MAX-MARGIN MIN-ENTROPY MODELS

Kevin Miller, M. Pawan Kumar, Ben Packer, Danny Goodman, Daphne Koller

Max-Margin Min-Entropy Models

Kevin Miller
Stanford University

M. Pawan Kumar
Ecole Centrale Paris & INRIA Saclay

Ben Packer
Stanford University

Danny Goodman
Stanford University

Daphne Koller
Stanford University

Abstract

We propose a new family of latent variable models called max-margin min-entropy (M3E) models, which define a distribution over the output and the hidden variables conditioned on the input. Given an input, an M3E model predicts the output with the smallest corresponding *Rényi entropy of generalized distribution*. This is equivalent to minimizing a score that consists of two terms: (i) the negative log-likelihood of the output, ensuring that the output has a high probability; and (ii) a measure of uncertainty over the distribution of the hidden variables conditioned on the input and the output, ensuring that there is little confusion in the values of the hidden variables. Given a training dataset, the parameters of an M3E model are learned by maximizing the margin between the Rényi entropies of the ground-truth output and all other incorrect outputs. Training an M3E can be viewed as minimizing an upper bound on a user-defined loss, and includes, as a special case, the latent support vector machine framework. We demonstrate the efficacy of M3E models on two standard machine learning applications, discriminative motif finding and image classification, using publicly available datasets.

1 Introduction

Latent variable models (LVM) provide an elegant formulation for several applications of practical importance. For example, in computer vision, we may wish to learn a model of an object category such as 'car' from images where the location of the car is unknown, and is therefore treated as a latent (or hidden) variable. In computational medicine, we may wish to diagnose a patient based on the observed symptoms as well as other unknown factors—represented using hidden variables—such as the family's medical history.

An LVM consists of three types of variables: (i) the observed variables, or input, whose values are known during both training and testing; (ii) the unobserved variables, or output, whose values are known only during training; and (iii) the hidden variables, whose values are unknown during both training and testing. An LVM models the distribution of the output and hidden variables conditioned on, or jointly with, the input. Modeling the conditional distribution results in discriminative LVMs, while modeling the joint distribution results in generative LVMs. Given an input, the output is typically predicted by either (i) computing the most probable assignment of the output and the hidden variables according to the aforementioned distribution [5, 25]; or (ii) computing the most probable assignment of the output by marginalizing out the hidden variables [4]. Both these prediction criteria ignore an important factor: how certain are we about the values of the hidden variables for the predicted output? Since the underlying assumption of LVM is that the hidden variables provide useful cues for predicting the output, we argue that minimizing the confusion in their values will help improve the accuracy of the model. Furthermore, in many cases there is value in obtaining an estimate of the hidden variables themselves. For example, using an LVM for a 'car' we would like not only to classify an image as containing a car or not, but also predict the location of the car if present.

We propose a novel family of discriminative LVMs, called max-margin min-entropy (M3E) models, that predicts the output by minimizing the Rényi entropy [18] of the corresponding *generalized distribu-*

Appearing in Proceedings of the 15^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2012, La Palma, Canary Islands. Volume XX of JMLR: W&CP XX.

tion (that is, the unnormalized part of the distribution that models the output under consideration). This amounts to minimizing a score that consists of two terms: (i) the negative log-likelihood of the output obtained by marginalizing the hidden variables; and (ii) the Rényi entropy of the normalized conditional probability of the hidden variables given the input and the output. In other words, the predicted output not only has a high probability, but also minimizes the uncertainty in the values of the hidden variables.

Given a training dataset, the parameters of an M3E model are learned by maximizing the margin between the Rényi entropies of the generalized distributions corresponding to the ground-truth output and all other outputs. Intuitively, this ensures that the output of a training sample is correctly predicted by the model. We show that the corresponding optimization problem amounts to minimizing an upper bound on a user-defined loss over the training dataset. Furthermore, we show that the M3E family includes, as a special case, the latent support vector machine (or latent SVM for short) formulation [5, 25].

In order to use the M3E family of models in practice, we propose an efficient trust region style algorithm for learning their parameters. Our approach relies only on a solver for structured support vector machine (or structured SVM for short) problems [23, 24], of which there are several reported in the literature [13, 20]. Our algorithm is directly applicable for problems where the space of latent variables is tractable (for example, small number of latent variables with a small number of putative values, or when the underlying graphical model is a tree). When faced with an intractable latent space, similar to other LVMs, we can resort to approximate inference schemes in order to obtain an estimate of the Rényi entropy.

We demonstrate the efficacy of M3E models on two standard machine learning applications using publicly available datasets: discriminative motif finding and image classification.

2 Related Work

The most commonly used method for learning the parameters of an LVM is the expectation-maximization (EM) algorithm [4, 22], or its many variants [6, 16], including discriminative EM [19]. The EM algorithm attempts to maximize the expected likelihood of the training data, where the expectation is taken over a distribution of the hidden variables. Once the parameters are learned, the output of the test sample is typically predicted by marginalizing out the hidden variables (corresponding to the objective optimized by *soft* EM) or by maximizing the joint probability of the output and the hidden variables (corresponding to the objective optimized by *hard* EM, which approximates the expectation by a pointwise estimate). As argued earlier, predicting the output in this manner does not take into account any measure of uncertainty in the values of the hidden variables.

Recently, Felzenszwalb *et al.* [5] and Yu and Joachims [25] independently proposed the latent SVM framework, that extends the structured SVM [23, 24] to handle hidden variables. The parameters of a latent SVM are learned by minimizing an upper bound on a user-defined loss, a process that is closely related to hard EM. The latent SVM formulation has steadily gained popularity, not least because its parameter learning problem only requires a *maximum a posteriori* inference algorithm—a well-studied problem with several accurate approximate (and in some cases, exact) methods. In section 5, we will show that latent SVM can be viewed as a special case of the M3E family.

Finally, we note that there have been several works reported in the literature based on the principle of maximum entropy [10], including classification [9] and feature selection [12]. Maximum entropy classification has also been extended to handle hidden variables [11]. However, unlike M3E, maximum entropy methods measure the entropy of the input and the output, and not the entropy of the hidden variables (which are, in fact, marginalized out).

3 Preliminaries

Notation. We denote the input by $\mathbf{x} \in \mathcal{X}$, the output by $\mathbf{y} \in \mathcal{Y}$ and the hidden variables by $\mathbf{h} \in \mathcal{H}$. As mentioned earlier, the value of input $\mathbf{x}$ is known during both training and testing, the value of the output $\mathbf{y}$ is only known during training and the value of the hidden variables $\mathbf{h}$ is not known during either training or testing. We denote the parameters of our model by $\mathbf{w}$. For simplicity, we assume a discrete setting. In this case, the conditional probability of the output and the hidden variables, given the input, can be viewed as a set $\mathcal{P}_{\mathbf{x}} = \{\Pr(\mathbf{y}, \mathbf{h}|\mathbf{x}; \mathbf{w}), \forall (\mathbf{y}, \mathbf{h}) \in \mathcal{Y} \times \mathcal{H}\}$, whose elements are non-negative and sum to one. Furthermore, we denote the conditional probability of the hidden variables, given the input and a particular output $\mathbf{y}$, as the set $\mathcal{P}_{\mathbf{x}}^{\mathbf{y}} = \{\Pr(\mathbf{h}|\mathbf{y}, \mathbf{x}; \mathbf{w}), \forall \mathbf{h} \in \mathcal{H}\}$. A generalized distribution refers to a subset of the distribution $\mathcal{P}_{\mathbf{x}}$ [18]. Of particular interest to us are those subsets that correspond to a particular output $\mathbf{y}$, that is, $\mathcal{Q}_{\mathbf{x}}^{\mathbf{y}} = \{\Pr(\mathbf{y}, \mathbf{h}|\mathbf{x}; \mathbf{w}), \forall \mathbf{h} \in \mathcal{H}\}$, where we use $\mathcal{Q}$ instead of $\mathcal{P}$ to indicate the fact that generalized distributions need not sum to one.

Rényi Entropy. Throughout the paper, we will employ the concept of Rényi entropy [18], a family of mea-

sures for the uncertainty in a distribution. The entire family of Rényi entropy measures is parametrized by a single positive scalar α. Formally, the Rényi entropy of a generalized distribution $\mathcal{Q}^{\mathbf{y}}_{\mathbf{x}}$ is given by

$$H_\alpha(\mathcal{Q}^{\mathbf{y}}_{\mathbf{x}};\mathbf{w}) = \frac{1}{1-\alpha}\log\left(\frac{\sum_{\mathbf{h}}\Pr(\mathbf{y},\mathbf{h}|\mathbf{x};\mathbf{w})^\alpha}{\sum_{\mathbf{h}}\Pr(\mathbf{y},\mathbf{h}|\mathbf{x};\mathbf{w})}\right). \quad (1)$$

Some interesting special cases of Rényi entropy include the well-known Shannon entropy (corresponding to taking the limit $\alpha \to 1$) and the minimum entropy (corresponding to taking the limit $\alpha \to \infty$),

$$\begin{aligned} H_1(\mathcal{Q}^{\mathbf{y}}_{\mathbf{x}};\mathbf{w}) &= \frac{-\sum_{\mathbf{h}}\Pr(\mathbf{y},\mathbf{h}|\mathbf{x};\mathbf{w})\log\Pr(\mathbf{y},\mathbf{h}|\mathbf{x};\mathbf{w})}{\sum_{\mathbf{h}}\Pr(\mathbf{y},\mathbf{h}|\mathbf{x};\mathbf{w})}, \\ H_\infty(\mathcal{Q}^{\mathbf{y}}_{\mathbf{x}};\mathbf{w}) &= -\log\max_{\mathbf{h}}\Pr(\mathbf{y},\mathbf{h}|\mathbf{x};\mathbf{w}). \end{aligned} \quad (2)$$

The Rényi entropy family is *complete* in that no other function can satisfy all the postulates of an uncertainty measure. We refer the reader to [18] for details.

4 M3E Models

We wish to develop an LVM such that, given an input $\mathbf{x}$, the best output $\mathbf{y}^*$ is predicted by optimizing an appropriate measure such that (i) $\mathbf{y}^*$ has a high probability; and (ii) $\mathbf{y}^*$ minimizes the confusion in the values of the hidden variables. Using this LVM will not only allow us to accurately predict the output (for example, whether the image contains a 'car' or not) but also the hidden variables (the location of the car in the image) with high certainty, which is important in many applications. The key observation of this work is that the readily available Rényi entropy of generalized distributions is just such a measure. Specifically, it can be verified that for any output $\mathbf{y}$, the following holds true:

$$H_\alpha(\mathcal{Q}^{\mathbf{y}}_{\mathbf{x}};\mathbf{w}) = -\log\Pr(\mathbf{y}|\mathbf{x};\mathbf{w}) + H_\alpha(\mathcal{P}^{\mathbf{y}}_{\mathbf{x}};\mathbf{w}). \quad (3)$$

In other words, the Rényi entropy of the generalized distribution of an output $\mathbf{y}$ is the sum of the negative log-likelihood of $\mathbf{y}$ (corresponding to point (i)) and the Rényi entropy of the normalized conditional probability of the hidden variables given $\mathbf{y}$ (corresponding to point (ii)). We now provide a formal description of the family of LVMs, which we refer to as the max-margin min-entropy (M3E) models, that uses Rényi entropy for prediction.

Given an input $\mathbf{x} \in \mathcal{X}$, an M3E model defines a conditional distribution over all possible outputs $\mathbf{y} \in \mathcal{Y}$ and hidden variables $\mathbf{h} \in \mathcal{H}$. For simplicity of the description, and computational tractability of the corresponding learning and inference algorithms, we focus on log-linear models. Specifically, for a given set of parameters $\mathbf{w}$, the distribution is given by

$$\Pr(\mathbf{y},\mathbf{h}|\mathbf{x};\mathbf{w}) = \frac{1}{Z(\mathbf{x};\mathbf{w})}\exp\left(\mathbf{w}^\top\Psi(\mathbf{x},\mathbf{y},\mathbf{h})\right), \quad (4)$$

where $\Psi(\mathbf{x},\mathbf{y},\mathbf{h})$ refers to the joint feature vector of the input, output and hidden variables, and $Z(\mathbf{x};\mathbf{w})$ is the partition function that normalizes the distribution to sum to one. Given an input $\mathbf{x}$, the corresponding output is predicted by minimizing the Rényi entropy of the corresponding generalized distribution, that is,

$$\mathbf{y}^* = \operatorname*{argmin}_{\mathbf{y}} H_\alpha(\mathcal{Q}^{\mathbf{y}}_{\mathbf{x}};\mathbf{w}). \quad (5)$$

5 Learning M3E Models

Given a training dataset $\mathcal{D} = \{(\mathbf{x}_i,\mathbf{y}_i), i = 1,\cdots,n\}$, we would like to learn the parameters $\mathbf{w}$ of an M3E model such that it predicts the correct output of a given instance. To this end, we propose a parameter estimation approach that tries to introduce a margin between the Rényi entropy of the ground-truth output and all other outputs. The desired margin is specified by a user-defined loss function $\Delta(\mathbf{y}_i,\overline{\mathbf{y}})$ that measures the *difference* between the two outputs $\overline{\mathbf{y}}$ and $\mathbf{y}_i$. Similar to previous max-margin formulations, we assume that $\Delta(\mathbf{y},\mathbf{y}) = 0$ for all $\mathbf{y} \in \mathcal{Y}$.

Formally, our parameter estimation approach is specified by the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w},\xi\geq 0} \quad & \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i \\ & H_\alpha(\mathcal{Q}^{\overline{\mathbf{y}}}_i;\mathbf{w}) - H_\alpha(\mathcal{Q}^{\mathbf{y}_i}_i;\mathbf{w}) \geq \Delta(\mathbf{y}_i,\overline{\mathbf{y}}) - \xi_i, \\ & \forall\overline{\mathbf{y}} \neq \mathbf{y}_i, \forall(\mathbf{x}_i,\mathbf{y}_i) \in \mathcal{D}, \end{aligned} \quad (6)$$

where we use $\mathcal{Q}^{\mathbf{y}}_i$ instead of $\mathcal{Q}^{\mathbf{y}}_{\mathbf{x}_i}$ for conciseness. The objective function of the above problem consists of two terms. The first term corresponds to regularizing the parameters by minimizing its ℓ_2 norm. The second term encourages the Rényi entropy for the ground-truth output to be smaller than the Rényi entropy of all other outputs by the desired margin. As can be seen from the constraints of the above problem, the greater the difference between $\mathbf{y}_i$ and $\overline{\mathbf{y}}$ (as specified by the loss function $\Delta(\cdot,\cdot)$), the more the desired margin. The fixed term $C > 0$ is the relative weight of these two terms.

Problem (6) can also be seen as minimizing a regularized upper bound on the user-defined loss $\Delta(\mathbf{y}_i,\mathbf{y}_i(\mathbf{w}))$ over the training dataset, where $\mathbf{y}_i(\mathbf{w})$ denotes the predicted output of the i^{th} training sample using the parameters $\mathbf{w}$. More precisely, the following proposition holds true.

Proposition 1. $\Delta(\mathbf{y}_i,\mathbf{y}_i(\mathbf{w})) \leq \xi_i$, where ξ_i are as defined in problem (6).

Proof. Since $\mathbf{y}_i(\mathbf{w})$ is the predicted output using the M3E model, $\mathbf{y}_i(\mathbf{w}) = \operatorname{argmin}_{\hat{\mathbf{y}}} H_\alpha(\mathcal{Q}^{\hat{\mathbf{y}}}_i;\mathbf{w})$. Using this

observation, we obtain the following:

$$\begin{aligned} & \Delta(\mathbf{y}_i, \mathbf{y}_i(\mathbf{w})) - H_\alpha(\mathcal{Q}_i^{\mathbf{y}_i}; \mathbf{w}) \\ \leq \quad & \Delta(\mathbf{y}_i, \mathbf{y}_i(\mathbf{w})) - H_\alpha(\mathcal{Q}_i^{\mathbf{y}_i(\mathbf{w})}; \mathbf{w}) \\ \leq \quad & \max_{\hat{\mathbf{y}}} \Big(\Delta(\mathbf{y}_i, \hat{\mathbf{y}}) - H_\alpha(\mathcal{Q}_i^{\hat{\mathbf{y}}}; \mathbf{w}) \Big) \\ = \quad & \xi_i - H_\alpha(\mathcal{Q}_i^{\mathbf{y}_i}; \mathbf{w}). \end{aligned} \tag{7}$$

Canceling the common term $H_\alpha(\mathcal{Q}_i^{\mathbf{y}_i}; \mathbf{w})$ in the first and last expressions of the above inequalities proves the proposition. ∎

The above proposition raises the question of the relationship between M3E models and the recently proposed latent SVM formulation [5, 25], which was also shown to minimize an upper bound on the loss function [25]. Our next proposition provides an answer to this question by showing that the M3E model corresponding to the minimum entropy (that is, $\alpha \to \infty$) is equivalent to latent SVM.

Proposition 2. When $\alpha = \infty$, problem (6) is equivalent to latent SVM.

The proof is omitted since it follows simply by substituting the minimum entropy H_∞ (see equation (2)) in problem (6).

6 Optimization for Learning M3E Models

While problem (6) is not convex, it has a *tractable* form that allows us to obtain an accurate set of parameters. Specifically, the following proposition holds true.

Proposition 3. Problem (6) is a difference-of-convex program for all values of $\alpha \neq 1$.

Proof Sketch. The objective function of problem (6) is clearly convex is $\mathbf{w}$ and slack variables ξ_i. The non-convexity arises due to the constraints. Specifically, the constraints can be simplified as

$$\begin{aligned} & \frac{1}{1-\alpha} \log \sum_{\mathbf{h}} \exp(\alpha \mathbf{w}^\top \Psi(\mathbf{x}_i, \overline{\mathbf{y}}, \mathbf{h})) \\ - \quad & \frac{1}{1-\alpha} \log \sum_{\mathbf{h}} \exp(\mathbf{w}^\top \Psi(\mathbf{x}_i, \overline{\mathbf{y}}, \mathbf{h})) \\ - \quad & \frac{1}{1-\alpha} \log \sum_{\mathbf{h}} \exp(\alpha \mathbf{w}^\top \Psi(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h})) \\ + \quad & \frac{1}{1-\alpha} \log \sum_{\mathbf{h}} \exp(\mathbf{w}^\top \Psi(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h})) \\ \geq \quad & \Delta(\mathbf{y}_i, \overline{\mathbf{y}}) - \xi_i. \end{aligned} \tag{8}$$

Since each term in the LHS of the above constraint has the so-called *log-sum-of-exponentials* form that is known to be convex, it follows that problem (6) is a difference-of-convex program. In other words, each of its constraints can be written in the form $f_i(\mathbf{w}) - g_i(\mathbf{w}) \leq 0$, where both $f_i(\mathbf{w})$ and $g_i(\mathbf{w})$ are convex. ∎

An approximate solution to difference-of-convex programs can be obtained using the concave-convex procedure (CCCP) [26]. Briefly, starting with an initial estimate $\mathbf{w}_0$, CCCP approximates the convex function $g_i(\mathbf{w})$ using a linear function $g_i'(\mathbf{w})$ whose slope is defined by the tangent of $g_i(\mathbf{w})$ at the current estimate $\mathbf{w}_t$. Replacing $g_i(\mathbf{w})$ by $g_i'(\mathbf{w})$ in the constraints results in a convex program, which is solved optimally to obtain a new estimate $\mathbf{w}_{t+1}$. The entire process is repeated until the objective function of the problem cannot be reduced below a user-specified tolerance.

The CCCP algorithm is guaranteed to provide a saddle point or local minimum solution to problem (6) [21]. However, it requires solving a series of optimization problems whose constraints are in the log-sum-of-exponentials form. While these constraints are convex, and the resulting problem can be solved in polynomial time, the typical runtime of the standard solvers is prohibitively large for real world applications. In § 6.2 we propose a novel trust region style algorithm that provides an approximate solution to problem (6) by solving a series of structured SVM problems. However, we begin by describing an important exception, corresponding to the minimum entropy (that is, $\alpha \to \infty$), where the CCCP algorithm itself reduces to a series of structured SVM problems.

6.1 Learning with the Minimum Entropy

Algorithm 1 *The* CCCP *algorithm for parameter estimation of the minimum entropy* M3E *model.*

input $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \cdots, (\mathbf{x}_n, \mathbf{y}_n)\}$, $\mathbf{w}_0$, ϵ.

1: $t \leftarrow 0$

2: **repeat**

3: Update $\mathbf{h}_i^* = \text{argmax}_{h_i \in \mathcal{H}} \mathbf{w}_t^\top \Phi(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}_i)$.

4: Update $\mathbf{w}_{t+1}$ by fixing the hidden variables to $\mathbf{h}_i^*$ and solving the following convex problem:

$$\begin{aligned} \min_{\mathbf{w}, \xi_i \geq 0} \quad & \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{n} \sum_i \xi_i, \\ & \mathbf{w}^\top (\Psi(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}_i^*) - \Psi(\mathbf{x}_i, \overline{\mathbf{y}}, \overline{\mathbf{h}})) \\ & \geq \Delta(\mathbf{y}_i, \overline{\mathbf{y}}) - \xi_i, \\ & \forall \overline{\mathbf{y}} \neq \mathbf{y}_i, \forall \mathbf{h} \in \mathcal{H}, \forall (\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}. \end{aligned} \tag{9}$$

5: $t \leftarrow t + 1$.

6: **until** Objective function cannot be decreased below tolerance ϵ.

While Proposition 2 demonstrates that the minimum entropy M3E model and the latent SVM are equivalent,

there is a subtle but important difference in their respective optimization using CCCP. Consider the CCCP algorithm for the minimum entropy M3E model, described in Algorithm 1. The M3E model specifies a margin between the ground-truth output $\mathbf{y}_i$ and all other *incorrect* outputs $\overline{\mathbf{y}} \neq \mathbf{y}_i$. In order to ensure that the problem defines a valid upper bound, it constrains the slack variables to be non-negative, that is, $\xi_i \geq 0$. During CCCP, this results in a succession of the convex optimization problems (9). In contrast, latent SVM simply specifies a margin between the ground-truth output and all outputs including the ground-truth (which ensures $\xi_i \geq 0$ since $\Delta(\mathbf{y}_i, \mathbf{y}_i) = 0$) [25]. During CCCP, this results in the following additional set of constraints:

$$\begin{aligned}&\mathbf{w}^\top(\Phi(\mathbf{x}_i, \mathbf{y}_i, \mathbf{h}_i^*) - \Phi(\mathbf{x}_i, \mathbf{y}_i, \overline{\mathbf{h}})) \geq -\xi_i,\ \xi_i \geq 0,\\ &\forall \overline{\mathbf{h}} \in \mathcal{H}, \forall (\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}. \end{aligned} \tag{10}$$

The additional constraints of latent SVM encourage the most likely estimates of the hidden variables to remain unchanged during the parameter update step (step 4), since they try to maximize the margin between the log probability of $(\mathbf{y}_i, \mathbf{h}_i^*)$ and the log probabilities of $(\mathbf{y}_i, \overline{\mathbf{h}})$. Intuitively, this is a bad idea since it could make the algorithm converge earlier than desired. In our experiments we show that the minimum entropy M3E model provides better results than latent SVM.

6.2 Learning with General Entropies

As mentioned earlier, when $\alpha \neq \infty$, the CCCP algorithm requires us to solve a series of convex problem whose constraints contain terms in the log-sum-of-exponentials form. This limits the ability of CCCP for learning the parameters of a general M3E model using large datasets. To make M3E practically useful, we propose a novel optimization approach for problem (6), which is outlined in Algorithm 2. Our approach consists of two main steps: (i) linearization (step 3); and (ii) parameter update (step 4). During linearization, we obtain an approximation of the Rényi entropy for a general α using a first-order Taylor's series expansion around the current parameter estimate $\mathbf{w}_t$. This approximation, denoted by $H'_\alpha(\cdot; \mathbf{w})$, is a linear function in $\mathbf{w}$. Hence, the parameter update step reduces to solving the structured SVM problem (13). Since linearization provides a good approximation for the Rényi entropy near $\mathbf{w}_t$, but a poor approximation far from $\mathbf{w}_t$, we restrict the update step to search for new parameters only around $\mathbf{w}_t$ (analogous to defining a trust region for non-convex problems [2]) by specifying the constraint $||\mathbf{w} - \mathbf{w}_t||^2 \leq \mu$. It is worth noting that this constraint can be easily incorporated into any standard structured SVM solver [13, 20, 23, 24], which makes Algorithm 2 computationally tractable.

Algorithm 2 *The algorithm for parameter estimation of the* M3E *model with general* α.

input $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \cdots, (\mathbf{x}_n, \mathbf{y}_n)\}$, $\mathbf{w}_0$, ϵ.

1: $t \leftarrow 0$

2: **repeat**

3: For each input $(\mathbf{x}_i)$ and output $\mathbf{y} \in \mathcal{Y}$, compute the following terms

$$\begin{aligned} G_\alpha(\mathcal{Q}_i^{\mathbf{y}}; \mathbf{w}_t) &= \nabla_{\mathbf{w}} H_\alpha(\mathbf{Q}_i^{\mathbf{y}}; \mathbf{w})|_{\mathbf{w}_t}, \\ C_\alpha(\mathcal{Q}_i^{\mathbf{y}}; \mathbf{w}_t) &= H_\alpha(\mathbf{Q}_i^{\mathbf{y}}; \mathbf{w}_t) - \mathbf{w}_t^\top G_\alpha(\mathbf{Q}_i^{\mathbf{y}}; \mathbf{w}_t). \end{aligned} \tag{11}$$

The above terms can be used to approximate the Rényi entropy $H_\alpha(\mathcal{Q}_i^{\mathbf{y}}; \mathbf{w})$ using the first-order Taylor's series approximation as

$$\begin{aligned} H_\alpha(\mathbf{Q}_i^{\mathbf{y}}; \mathbf{w}) &\approx H'_\alpha(\mathbf{Q}_i^{\mathbf{y}}; \mathbf{w}) \\ &= \mathbf{w}^\top G_\alpha(\mathbf{Q}_i^{\mathbf{y}}; \mathbf{w}_t) + C_\alpha(\mathbf{Q}_i^{\mathbf{y}}; \mathbf{w}_t) \end{aligned} \tag{12}$$

4: Update $\mathbf{w}_{t+1}$ by solving the following convex problem:

$$\begin{aligned} \min_{\mathbf{w}, \xi_i \geq 0} \quad & \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{n}\sum_i \xi_i, \\ & H'_\alpha(\mathcal{Q}_i^{\overline{\mathbf{y}}}; \mathbf{w}) - H'_\alpha(\mathcal{Q}_i^{\mathbf{y}_i}; \mathbf{w}) \\ & \geq \Delta(\mathbf{y}_i, \overline{\mathbf{y}}) - \xi_i, \\ & \forall \overline{\mathbf{y}} \neq \mathbf{y}_i, \forall (\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}, \\ & ||\mathbf{w} - \mathbf{w}_t||^2 \leq \mu. \end{aligned} \tag{13}$$

The term μ specifies a trust region where $H'_\alpha(\cdot)$ accurately approximates $H'_\alpha(\cdot)$.

5: $t \leftarrow t + 1$.

6: **until** Objective function cannot be decreased below tolerance ϵ.

The parameter μ governs the size of the trust region, and therefore, influences the trade-off between the speed and the accuracy of our algorithm. Specifically, a large μ will allow us to search over a large space thereby increasing the speed, but may converge to an inaccurate solution due to the poor approximation provided by the linearization step over the entire trust region. A small μ will restrict us to a region where the approximation provided by the linearization is accurate, but will slow down the algorithm. In practice, we found that the following simple strategy provided a desirable trade-off. We start with a large value $\mu = \mu_{max}$, obtain the solution $\mathbf{w}'$ and compute the objective of problem (6). If the objective function has decreased above the tolerance ϵ since the previous iteration, then we set $\mathbf{w}_{t+1} = \mathbf{w}'$. Otherwise, we anneal $\mu \leftarrow \mu/\lambda$ and solve problem (13) to obtain a new $\mathbf{w}'$. Algorithm 2 is said to converge when the difference in the objective of problem (6) computed at $\mathbf{w}_{t+1}$ and $\mathbf{w}_t$ is below tolerance ϵ.

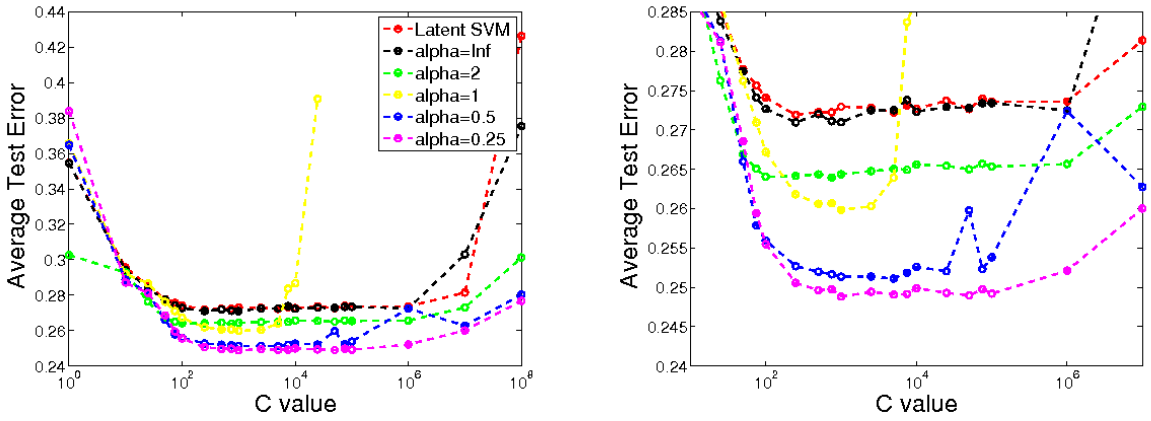

Figure 1: *The average (over all proteins and folds) test errors for the motif finding experiment across varying values of C and α. Left: All values of C and α that were used in our experiments. Right: Zoomed-in version to highlight the difference in performance among the various methods. For each (protein,fold) pair, the model with the best train error out of 4 random initializations was chosen. Further results are provided in Table 1. As can be seen, lower values of α achieve the best test errors, and larger values of α approach the performance of latent* SVM*, which solves the same problem as an* M3E *with $\alpha = \infty$ with a slightly different optimization procedure. Note that the results for $\alpha = 1$ become unstable for larger values of C due to numerical instability during parameter estimation. Best viewed in color.*

7 Experiments

We now demonstrate the efficacy of M3E models using two standard machine learning applications that have previously been addressed using the latent SVM formulation: motif finding and image classification [14, 25]. Specifically, we show how the more general M3E formulation can be used to significantly improve the results compared to latent SVM.

To help other researchers use M3E models in their work, we will make all the code necessarily to replicate our experiments available online. All the datasets used in our experiments are publicly available. As these datasets were previously used in [14], we borrow heavily from their text to describe the experimental setup.

7.1 Motif Finding

Problem Formulation. We consider the problem of binary classification of DNA sequences. Specifically, the input vector $\mathbf{x}$ consists of a DNA sequence of length l (where each element of the sequence is a nucleotide of type A, G, T or C) and the output space $\mathcal{Y} = \{0, 1\}$. In our experiments, the classes correspond to two different types of genes: those that bind to a protein of interest with high affinity and those that do not. The positive sequences are assumed to contain particular patterns, called *motifs*, of length m that are believed to be useful for classification. However, the starting position of the motif within a gene sequence is often not known. Hence, this position is treated as the hidden variable $\mathbf{h}$. Given an input $\mathbf{x}$, an output $\mathbf{y}$ and a hidden variable $\mathbf{h}$, we use the joint feature vector suggested by [25]. The loss function Δ is the standard 0-1 classification loss. The number of possible values of the hidden variables is small (of the order of the size of the DNA sequence), which makes this problem tractable within the M3E formulation without having to resort to approximate inference schemes.

Dataset. We use the publicly available UniProbe dataset [1] that provides positive and negative DNA sequences for 177 proteins. For this work, we chose five proteins at random. The total number of sequences per protein is roughly $40,000$. For all the sequences, the motif length m is known. In order to specify a classification task for a particular protein, we randomly split the sequences into roughly 50% for training and 50% for testing. We report results using 5 folds.

Results. Figure 1 shows the test errors for latent SVM and various M3E models across different values of C. The values are averaged over all 25 (protein, fold) pairs. For each protein and each fold, we initialize the methods using four different random seeds, and report the test error corresponding to the seed with the best training error (with ties broken by training objective value). As the results indicate, using high values of α provides similar results to latent SVM. Recall that while the objective for an M3E model with $\alpha = \infty$ is equivalent to that of latent SVM, the optimizations for each are different, thereby yielding different results. The M3E models with low values of α achieve significantly better performance than latent SVM, indicating that these values are more suitable for predicting whether a DNA sequence has a high affinity towards binding to a particular protein. Table 1 shows the average test error for the best C and α values. The best

M3E model achieves 2.2% lower test error than the best latent SVM model. The improvements are statistically significant for each of the 5 proteins using a paired t-test, with a maximum p-value of 3.0e-4.

	Latent SVM	M3E
Protein 052	$C = 5000$	$C = 7500, \alpha = 0.25$
Train Error	28.6%	**26.9%**
Test Error	29.2%	**27.4%**
Protein 074	$C = 5000$	$C = 10000, \alpha = 0.25$
Train Error	26.7%	**23.6%**
Test Error	27.6%	**24.2%**
Protein 108	$C = 500$	$C = 10000, \alpha = 0.25$
Train Error	26.8%	**25.0%**
Test Error	27.1%	**25.3%**
Protein 131	$C = 750$	$C = 750, \alpha = 0.25$
Train Error	28.8%	**27.3%**
Test Error	29.2%	**27.6%**
Protein 146	$C = 1000$	$C = 5000, \alpha = 0.25$
Train Error	22.2%	**19.9%**
Test Error	22.5%	**20.1%**
Average		
Train Error	26.6%	**24.5%**
Test Error	27.1%	**24.9%**

Table 1: *Average training and test errors for 5 randomly chosen proteins, split into 5 random folds. For each protein, the parameters that achieved the best mean training error across folds were chosen, and those parameters are shown. The* M3E *models outperform latent* SVM *on each protein, and overall yield an improvement of over* 2% *in terms of both the training error and the test error.*

7.2 Image Classification

Problem Formulation. Given a set of images along with labels that indicate the presence of a particular object category in the image (for example, a mammal), our goal is to learn discriminative object models. Specifically, we consider two types of problems: (i) given an image containing an instance of an object category from a fixed set of c categories, predict the correct category (that is, a multi-class classification problem, where the set of outputs $\mathcal{Y} = \{0, 1, \cdots, c-1\}$); (ii) given an image, predict whether it contains an instance of an object category of interest or not (that is, a binary classification problem, where $\mathcal{Y} = \{0, 1\}$). In practice, although it is easy to mine such images from free photo-sharing websites such as Flickr, it is burdensome to obtain ground truth annotations of the exact location of the object in each image. To avoid requiring these human annotations, we model the location of objects as hidden variables. Formally, for a given image $\mathbf{x}$, label $\mathbf{y} \in \mathcal{Y}$ and location $\mathbf{h}$, the score is modelled as $\mathbf{w}^\top \Phi(\mathbf{x}, \mathbf{y}, \mathbf{h}) = \mathbf{w}_\mathbf{y}^\top \Phi_\mathbf{h}(\mathbf{x})$, where $\mathbf{w}_\mathbf{y}$ are the parameters that corresponds to the label $\mathbf{y}$ and $\Phi_\mathbf{h}(\cdot)$ is the HOG [3, 5] feature extracted from the image at position $\mathbf{h}$ (the size of the object is assumed to be the same for all images—a reasonable assumption for our datasets). The number of possible values of the hidden variables is of the order of the number of pixels in an image, which makes M3E learning tractable without resorting to approximate inference. For both the settings (multi-class classification and binary classification), the loss function $\Delta(\mathbf{y}, \hat{\mathbf{y}})$ is the standard 0-1 classification loss.

Dataset. We use images of 6 different mammals (approximately 45 images per mammal) that have been previously employed for object localization [8, 14]. We split the images of each category into approximately 90% for training and 10% for testing. We report results for 5 such randomized folds.

Results. As in the motif finding application, we initialize each method using four different random seeds, and report the test error corresponding to the seed with the best training error (with ties broken by training objective value). Fig. 3 shows the results of the multi-class classification setting, averaged over all 5 folds. As can be seen, latent SVM performs poorly compared to the M3E models. All M3E models with $\alpha \geq 1000.0$ (including $\alpha = \infty$) provide the best test error of 12.3%. Fig. 2 shows the average (over 5 folds) test errors for all 6 binary classification problems. For the "llama" class, M3E models achieve the same performance as latent SVM. For the "rhino" class, similar to the multi-class classification setting, $\alpha = \infty$ provides the best results. For the other four classes ("bison", "deer", "elephant", and "giraffe"), the best performing M3E models use a smaller value of α (between 2.0 and 8.0). This illustrates the importance of selecting the *right* value of α for the problem at hand, instead of relying solely on the minimum entropy, as is the case with latent SVM. Overall, the average test classification errors across all six mammals are 5.7% for latent SVM, 5.4% for the minimum entropy M3E model, and 4.2% for the best M3E model. The improvements of M3E over latent SVM are statistically significant in 4 of the 6 classes over all random seeds as well as for the "elephant" class when only the best seed value is used.

8 Discussion

We presented a new family of LVMs called M3E models that predict the output of a given input as the one that results in the minimum Rényi entropy of the corresponding generalized distribution. In the M3E framework the predicted output (i) has a high probability and (ii) minimizes the uncertainty in the hidden variables. We showed how the parameters of an M3E model are learned using a max-margin formulation can that be viewed as minimizing an upper bound on a user-defined loss. Latent SVM is a special case in our family of models. Empirically, we demonstrated that the more general M3E models can outperform latent SVM.

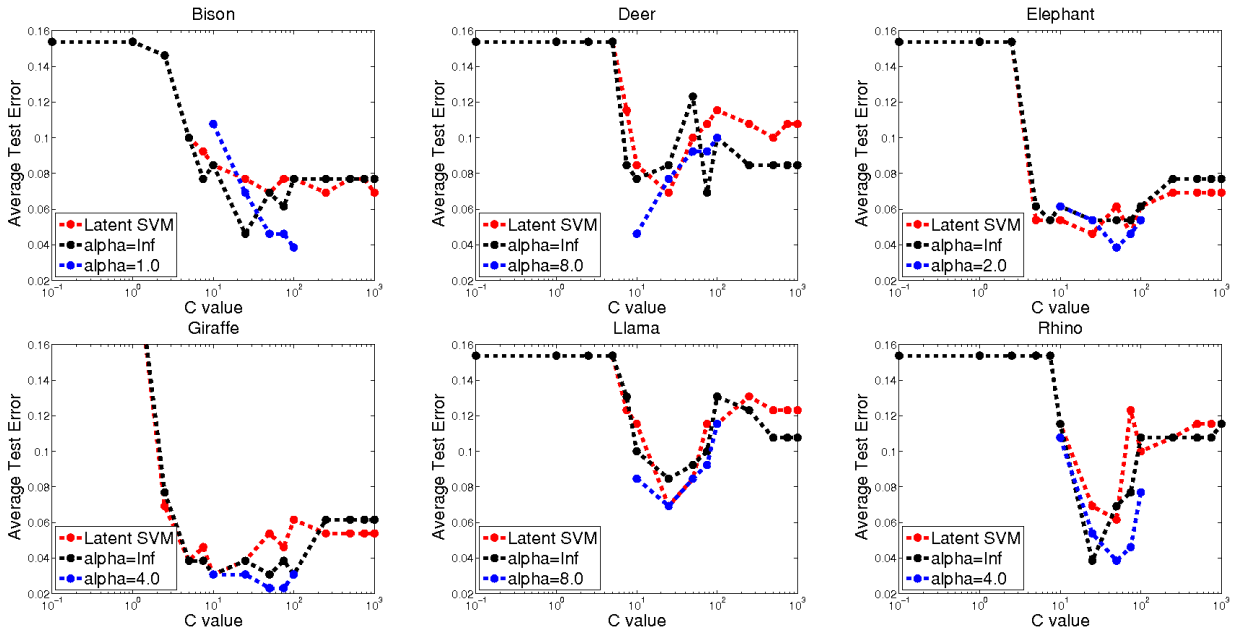

Figure 2: *Image classification test errors for all six mammal classes. Each number is averaged across 5 random folds; in each fold, the model with the best training error out of 4 random initializations was chosen. For each mammal, the* M3E *model that achieved the best test error is shown with the corresponding* α *value indicated, along with latent* SVM *and the minimum entropy* M3E *model (*$\alpha = \infty$*). The average test classification errors across all six mammals are* 5.7% *for latent* SVM*,* 5.4% *for the minimum entropy* M3E *model, and* 4.2% *for the best* M3E *model. Best viewed in color.*

Similar to other LVMs, when the latent variable space is small, or when the underlying distribution is tractable (for example, a small tree-width distribution), the parameters of an M3E model can be learned accurately. Specifically, in this case, parameter learning is equivalent to solving a difference-of-convex optimization problem using CCCP [26] or other recently proposed algorithms [14]. When the latent variables lie in an exponentially large space, M3E can lend itself to approximate optimization. For example, we could design an appropriate variational inference procedure that best approximates a Rényi entropy of interest. This offers an interesting direction for future work.

The introduction of M3E models yields several interesting questions. For example, is it possible to determine the *best* value of α for a type of hidden variable? Given a problem that requires different types of hidden variables (say, learning an image segmentation model using partially segmented images, bounding box annotations, image-level labels), should we employ different α values for them? Can these α values be learned? Answers to these questions would not only be of great practical importance, but would also reveal interesting theoretical properties of the M3E family of models.

Finally, we note that while the method described in this paper employs Rényi entropy, other forms of entropy, such as the generalized Rényi entropy [15], the Havrada-Charvat entropy [7] or Rao's quadratic entropy [17], are also readily applicable within our max-margin learning framework. In addition, the generalized Rényi entropy can be easily optimized using our trust region style algorithm. Designing efficient optimization techniques for learning M3E models with other entropies remains an open challenge.

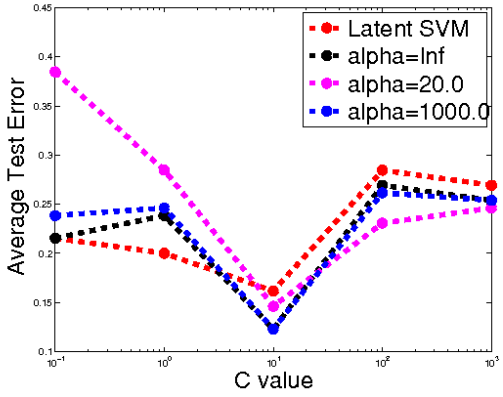

Figure 3: *Test errors for the multi-class classification setting. Latent* SVM *performs poorly compared to the* M3E *models, which attain the best test error of* 12.3% *for all* $\alpha \geq 1000.0$ *in our experiments.*

Acknowledgements. This work is supported by NSF under grant IIS 0917151, MURI contract N000140710747, and the Boeing Corporation.

References

[1] M. Berger, G. Badis, A. Gehrke, and S. Talukder et al. Variation in homeodomain DNA binding revealed by high-resolution analysis of sequence preferences. *Cell*, 2008.

[2] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999.

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[4] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society*, 1977.

[5] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.

[6] A. Gelman, J. Carlin, H. Stern, and D. Rubin. *Bayesian Data Analysis*. Chapman and Hall, 1995.

[7] J. Havrada and F. Charvat. Quantification method in classification processes: Concept of structural α-entropy. *Kybernetika*, 1967.

[8] G. Heitz, G. Elidan, B. Packer, and D. Koller. Shape-based object localization for descriptive classification. *IJCV*, 2009.

[9] T. Jaakkola, M. Meila, and T. Jebara. Maximum entropy discrimination. In *NIPS*, 1999.

[10] E. Jaynes. *Probability theory: The logic of science*. Cambridge University Press, 2003.

[11] T. Jebara. *Discriminative, generative and imitative learning*. PhD thesis, MIT, 2001.

[12] T. Jebara and T. Jaakkola. Feature selection and dualities in maximum entropy discrimination. In *UAI*, 2000.

[13] T. Joachims, T. Finley, and C.-N. Yu. Cutting-plane training for structural SVMs. *Machine Learning*, 2009.

[14] M. P. Kumar, B. Packer, and D. Koller. Self-paced learning for latent variable models. In *NIPS*, 2010.

[15] A. Mathai and P. Rathie. *Basic Concepts in Information Theory and Statistics*. Wiley (Halsted Press), New York, 1974.

[16] R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M. Jordan, editor, *Learning in Graphical Models*. MIT Press, 1999.

[17] C. Rao. Diversity and dissimilarity coefficients: A unified approach. *Theoretical Population Biology*, 1982.

[18] A. Renyi. On measures of information and entropy. In *Berkeley Symposium on Mathematics, Statistics and Probability*, 1961.

[19] J. Salojarvi, K. Puolamaki, and S. Kaski. Expectation maximization algorithms for conditional likelihoods. In *ICML*, 2005.

[20] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for SVM. In *ICML*, 2009.

[21] B. Sriperumbudur and G. Lanckriet. On the convergence of concave-convex procedure. In *NIPS Workshop on Optimization for Machine Learning*, 2009.

[22] R. Sundberg. Maximum likelihood theory for incomplete data from an exponential family. *Scandinavian Journal of Statistics*, 1974.

[23] B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *NIPS*, 2003.

[24] I. Tsochantaridis, T. Hofmann, Y. Altun, and T. Joachims. Support vector machine learning for interdependent and structured output spaces. In *ICML*, 2004.

[25] C.-N. Yu and T. Joachims. Learning structural SVMs with latent variables. In *ICML*, 2009.

[26] A. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Computation*, 2003.