Toronto, 6 King's College Rd., Toronto M5S 3G4 CANADA

Abstract

In labelling or prediction tasks, a trained model's test performance is often based on the quality of its single-time marginal distributions over labels rather than its joint distribution over label sequences. We propose using a new cost function for discriminative learning that more accurately reflects such test time conditions. We present an efficient method to compute the gradient of this cost for Maximum Entropy Markov Models, Conditional Random Fields, and for an extension of these models involving hidden states. Our experimental results show that the new cost can give significant improvements and that it provides a novel and effective way of dealing with the 'label-bias' problem.

1 Input-Output Markovian Models

Input-output modelling of sequential data is a fundamental problem arising in many machine learning applications, including tracking objects in video streams, labelling/tagging sections of documents, and identifying motifs or functional regions in amino acid or nucleotide chains. The problem can often be cast as one of estimating a "state" or "label" sequence $S = \{s_1, s_2, \ldots s_T\}$ given some observations $X = \{o_1, o_2, \ldots o_T\}$. The labels s_t (states) may be missing at test time only, or both at training and testing time. In practice, the observations o_t are often noncausal feature vectors, designed by the algorithm implementer, which summarize a more complex underlying raw data stream by indicating the presence of certain elements at the current time as well as in the past/future. The goal is to model certain aspects of the joint distribution over states and labels.

In this paper we focus on the supervised learning setting in which the labels are (at least partially) observed at training time and the task at test time is to predict labels for a given sequence of input observations. We can either consider online prediction (filtering) in which only the observations up to time t are given or delayed processing (smoothing) in which all observations are given. In these settings, a density model of the observations is not required for good performance; indeed it could be a liability. Popular generative models for sequential data such as the hidden Markov model (HMM) (see figure 1) are not necessarily appropriate, since they are attempting to solve the more difficult problem of learning the full distribution and thus often require more training data in order to achieve good performance. Furthermore, any such causal directed graphical model that attempts to capture the full joint distribution over outputs and labels cannot use large, rich, nonlocal features in the observation stream.

A more appropriate approach is to model only the *conditional* distribution of label sequences given the features. Two related architectures have recently emerged for capturing this dependence, both of which allow the use of rich nonlocal features as observations (see figure 1). McCallum et al. (2000) proposed the Maximum Entropy Markov Model (MEMM) which learns an observation-dependent transition from the previous label to the current label. While MEMMs are an improvement over the generative HMM, they suffer a "label-bias" problem, which, crudely speaking, unfairly favours labels with few successors. In an attempt to address this limitation, Lafferty et al. (2001) proposed the Conditional Random Field (CRF), an undirected model which normalizes the probability of a label sequence globally rather than locally.

Although there has been considerable investigation into architectures and representations for input-output sequence models, discussion of appropriate objective functions for training these architectures has been relatively absent. In this paper we focus on the latter question: what should a model capture about the conditional label sequence distribution $p(S|X)$? We present

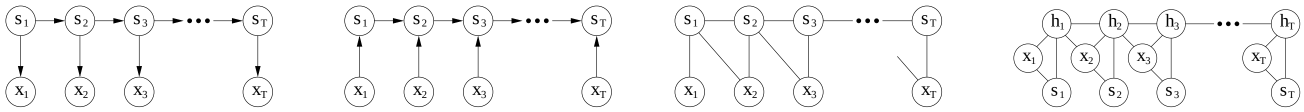

Figure 1. Graphical models for HMM, MEMM, CRF and HRF (described in text).

a new objective function, derive efficient algorithms for training Markovian models using it, and demonstrate its improved performance on several tasks. Motivated by our results, we also introduce an extension to the CRF model, Hidden Random Fields (HRF).

2 A New Objective Function

In many practical applications, modelling the full conditional distribution of labels given observations might be unnecessary. For example, when identifying functional regions in nucleotide chains our goal might be to minimize the total number of errors in labelling. We might not be interested in modelling correlations of where these functional regions occur with respect to each other. Similarly, in visual tracking tasks, predicting an object's location accurately at each timestep may be important even if the sequence of predictions is not a probable trajectory. In such cases, we are only interested in the *marginal conditional distributions* $P(s_t|X)$ rather than the full *joint conditional distribution* $P(S|X)$. Below, we introduce a cost function that penalizes a model based only on the quality of these single-time marginal distributions.

The standard objective function for Markovian sequence-label models is the log probability of the entire label sequence given the observation sequence:

$$C_0(\theta; S, X) = \log p(S|X;\theta) = \log p(s_1^T|X;\theta) \qquad (1)$$

where we use notation $s_{t_1}^{t_2} = \{s_{t_1}, \dots, s_{t_2}\}$. Optimizing this objective on a training set attempts to maximize the probability of correctly labelling an entire observation sequence. The approach at test time is to estimate the labelling using the most likely state sequence (the Viterbi path). If this accurately reflects the task at hand, this cost is appropriate.

However, often the real criterion at test time is to maximize the expected number of correct labels, or equivalently to minimize the number of mislabellings, even if very few input sequences are jointly labelled perfectly. A simple decomposition of C_0 above reveals a crucial assumption underlying why training with the standard objective function is not appropriate for this criterion. Using the "chain rule" of conditional probability:

$$C_0(\theta; S, X) = \sum_t \log p(s_t|s_1^{t-1}, X;\theta) \qquad (2)$$

We see that models trained using C_0 are learning to predict the next label given the observations and *the correct labels up to that point.*

In the spirit of discriminative learning, we consider a new objective, the average *single-time prediction* costs, that is better suited to this goal:

$$C_1(\theta; S, X) = \frac{1}{T}\sum_t \log p(s_t|X;\theta) \qquad (3)$$

which implicitly integrates over the model's uncertainly about all the labels before time t when predicting the label at time t.

This cost function is only concerned with the quality of the marginals $p(s_t|X;\theta)$. At test time, we generate a labelling using the "gamma-path", taken by choosing the best state from each marginal. Such labellings may have very poor (or zero) probabilities under the joint model $p(S|X)$ but will have fewer single-time errors.

Of course, if a model's joint distribution $p(S|X)$ over the sequence of labels is correct then this implies its marginal distributions $p(s_t|X)$ are also correct. However, with limited data it is often difficult to accurately model sophisticated distributions (e.g. with long range dependencies). If our performance depends only on accurate estimation of marginal distributions and not on correctly modelling the entire joint distribution, we may be better off learning these quantities directly than learning a more complex model. This is analogous to the distinction between generative and discriminative learning methods for classification: generative methods learn a joint model of inputs and class labels and appeal to its conditional at test time, while discriminative models do not attempt to capture the joint but rather model the conditional directly.

3 MaxEnt & Random Field Models

Below we discuss how our new objective function C_1 can affect the behaviour and performance of the MEMM, CRF and HRF (our extension).

3.1 MEMMs and Label-Bias

A MEMM defines a conditional distribution over labels S given features X:

$$p(S|X,\theta) = \prod_{t=1}^{T} p(s_t|s_{t-1}, x_t) \qquad (4)$$

where for simplicity we assume s_0 is a fixed initial state. *Here, and throughout the paper, x_t can be a feature of the entire observation stream X.* The individual conditional distributions are parameterized as maximum entropy logistic models

$$p(s_t|s_{t-1}, x_t) = \frac{1}{Z} \exp\left(\sum_k \lambda_k f_k(s_{t-1}^t, x_t)\right) \quad (5)$$

The predefined potentials f_k describe how s_t depends on both s_{t-1} and x_t, and the learned weights λ_k determine the contribution of each dependence f_k. Z is a local normalization constant. Since x_t is a feature on X, it is unnecessary to write f_k as a function of X.

MEMMs suffer from the so called "label-bias" problem (Bottou, 1991). As a didactic example, we revisit the 'rib-rob' problem discussed in Lafferty et al. (2001). The training data consists of the observation sequence $rib\_$ labelled 0123 and the sequence $rob\_$ labelled 0453, both of which occur with equal probabilities. Let us consider the case where x_t is simply the observed letter at time t. Note that the training data never contain an observation of $-o-$ corresponding to a transition from 1 to 2. Thus, C_0 cannot accurately estimate this transition probability, $P(2|1, -o-)$, since this term does not appear in the cost function (see equation 2). However, at test time (even if we only test on the training data), the model needs to evaluate $P(0123|rob)$ which requires an estimate of $P(2|1, -o-)$. In general, we can think of equation 4 as a way of redistributing probability mass from s_t to s_{t+1}. With the objective C_0, the MEMM cannot learn how to do this redistribution for situations not represented in the training corpus.

In contrast, using the new objective C_1, training will implicitly integrate over the uncertain previous labels at each time step. In the example above, the MEMM will be forced to estimate $P(2|1, -o-)$ even though this "impossible" transition was not present in the training corpus. To see this, note that the marginal distribution, which is used in training, for a given label l_t at time t is:

$$p(l_t|x_1^t) = \sum_{s_{t-1}} p(l_t|s_{t-1}, x_t) p(s_{t-1}|x_1^{t-1}) \quad (6)$$

which shows that under C_1, training is dependent on all possible previous states s_{t-1}, not only on those seen in the training set.

When an MEMM is trained on this toy example under C_0, the "per-symbol" labelling error is 33% (not including label 0). Under C_1, this error rate is roughly 17%, since the model still makes errors in labelling the states corresponding to the observation $-r-$, but no longer makes errors on $-o-$ and $-i-$.

In section 5, we provide experimental results demonstrating that this improvement is not trivial: our new objective function significantly outperforms the standard one for training the MEMM architecture on a synthetic robot navigation problem and on a real document labelling task.

3.2 Sequential Random Fields

Although training with C_1 can alleviate the "label-bias" problem in the MEMM architecture, it can be argued that the problem is with the model itself and not with the standard objective function C_0. Indeed, the CRF (Lafferty et al., 2001) provides a more principled solution by changing the basic model. In effect, a CRF infers the label at time t (given the observations) based on its inferred belief about other labels both before *and after* time t. In contrast, the MEMM infers the label at time t based only on how it infers the labels before time t. In fact, for the previous 'rib-rob' example,the CRF error rate is 0. We now consider the difference between the two cost functions when using the CRF and the HRF, an extension incorporating hidden states (see figure 1).

3.2.1 Conditional Random Fields

The CRF defines a conditional distribution over labels:

$$p(S|X, \theta) = \frac{1}{Z} \prod_{t=1}^{T} \exp\left(\sum_k \lambda_k f_k(s_{t-1}^t, x_t)\right) \quad (7)$$

Again for simplicity we assume a fixed initial state s_0. The potentials f_k are again hand-crafted while λ_k's are parameters to be learned from data. Crucially, the CRF has a joint (global) normalization factor Z which allows it to circumvent the 'label-bias' problem (see (Lafferty et al., 2001)).

The difference between the objectives C_0 and C_1 is considerably more subtle in the CRF case than in the MEMM case since the CRF does not suffer from the label-bias problem. How, then, will C_1 set the feature weights differently than C_0?

Consider a simple situation in which a particular transition is never observed. If there is feature that is nonzero only for this transition, then under C_0 the model must give this feature an infinitely large negative weight. However, since the real world often violates our modelling assumptions, such a feature may still be useful in improving the marginal distributions, even at the expense of a worse joint distribution.

We demonstrate this difference with another didactic example, which has four labels $1, 2, 3, 4$ and two observations a, b. The generative process is as follows: while the observations are $-a-$ the labels always alternate between 1 and 2, and while the observations are $-b-$, the labels always alternate between 3 and 4. The process switches between these two modes with equal probability, and spends, on average, the same amount of time in each mode. Thus, the minimal possible labelling error rate is 50%.

An example sequence looks like this:

```
label      : 121234343212121243434343212434341212 1
observation: aaaabbbbbaaaaaaabbbbbbbbaaabbbbbaaaaa
```

Let us consider an impoverished model with only one feature. The binary feature $f(s^i_{i-1}, x_i)$ is equal to 1 only for the arguments: $(\{1,1\}, a)$, $(\{2,2\}, a)$, $(\{3,3\}, b)$, or $(\{4,4\}, b)$. Note that these self transitions will never occur in a training set. Hence, under C_0, the feature must be given a large negative weight. We find that the resulting error rate is close to 100% (since this large weight forces the incorrect labellings of 3 or 4 with an $-a-$ and vice versa). In contrast, under C_1, this feature is given a large positive weight, since it accurately captures the marginal distributions, and we find that the resulting error rate is roughly 50% (which is optimal).

This example was constructed to demonstrate a potential tradeoff in modelling the joint vs. the marginal distributions. In section 5, we give experimental results for a toy HMM and a document labelling task.

3.2.2 Hidden Random Fields

As conditional models, both the CRF and MEMM are designed to be able to use long range, nonlocal features as observations. However, these models still assume a Markovian structure among the labels. Extending the CRF, we consider the Hidden Random Field (HRF) model shown in figure 1. A HRF is like a CRF but with a hidden Markov model on the hidden states.[1] This allows past information in the label history to be held in the hidden states and used, along with the features, to predict future labels. (Note however, that these hidden models no longer share the convexity properties of maximum entropy models Della Pietra et al. (1997); thus they are susceptible to local optima problems during training.)

The conditional distribution of the labels for HRFs is

$$p(s_1^T|X) = \frac{1}{Z}\sum_{h_1^T}\exp\{\sum_{t=1}^{T}\sum_k \lambda_k f_k(h_{t-1}^t, x_t) + \sum_j \alpha_j g_j(s_t, h_t, x_t)\} \quad (8)$$

[1] We could also consider a slightly more powerful model by adding links $\{s_i, s_{i+1}\}$ to the graphical model, which permits Markov dependence directly between the labels.

As in CRFs, there is a joint normalization factor. Note that the distribution integrates out the hidden states.

An important distinction with the CRF is that the hidden states are not given during training. Hence, the hidden states must be integrated over to compute either cost function. This model is significantly more unconstrained since both cost functions do not explicitly depend on the joint distribution of the hidden states. We give an example in Section 5 that demonstrates how C_1 can obtain more accurate marginal distributions than C_0 using this additional freedom.

4 Parameter Estimation

In this section we derive an algorithm to efficiently optimize C_1 for a CRF. Algorithms for the MEMM and HRF can be derived similarly. Most of the work is involved in computing sufficient statistics corresponding to each parameter. This can then be used by various optimizers like conjugate gradient or iterative scaling type algorithms to perform the actual update[2]. In the case of an MEMM sometimes a full M-step is even feasible.

Let $\hat{l}_1^T$ be the given sequence of labels. Using the definition of a CRF (equation 7) and differentiating C_1 with respect to λ_k, we obtain

$$\begin{aligned}\frac{\partial C_1}{\partial \lambda_k} =& \frac{1}{T}\sum_{t,i}\langle f_k(s^i_{i-1}, x_i)\rangle_{p(s^i_{i-1}|s_t=\hat{l}_t, X, \theta)} \\ &- \sum_i \langle f_k(s^i_{i-1}, x_i)\rangle_{p(s^i_{i-1}|X,\theta)}\end{aligned} \quad (9)$$

where $\langle f\rangle_p$ is the expectation of f under distribution p. The second term of (9) can be calculated using belief propagation to compute the joint probabilities

$$\mathrm{w}_i(l^i_{i-1}) = p(s^i_{i-1} = l^i_{i-1}|X, \theta) \quad (10)$$

Here, each l_i denotes a value that variable s_i can take. Then the expectations of f_k are taken explicitly. The first term of (9) can be similarly calculated if we have the following quantities:

$$\omega_i(l^i_{i-1}) = \sum_{t=1}^T p(s^i_{i-1} = l^i_{i-1}|s_t = \hat{l}_t, X, \theta) \quad (11)$$

Using another two passes over the network, we compute the following "forward" and "backward" sums

$$\omega^f_i(l^i_{i-1}) = \sum_{t=1}^{i-1} p(s^i_{i-1} = l^i_{i-1}|s_t = \hat{l}_t, X, \theta) \quad (12)$$
$$\omega^b_i(l^i_{i-1}) = \sum_{t=i}^{T} p(s^i_{i-1} = l^i_{i-1}|s_t = \hat{l}_t, X, \theta) \quad (13)$$

Then $\omega_i(l^i_{i-1}) = \omega^f_i(l^i_{i-1}) + \omega^b_i(s^i_{i-1})$. The w^f_i are com-

[2] Some results (Minka, 2001) suggest that conjugate gradient is more efficient than iterative scaling.

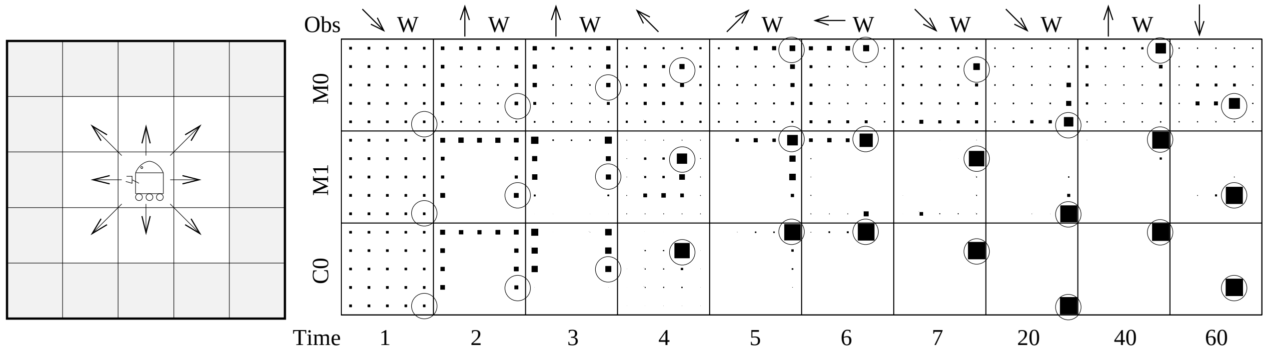

Figure 2. Left: The blind robot problem; the shaded locations indicate where the proximity sensor detects a wall. Right: Predicted distributions over the location of the robot by the 3 models. At the top are the observations (the direction of last movement, and whether it is next to a wall). In each square the area of the blobs correspond to the probability of being in that location, and the circle denotes the true location.

puted using the "forward" recursion:

$$\nu^f_{i+1}(l_i) = \begin{cases} \delta_{l_i,\hat{l}_i} & i = 1 \\ \delta_{l_i,\hat{l}_i} + \sum_{l_{i-1}} \omega^f_i(l^i_{i-1}) & i > 1 \end{cases} \quad (14)$$

$$\omega^f_{i+1}(l^{i+1}_i) = \mathrm{w}_{i+1}(s_{i+1} = l_{i+1}|s_i = l_i)\nu^f_{i+1}(l_i) \quad (15)$$

where $\delta_{a,b} = 1$ if $a = b$ and 0 otherwise, and ν^f_i are intermediate values used in the recursion. Similarly, the "backward" recursion is:

$$\nu^b_i(l_i) = \begin{cases} \delta_{l_i,\hat{l}_i} & i = T \\ \delta_{l_i,\hat{l}_i} + \sum_{l_{i+1}} \omega^b_{i+1}(l^{i+1}_i) & i < T \end{cases} \quad (16)$$

$$\omega^b_i(l^i_{i-1}) = \mathrm{w}_i(s_{i-1} = l_{i-1}|s_i = l_i)\nu^b_i(l_i) \quad (17)$$

These recursions are as efficient as the belief propagation updates used to compute w_i (up to a constant).

5 Results

We have tested our new cost function on several synthetic examples and on a real document labelling task, demonstrating its superior performance in various cases and the practicality of the associated training algorithms for MEMMs, CRFs and HRFs.

One could also consider training vanilla HMMs with different cost functions. However, as discussed in (Lafferty et al., 2001), a discriminatively trained HMM, under C_0, is equivalent to a CRF trained with C_0 using table based features. Similarly, a HMM discriminatively trained with C_1 is equivalent to a CRF trained with C_1 using table based features. For both these cases we present results below. It is also possible to consider training the HMM with the naive extension $\sum_t \log p(s_t, X)$ of the standard HMM cost function $\log p(S, X)$. However, this naive extension performs particularly poorly due a drastic tendency to focus on modelling the entire observation sequence X rather than each s_t; thus these results are not appropriate for comparison.

5.1 Blind Robot Example

The blind robot example of this section highlights the difference between optimizing the ordinary cost function C_0 and the proposed cost C_1 for the MEMM.

We have a robot moving in a grid world. Initially the robot's location is uniformly distributed across the room. At each step the robot moves in any of 8 compass directions, and a proximity sensor tells it whether it is touching a wall or not (see figure 2). We assume that there is no noise in the system. Given an unknown starting location, the sequence of movements of the robot, and wall sensor readings, we wish to predict the location of the robot. We compared three models: an MEMM trained with the old cost function ($M0$), the MEMM but trained with the new cost function ($M1$), and a CRF trained with the old cost function ($C0$). The performance using a CRF trained with C_1 is essentially identical to the performance with C_0 and is not shown. The MEMMs were trained with full M steps, and the CRF was trained using conjugate gradient. Full table-based parameterizations were used in all three models.

The right panel of figure 2 shows a typical observation sequence for a 5x5 world, and the corresponding distribution over locations (labels) predicted by all three models. As expected, $C0$ was able to locate the robot quickly, as was $M1$, while $M0$ performed the worst. Notice that at time 4, $C0$ was able to determine that only one out of the six possible locations at time 3 was possible; $M1$ was also able to predict the true location

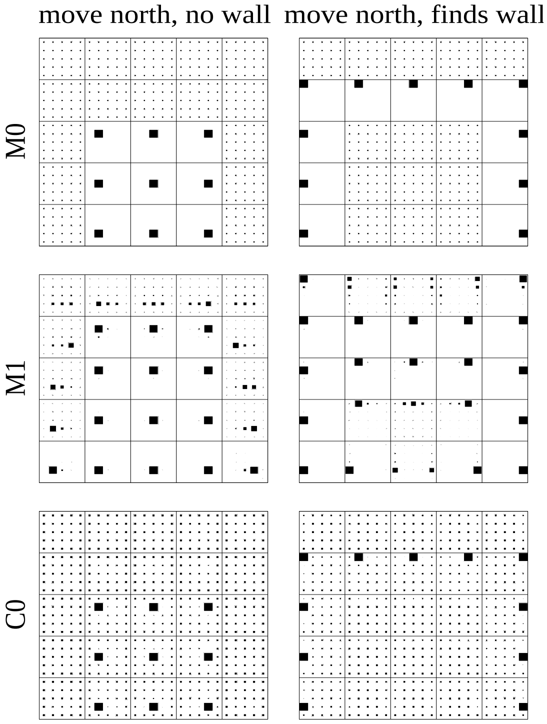

Figure 3. The features learned by the three models. The location of each small square within the 5x5 grid corresponds to the previous location of the robot, and each blob in the squares corresponds to the next location. For the MEMMs, the size of each blob describes the probability of transiting to the next location, while for the CRF it is related to the chance of seeing both locations together.

quite accurately. However, $M0$ was not able to do well because most of the probability mass over the previous location is concentrated on those locations which are inconsistent with the observations at time 4. In figure 3, we show the parameters learned by the models for observations {`north,wall`} (right column) and {`north,no-wall`} (left). (See figure caption.) The parameters for other observations follow similar patterns. As expected, both $M0$ and $C0$ learned sensible features when the current observations are consistent with the previous location, but do not learn anything when they are inconsistent. However, due to the 'label-bias' problem, $M0$ does not learn appropriate features for the inconsistent {`north,wall`} case (see figure 3). On the other hand, $M1$ learns features which let it predict a likely location of the robot even when the previous location is inconsistent. This location is normally the one consistent with the current observations and closest to where the robot would have been if the previous location were correct.

5.2 The FAQ Dataset

We tested the viability of the new cost function on the Frequently Asked Questions (FAQ) dataset introduced by McCallum et al. (2000). The data consists of 38 files belonging to 7 Usenet newsgroup FAQs. Each file consists of a header, followed by question/answer pairs, and ends with a tail section. The task is to label the 300–2500 lines in each file according to whether it is in the header (H), a question (Q), an answer (A) or in the tail (T) using a set of 24 features such as begins-with-number, begins-with-question-word, and indented-1-to-4 (see (McCallum et al., 2000) for the full list).

We trained a HMM by optimizing the full joint distribution over labels and observations (which are the features), and MEMMs and CRFs using both the C_0 and C_1 objective. A zero-mean Gaussian prior with a variance of 10 is imposed on the weights of each model for weight decay. For each FAQ, we performed leave-1-out evaluation by training the models on all but one file which we reserved for testing.

Table 1 (top) shows the prediction errors given only the observation sequence of each test case. The prediction error is calculated as the percentage of lines labelled wrongly over the 38 tests. The HMM performed the worst here as expected. Training the MEMM with the C_1 objective lowered the error by 84.8% over using C_0. This significant improvement is due to $M1$ capturing the marginal distributions over labels more accurately using C_1 — though it is not clear if the 'label-bias' problem has a role here. For the CRF, training with C_1 gives a substantial although less dramatic improvement of 16.2% over C_0. It is possible that the CRF performed worse than the MEMM due to overfitting – observe the low training errors of the CRFs.

Figure 4 shows a histogram of the probabilities the model assigned to the correct labels in the test set while figure 6 shows the predicted distribution over the labels for each line in the FAQs. We see a drastic improvement for $M1$.

Table 1 (bottom) show the errors in predicting each label s_t for the unusual case that we are given the observations X and *the correct sequence of labels* s_1^{t-1} *up to time* t. The models trained under C_0 performed better than those trained with C_1. This supports the argument in section 2 that the C_0 objective minimizes the error of predicting the correct label given the observations and the correct labels up to that point (see (2)), and shows that under C_0 some aspects of the joint distribution are more accurately captured.

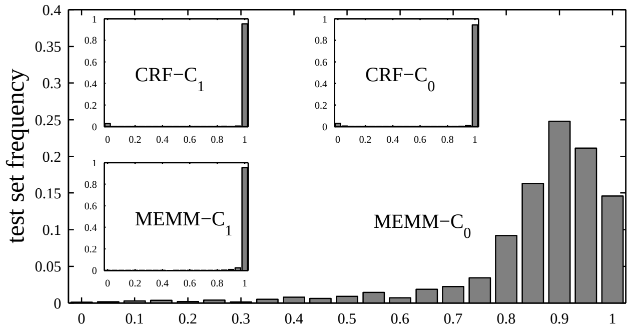

Figure 4. Histogram plots of the probability mass that various models assigned to the correct label in the test set. Note that $M1$ assigns probabilities close to 0 significantly less frequently than MEMM–C_0 or CRF–C_1/C_0.

	Test error (%)	Training error (%)
HMM	14.71	8.20
MEMM-C_0	3.57	0.78
MEMM-C_1	0.54	0.22
CRF-C_0	3.53	0.02
CRF-C_1	2.96	0.02
HMM	4.23	2.52
MEMM-C_0	0.08	0.05
MEMM-C_1	0.16	0.08
CRF-C_0	0.25	0.01
CRF-C_1	0.25	0.01

Table 1. Errors on the FAQ dataset for various models and test conditions. Top: Observations are given at test time. Bottom: Unusual test condition where both the observations and *the correct labels up to that point* are given.

5.3 Conditional Random Fields

As an illustrative example, consider training a CRF on data generated from a simple 4 state HMM with 4 labels. All states i can transition only to i or $i+1$ with the probability p and $1-p$, and we assume circular boundary conditions ($4 \rightarrow 1$). Also, each state i emits the observation i with probability q or else emits $i+1$ or $i-1$ with equal probability. If a full table based parameterization is used, then both models perform equally well.

However, we consider an interesting variant where we "corrupt" each feature. In addition to the feature $f_{i,j}(s^t_{t-1})$ being one if $s^t_{t-1} = \{i,j\}$, the feature now takes on a positive value for another transition. We set this transition and value randomly, and the state-observation features are similarly corrupted. We tested how well the CRF model performs for a variety of these corrupted features and data sequences (using different p's and q's). Figure 5 shows that the objective C_1 often outperforms C_0. Informally, these results vary depending on the parameters and the amount of data, though we rarely find C_1 performing worse than C_0. The results from the FAQ dataset also show a significant improvement with C_1.

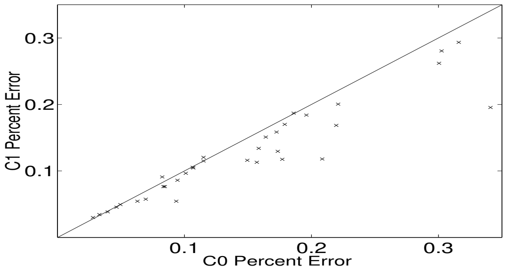

Figure 5. C_0 vs. C_1 error rates for CRF models that have 'corrupted' table based features (see text). The region below the diagonal line is where C_1 outperforms C_0

5.4 Hidden Random Fields

As a final demonstration of the different objectives, we trained a simple Hidden Random Field model on a synthetic example requiring the labeler to remember state information about the past. The task has two labels, 0 and 1, and four observations, A, B, R, I. When the observation is A the label is 1, and when B the label is 0. Obviously neither model has problems learning this aspect. In addition, there is a "resume" observation R, which repeats the most recent label from the A/B observation. One bit of memory is required to label R correctly. Finally, when the observation is the "interrupt" I, the the labels "cycle", inverting the value of the label when the observation was *last equal* to I. Again, one bit of memory is needed to model the joint label distribution during the interrupt I. However, due to randomness, it is never possible to distinguish between 0101 or 1010 during the interrupt sequence (thus the error rate will be 50% during the Is). The standard CRF clearly cannot correctly label the R's with no memory.

To obtain the minimal error rate, all that is required is to use only one bit of information to use with the "Resume" mode. However, two bits of information are required to capture the joint distribution. An HRF with only a single bit (two hidden states) is forced to chose between modelling the "Resume" or "Interrupt" mode. We trained an HRF with C_0 (model $H0$) and C_1 (model $H1$). Under C_1, the model appropriately uses its memory, while under C_0 it often chooses to model "Interrupt" mode (during which the error rate is 50%). Below we show the hidden states, observations, and label performance (errors are X) of both models over two runs.

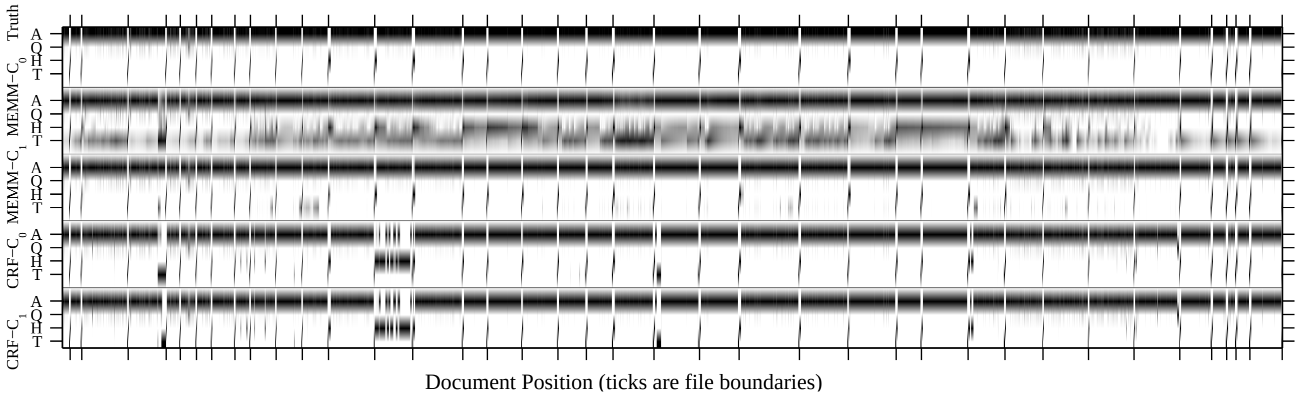

Figure 6. FAQ line labelling results. Each block of four horizontal strips shows the log probability assigned to each of the four labels (Answer,Question,Head,Tail) by various models. Horizontal position moves through the data; the tick marks at top and bottom denote the file boundaries. The intensity of each strip corresponds to log probability assigned to a label given the features of the line in the FAQ file. Blocks from top to bottom: true labels, $M0$, $M1$, $C0$, $C1$.

```
H0 hid state:22211221112211222211222112121111211111222211121112
H0 errors   :  XXXXXXXXXXXXXXXXXXXXXXXXXXX   XXX X X XXXX XX  X
Observations:BBRIRIRIRRIRIRIRRRIRIRRIRIIIIAABIIRBRBIBRRIRAIIRAI
H1 errors   :     X    X   X     X    X X    X     X       X
H1 hid state:22222222222222222222222222222112222222222222111111

H0 hid state:21222111122221112222111222211111212221121211122112
H0 errors   :               X XX                   X    X  X X
Observations:AIIRRIRRRIRRRIBRIRRIAAIRRRRIBARBIIIBBIRIIIIRBIRIRI
H1 errors   :  X      X         X  X          X   X  X X    X
H1 hid state:11111111111111222222211111111211222222222222222222
```

More experimental results are needed to understand how fruitful this extension is.

6 Discussion & Conclusions

With limited data, it is important to match model training with test conditions. In the spirit of discriminative training, we have introduced an objective function that more closely resembles a test condition of common interest. The very simple idea is that in many cases test performance only depends on the quality of our model's marginal conditional distribution, rather than a good joint conditional distribution.

We have demonstrated significant improvements in both synthetic and real-world problems of C_1 over C_0 for a class of Markovian models. The most drastic improvement is for the MEMM. We find that the our new cost function provides a novel and interesting way to deal with the tricky label-bias problem. Even though the CRF solves this problem in a more principled manner, it is not clear that the CRF is always preferable to the MEMM. (For example, our FAQ experiment shows the MEMM under C_1 significantly outperforming other models.) In some cases the MEMM architecture captures important aspects of the underlying process, and we have provided a way of retaining the benefits of this architecture while still solving the label-bias problem. Finally, our new objective function C_1 can be used to train MEMM and CRF models just as efficiently as the standard joint likelihood.

Acknowledgments

We thank Peter Dayan for many helpful discussions, Andrew McCallum for providing the FAQ data, and our reviewers for their very useful comments.

References

Bottou, L. (1991). Une approche th eorique de l'apprentissage connexionniste: Applications a la reconnaissance de la parole.

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

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proc. 18th International Conf. on Machine Learning* (pp. 282–289). Morgan Kaufmann, San Francisco, CA.

McCallum, A., Freitag, D., & Pereira, F. (2000). Maximum entropy Markov models for information extraction and segmentation. *Proc. 17th International Conf. on Machine Learning* (pp. 591–598). Morgan Kaufmann, San Francisco, CA.

Minka, T. (2001). Algorithms for maximum-likelihood logistic regression. *TechReport. Carnegie Melon Universtiy.*