Hidden Markov Support Vector Machines

Yasemin Altun ALTUN@CS.BROWN.EDU
Ioannis Tsochantaridis IT@CS.BROWN.EDU
Thomas Hofmann TH@CS.BROWN.EDU
Department of Computer Science, Brown University, Providence, RI 02912 USA

Abstract

This paper presents a novel discriminative learning technique for label sequences based on a combination of two of the most successful learning algorithms, Support Vector Machines and Hidden Markov Models which we call Hidden Markov Support Vector Machine (HM-SVM). The proposed architecture handles dependencies between neighboring labels using highly efficient Viterbi decoding. In contrast to standard HMM training, however, the learning procedure is discriminative and is based on a maximum/soft margin criterion. Compared to previous methods like Conditional Random Fields, Maximum Entropy Markov Models and label sequence boosting, HM-SVMs have a number of advantages. Most notably, it is possible to learn non-linear discriminant functions via kernel functions. At the same time HM-SVMs share the key advantages with other discriminative methods, in particular the capability to deal with overlapping features. Application areas for the presented technique range from natural language processing and information extraction to computational biology. We report experimental evaluations on three different tasks: named entity recognition, part-of-speech tagging, and protein secondary structure prediction. The results demonstrate the competitiveness of the proposed approach.

1. Introduction

Learning from observation sequences is a fundamental problem in machine learning. One facet of the problem generalizes supervised classification by predicting label sequences instead of individual class labels. The latter is also known as *label sequence learning*. It subsumes problems like segmenting observation sequences, annotating observation sequences, and recovering underlying discrete sources. The potential applications are widespread, ranging from natural language processing and speech recognition to computational biology and system identification.

Up to now, the predominant formalism for modeling and predicting label sequences has been based on Hidden Markov Models (HMMs) and variations thereof. HMMs model sequential dependencies by treating the label sequence as a Markov chain. This avoids direct dependencies between subsequent observations and leads to an efficient dynamic programming formulation for inference and learning. Yet, despite their success, HMMs have at least three major limitations. (i) They are typically trained in a non-discriminative manner. (ii) The conditional independence assumptions are often too restrictive. And (iii) they are based on explicit feature representations and lack the power of kernel-based methods.

In this paper, we propose an architecture for learning label sequences which combines HMMs with Support Vector Machines (SVMs) in an innovative way. This novel architecture is called Hidden Markov SVM (HM-SVM). HM-SVMs address all of the above shortcomings, while retaining some of the key advantages of HMMs, namely the Markov chain dependency structure between labels and an efficient dynamic programming formulation. Our work continues a recent line of research that includes Maximum Entropy Markov Models (MEMMs) [7, 10], Conditional Random Fields (CRFs) [6], perceptron re-ranking [2, 3] and label sequence boosting [1]. The basic commonality between HM-SVMs and these methods is their discriminative approach to modelling and the fact that they can account for overlapping features, which means that labels can depend directly on features of past or future observations. The two crucial ingredients added by HM-SVMs are the maximum margin principle and a kernel-centric approach to learning non-linear discriminant functions, two properties inherited from SVMs.

2. Input-Output Mappings via Joint Feature Functions

Before focusing on the label learning problem, let us outline a more general framework for learning mappings to discrete output spaces of which the proposed HM-SVM method is a special case [4]. This framework subsumes a number of problems such as binary classification, multiclass classification, multi-label classification, classification with class taxonomies and last but not least, label sequence learning.

The general approach we pursue is to learn a *discriminant function* $F : \mathcal{X} \times \mathcal{Y} \to \Re$ over input/output pairs and to maximize this function over the response variable to make a prediction. Hence, the general form for f is

$$f(\mathbf{x}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} F(\mathbf{x}, \mathbf{y}) \,. \tag{1}$$

In particular, we are interested in a setting, where F is linear in some combined feature representation of inputs and outputs $\Phi(\mathbf{x}, \mathbf{y})$, i.e.

$$F(\mathbf{x}, \mathbf{y}; \mathbf{w}) = \langle \mathbf{w}, \Phi(\mathbf{x}, \mathbf{y}) \rangle \,. \tag{2}$$

Moreover, we would like to apply kernel functions to avoid performing an explicit mapping Φ when this may become intractable, thus leveraging the theory of kernel-based learning. This is possible due to the linearity of the function F, if we have a kernel K over the joint input/output space such that

$$K((\mathbf{x}, \mathbf{y}), (\bar{\mathbf{x}}, \bar{\mathbf{y}})) = \langle \Phi(\mathbf{x}, \mathbf{y}), \Phi(\bar{\mathbf{x}}, \bar{\mathbf{y}}) \rangle \tag{3}$$

and whenever the optimal function F has a dual representation in terms of an expansion $F(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m} \alpha_i K((\tilde{\mathbf{x}}_i, \tilde{\mathbf{y}}_i), (\mathbf{x}, \mathbf{y}))$ over some finite set of samples $(\tilde{\mathbf{x}}_1, \tilde{\mathbf{y}}_1), \ldots (\tilde{\mathbf{x}}_m, \tilde{\mathbf{y}}_m)$.

The key idea of this approach is to extract features not only from the input pattern as in binary classification, but jointly from input-output pairs. The compatibility of an input $\mathbf{x}$ and an output $\mathbf{y}$ may depend on a particular property of $\mathbf{x}$ in conjunction with a particular property of $\mathbf{y}$. This is especially relevant, if $\mathbf{y}$ is not simply an atomic label, but has an internal structure that can itself be described by certain features. These features in turn may interact in non-trivial ways with certain properties of the input patterns, which is the main difference between our approach and the work presented in [11].

3. Hidden Markov Chain Discriminants

Learning label sequences is a generalization of the standard supervised classification problem. Formally, the goal is to learn a mapping f from observation sequences $\mathbf{x} = (x^1, x^2, \ldots, x^t, \ldots)$ to label sequences $\mathbf{y} = (y^1, y^2, \ldots, y^t, \ldots)$, where each label takes values from some label set Σ, i.e. $y^t \in \Sigma$. Since for a given observation sequence $\mathbf{x}$, we only consider label sequences $\mathbf{y}$ of the same (fixed) length, the admissible range is effectively finite for every $\mathbf{x}$. The availability of a training set of labeled sequences $\mathcal{X} \equiv \{(\mathbf{x}_i, \mathbf{y}_i) : i = 1, \ldots, n\}$ to learn the mapping f from data is assumed.

In order to apply the above joint feature mapping framework to label sequence learning, we define the output space $\mathcal{Y}$ to consist of all possible label sequences. Inspired by HMMs, we propose to define two types of features for the feature mapping Φ in Eq. (2). The first type of features combine attributes or input features $\psi_r(x^s) \in \Re$, $r = 1, \ldots, d$ extracted from some observation pattern x^s with an indicator function for the label y^t,

$$\phi^{st}_{r\sigma} = [\![y^t = \sigma]\!] \psi_r(x^s) \,. \tag{4}$$

Here $[\![Q]\!]$ denotes the indicator function for the predicate Q.

To illustrate this point, we discuss a concrete example from part-of-speech tagging: $\psi_r(x^s)$ may denote the input feature of a specific word like 'rain' occurring in the s-th position in a sentence, while $[\![y^t = \sigma]\!]$ may encode whether or not the t-th word is a noun or not. $\phi^{st}_{r\sigma} = 1$ would then indicate the conjunction of these two predicates, a sequence for which the s-th word is 'rain' $(= r)$ and in which the t-th word has been labeled as a noun $(= \sigma)$. Notice that in general, ψ_r may not be binary, but real-valued and so may $\phi^{st}_{r\sigma}$.

The second types of features we consider deal with inter-label dependencies

$$\bar{\phi}^{st}_{\sigma\tau} = [\![y^s = \sigma \wedge y^t = \tau]\!]. \tag{5}$$

In terms of these features, a (partial) feature map $\Phi(\mathbf{x}, \mathbf{y}; t)$ at position t can be defined by selecting appropriate subsets of the features $\{\phi^{st}_{r\sigma}\}$ and $\{\bar{\phi}^{st}_{\sigma\tau}\}$. For example, an HMM only uses input-label features of the type $\phi^{tt}_{r\sigma}$ and label-label features $\bar{\phi}^{t(t+1)}_{\sigma\tau}$, reflecting the (first order) Markov property of the chain. In the case of HM-SVMs we maintain the latter restriction (although it can trivially be generalized to higher order Markov chains), but we also include features $\phi^{st}_{r\sigma}$, where $s \neq t$, for example, $s = t - 1$ or $s = t + 1$ or larger windows around t. In the simplest case, a feature map $\Phi(\mathbf{x}, \mathbf{y}; t)$ can then be specified by defining a feature representation of input patterns Ψ and by selecting an appropriate window size.[1] All the features

[1] Of course, many generalizations are possible, for ex-

extracted at location t are simply stacked together to form $\Phi(\mathbf{x}, \mathbf{y}; t)$. Finally this feature map is extended to sequences $(\mathbf{x}, \mathbf{y})$ of length T in an additive manner as follows

$$\Phi(\mathbf{x}, \mathbf{y}) = \sum_{t=1}^{T} \Phi(\mathbf{x}, \mathbf{y}; t)\,. \tag{6}$$

Notice that as long as the label-label interactions are restricted to nearest-neighbor dependencies the maximization in Eq. (1) can be carried out efficiently with the help of Viterbi decoding. We will present further details in the next section.

In order to better understand the definition of the feature mapping Φ, it is revealing to rewrite the inner product between feature vectors for different sequences,

$$\begin{aligned}\langle \Phi(\mathbf{x}, \mathbf{y}), \Phi(\bar{\mathbf{x}}, \bar{\mathbf{y}}) \rangle \quad &= \quad \sum_{s,t} [\![y^{s-1} = \bar{y}^{t-1} \wedge y^s = \bar{y}^t]\!] \\ &\quad + \sum_{s,t} [\![y^s = \bar{y}^t]\!] k(x^s, \bar{x}^t) \end{aligned} \tag{7}$$

where

$$k(x^s, \bar{x}^t) = \langle \Psi(x^s), \Psi(\bar{x}^t) \rangle\,. \tag{8}$$

Hence the similarity between two sequences depends on the number of common two-label fragments as well as the inner product between the feature representation of patterns with common label.

4. Hidden Markov Perceptron Learning

We will first focus on an on-line learning approach to label sequence learning, which generalizes perceptron learning and was first propose in the context of natural language processing in [3].

In a nutshell, this algorithm works as follows. In an on-line fashion, pattern sequences $\mathbf{x}_i$ are presented and the optimal decoding $f(\mathbf{x}_i)$ is computed. This amounts to Viterbi decoding in order to produce the most 'likely', i.e. highest scored, label sequence $\hat{\mathbf{y}}$. If the predicted label sequence is correct $\hat{\mathbf{y}} = \mathbf{y}_i$, no update is performed. Otherwise, the weight vector $\mathbf{w}$ is updated based on the difference vector $\triangle\Phi = \Phi(\mathbf{x}_i, \mathbf{y}_i) - \Phi(\mathbf{x}_i, \hat{\mathbf{y}})$, namely $\mathbf{w}^{new} \leftarrow \mathbf{w}^{old} + \triangle\Phi$.

In order to avoid an explicit evaluation of the feature map as well as a direct (i.e. primal) representation of the discriminant function, we would like to derive an equivalent dual formulation of the perceptron algorithm. Notice that in the standard perceptron learning case, $\Phi(\mathbf{x}, 1) = -\Phi(\mathbf{x}, -1)$, so that it is sufficient to store those training patterns that have been used during a weight update. In the label sequence perceptron algorithm, one also needs to store the incorrectly decoded sequence (which we call *negative pseudo-example*) $(\mathbf{x}_i, f(\mathbf{x}_i))$. More precisely, one only needs to store how the decoded $f(\mathbf{x}_i)$ differs from the correct $\mathbf{y}_i$, which typically results in a more compact representation.

The dual formulation of the discriminant function with kernel K is as follows. One maintains a set of dual parameters $\alpha_i(\mathbf{y})$ such that

$$F(\mathbf{x}, \mathbf{y}) = \sum_i \sum_{\bar{\mathbf{y}}} \alpha_i(\bar{\mathbf{y}}) K((\mathbf{x}_i, \bar{\mathbf{y}}), (\mathbf{x}, \mathbf{y}))\,. \tag{9}$$

Once an update is necessary for training sequence $(\mathbf{x}_i, \mathbf{y}_i)$ and incorrectly decoded $\hat{\mathbf{y}}$, one simply increments $\alpha_i(\mathbf{y}_i)$ and decrements $\alpha_i(\hat{\mathbf{y}})$ by one. Of course, as a practical matter of implementation one will only represent those $\alpha_i(\mathbf{y})$ which are non-zero. Notice that this requires to keep track of the α values themselves as well as the pairs $(\mathbf{x}_i, \mathbf{y})$ for which $\alpha_i(\mathbf{y}) < 0$.

The above formulation is valid for any kernel function on label sequences. In the case of nearest neighbor label interactions, one can make use of the additivity of the sequence feature map in Eq. (7) to come up with a more efficient scheme. Decomposing F into two contributions, $F(\mathbf{x}, \mathbf{y}) = F_1(\mathbf{x}, \mathbf{y}) + F_2(\mathbf{x}, \mathbf{y})$, where

$$F_1(\mathbf{x}, \mathbf{y}) = \sum_{\sigma,\tau} \rho(\sigma, \tau) \sum_s [\![y^{s-1} = \sigma \wedge y^s = \tau]\!]\,. \tag{10}$$

with

$$\rho(\sigma, \tau) = \sum_{i,\bar{\mathbf{y}}} \alpha_i(\bar{\mathbf{y}}) \sum_t [\![\bar{y}^{t-1} = \sigma \wedge \bar{y}^t = \tau]\!] \tag{11}$$

and where

$$F_2(\mathbf{x}, \mathbf{y}) \quad = \quad \sum_{s,\sigma} [\![y^s = \sigma]\!] \sum_i \sum_t \beta(i, t, \sigma) k(x^s, x_i^t)\,.$$

with

$$\beta(i, t, \sigma) = \sum_{\mathbf{y}} [\![y^t = \sigma]\!]\, \alpha_i(\mathbf{y}) \tag{12}$$

In summary, one needs to keep track of how often each label pair incorrectly appeared in a decoded sequence and how often the label of a particular observation x_i^s was incorrectly decoded. The advantage of using the representation via $\rho(\sigma, \tau)$ and $\beta(i, t, \sigma)$ is that it is independent of the number of incorrect sequences $\hat{\mathbf{y}}$ and can be updated very efficiently.

ample, one may extract different input features dependent of the relative distance $|t - s|$ in the chain.

In order to perform the Viterbi decoding, we have to compute the transition cost matrix and the observation cost matrix H_i for the i-th sequence. The latter is given by

$$H_i^{s\sigma} = \sum_j \sum_t \beta(j,t,\sigma) k(x_i^s, x_j^t) \tag{13}$$

The coefficients of the transition matrix are simply given by the values $\rho(\sigma,\tau)$.

Algorithm 1 Dual perceptron algorithm for learning via joint feature functions.

initialize all $\alpha_i(\mathbf{y}) = 0$
repeat
  for all training patterns $\mathbf{x}_i$ **do**
    compute $\hat{\mathbf{y}}_i = \arg\max_{\mathbf{y}\in\mathcal{Y}} F(\mathbf{x}_i, \mathbf{y})$, where $F(\cdot,\cdot) = \sum_j \sum_{\mathbf{y}} \alpha_j(\mathbf{y}) K((\cdot,\cdot),(\mathbf{x}_j,\mathbf{y}))$
    if $\mathbf{y}_i \neq \hat{\mathbf{y}}_i$ **then**
      $\alpha_i(\mathbf{y}_i) \leftarrow \alpha_i(\mathbf{y}_i) + 1$
      $\alpha_i(\hat{\mathbf{y}}_i) \leftarrow \alpha_i(\hat{\mathbf{y}}_i) - 1$
    end if
  end for
until no more errors

In order to prove the convergence of this algorithm, it suffices to apply [2, Theorem 1] which is a simple generalization of Novikoff's theorem.

Theorem 1 *Given a training set $(\mathbf{x}_i, \mathbf{y}_i)$, $i = 1,\ldots,n$ and for each training label a set of candidate labels $\mathcal{Y}_i \subseteq \mathcal{Y} - \{\mathbf{y}_i\}$. If there exists a weight vector $\mathbf{w}$ such that $\|\mathbf{w}\| = 1$ and*

$$\langle \mathbf{w}, \Phi(\mathbf{x}_i, \mathbf{y}_i)\rangle - \langle \mathbf{w}, \Phi(\mathbf{x}_i, \mathbf{y})\rangle \geq \gamma, \quad \text{for all } \mathbf{y} \in \mathcal{Y}_i$$

then the number of update steps performed by the above perceptron algorithm is bounded from above by $\frac{R^2}{\gamma^2}$, where $R = \max_i \|\Phi(\mathbf{x}_i, \mathbf{y})\|$ for $\mathbf{y} \in \mathcal{Y}_i \cup \{\mathbf{y}_i\}$.

5. Hidden Markov Support Vector Machine

Our goal in this section is to derive a maximum margin formulation for the joint kernel learning setting. We generalize the notion of a separation margin, by defining the margin of a training example with respect to a discriminant function, F, as:

$$\gamma_i = F(\mathbf{x}_i, \mathbf{y}_i) - \max_{\mathbf{y}\neq\mathbf{y}_i} F(\mathbf{x}_i, \mathbf{y}) \,. \tag{14}$$

Then the maximum margin problem can be defined as finding a weight vector $\mathbf{w}$ that maximizes $\min_i \gamma_i$. Obviously, like in the standard setting of maximum margin classification with binary labels, one has to either restrict the norm of $\mathbf{w}$ (e.g. $\|\mathbf{w}\| = 1$), or to fix the functional margin ($\min_i \gamma_i \geq 1$). The latter results in the following optimization problem with a quadratic objective

$$\min \quad \frac{1}{2}\|\mathbf{w}\|^2 \tag{15}$$

$$\text{s.t.} \quad F(\mathbf{x}_i, \mathbf{y}_i) - \max_{\mathbf{y}\neq\mathbf{y}_i} F(\mathbf{x}_i, \mathbf{y}) \geq 1, \ \forall i \tag{16}$$

Each non-linear constraint in Eq. (16) can be replaced by an equivalent set of linear constraints,

$$F(\mathbf{x}_i, \mathbf{y}_i) - F(\mathbf{x}_i, \mathbf{y}) \geq 1 \,, \ \forall i \text{ and } \forall \mathbf{y} \neq \mathbf{y}_i \tag{17}$$

Let us further rewrite these constraints by introducing an additional threshold θ_i for every example, which yields

$$z_i(\mathbf{y})\,(F(\mathbf{x}_i, \mathbf{y}) + \theta_i) \geq \frac{1}{2} \tag{18}$$

with the definition $z_i(\mathbf{y}) = 1$, if $\mathbf{y} = \mathbf{y}_i$ and $z_i(\mathbf{y}) = -1$, otherwise.

Proposition 1 *For any discriminant function F, F fulfills the constraints in Eq. (17) for an example $(\mathbf{x}_i, \mathbf{y}_i)$ if and only if there exists $\theta_i \in \Re$ such that F fulfills the constraints in Eq. (18).*

Proof:
If F fulfills Eq. (17) then define $\theta_i \equiv -\frac{1}{2}(F(\mathbf{x}_i, \mathbf{y}_i) + \max_{\mathbf{y}\neq\mathbf{y}_i} F(\mathbf{x}_i, \mathbf{y}))$. Hence

$$F(\mathbf{x}_i, \mathbf{y}_i) + \theta_i = \frac{1}{2}(F(\mathbf{x}_i, \mathbf{y}_i) - \max_{\mathbf{y}\neq\mathbf{y}_i} F(\mathbf{x}_i, \mathbf{y})) \geq \frac{1}{2}.$$

Similarly for $\mathbf{y} \neq \mathbf{y}_i$,

$$\begin{aligned} F(\mathbf{x}_i, \mathbf{y}) + \theta_i &= F(\mathbf{x}_i, \mathbf{y}) - \frac{1}{2}F(\mathbf{x}_i, \mathbf{y}_i) - \frac{1}{2}\max_{\mathbf{y}'\neq\mathbf{y}_i} F(\mathbf{x}_i, \mathbf{y}') \\ &\leq \frac{1}{2}(F(\mathbf{x}_i, \mathbf{y}) - F(\mathbf{x}_i, \mathbf{y}_i)) \leq \frac{1}{2}. \end{aligned}$$

Conversely if there is exists a threshold θ_i, then

$$F(\mathbf{x}_i, \mathbf{y}_i) - F(\mathbf{x}_i, \mathbf{y}) \geq \frac{1}{2} - \theta_i + \frac{1}{2} + \theta_i = 1.$$

The formulation with the thresholds ensures that only the relative distances between different labels of the same example matter, not the distances between different examples. We introduce the functions z_i to stress that we have basically obtained a binary classification problem, where $(\mathbf{x}_i, \mathbf{y}_i)$ take the role of positive examples and $(\mathbf{x}_i, \mathbf{y})$ for $\mathbf{y} \neq \mathbf{y}_i$ take the role of $|\mathcal{Y}| - 1$ negative pseudo-examples. The only difference to binary classification is that the bias can be adjusted for

each 'group' sharing the same pattern $\mathbf{x}_i$. Hence there is some additional interaction among pseudo-examples created from the same example $(\mathbf{x}_i, \mathbf{y}_i)$.

Following the standard procedure, we derive the dual formulation of this quadratic program. Straightforward calculation shows that the Lagrangian dual is given by

$$\begin{aligned} \max W(\alpha) = & -\frac{1}{2}\sum_{i,\mathbf{y}}\sum_{j,\bar{\mathbf{y}}}\alpha_i(\mathbf{y})\alpha_j(\bar{\mathbf{y}})z_i(\mathbf{y})z_j(\bar{\mathbf{y}})k_{i,j}(\mathbf{y},\bar{\mathbf{y}}) \\ & + \sum_{i,\mathbf{y}}\alpha_i(\mathbf{y}) \qquad (19) \\ \text{s.t.} \quad & \alpha_i(\mathbf{y}) \geq 0, \quad \forall i = 1,\ldots,n, \quad \forall \mathbf{y} \in \mathcal{Y} \\ & \sum_{\mathbf{y}\in\mathcal{Y}} z_i(\mathbf{y})\alpha_i(\mathbf{y}) = 0\,, \forall i = 1,\ldots,n \end{aligned}$$

where $k_{i,j}(\mathbf{y},\bar{\mathbf{y}}) = K((\mathbf{x}_i,\mathbf{y}),(\mathbf{x}_j,\bar{\mathbf{y}}))$. Notice that the equality constraints, which generalize the standard constraints for binary classification SVMs ($\sum_i y_i\alpha_i = 0$), result from the optimality conditions for the thresholds θ_i. In particular, this implies that $\alpha_i(\mathbf{y}) = 0$, if $\alpha_i(\mathbf{y}_i) = 0$, i.e. only if the positive example $(\mathbf{x}_i, \mathbf{y}_i)$ is a support vector, will there be corresponding support vectors created from negative pseudo-examples.

6. Optimization Algorithm for HM-SVM

Although it is one of our fundamental assumptions that a complete enumeration of the possible label set $\mathcal{Y}$ is intractable, the actual solution might be extremely sparse, since we expect that only very few negative pseudo-examples (which is possibly a very small subset of $\mathcal{Y}$) will become support vectors. Then, the main challenge in terms of computational efficiency is to design a computational scheme that exploits the anticipated sparseness of the solution.

Since the constraints only couple Lagrange parameters for the same training example, we propose to optimize W iteratively, at each iteration optimizing over the subspace spanned by all $\alpha_i(\mathbf{y})$ for a fixed i. Obviously, by repeatedly cycling through the data set and optimizing over $\{\alpha_i(\mathbf{y}) : \mathbf{y} \in \mathcal{Y}\}$, one defines a coordinate ascent optimization procedure that converges towards the correct solution, provided the problem is feasible (i.e., the training data is linearly separable). We prove this proposition by making use of the following two Lemmas.

Lemma 1 *Assume α^* is a solution of the Lagrangian dual problem in Eq. (19), then $\alpha_i^*(\mathbf{y}) = 0$ for all pairs $(\mathbf{x}_i, \mathbf{y})$ for which $F(\mathbf{x}_i, \mathbf{y}; \alpha^*) < \max_{\bar{\mathbf{y}}\neq\mathbf{y}_i} F(\mathbf{x}_i, \bar{\mathbf{y}}; \alpha^*)$.*

Proof by contradiction:
Define $\tilde{F}(\mathbf{x}_i;\alpha) = \max_{\mathbf{y}\neq\mathbf{y}_i} F(\mathbf{x}_i,\mathbf{y}_i;\alpha)$, then the optimal threshold needs to fulfill $\theta_i^* = -(F(\mathbf{x}_i,\mathbf{y}_i;\alpha^*) + \tilde{F}(\mathbf{x}_i;\alpha^*))/2$. Hence if $\mathbf{y}$ is a label sequences such that $F(\mathbf{x}_i,\mathbf{y};\alpha^*) < \tilde{F}(\mathbf{x}_i;\alpha^*)$ then

$$-F(\mathbf{x}_i,\mathbf{y};\alpha^*) - \theta_i^* > -\tilde{F}(\mathbf{x}_i;\alpha^*) - \theta_i^* = \frac{1}{2}(F(\mathbf{x}_i,\mathbf{y}_i;\alpha^*) - \tilde{F}(\mathbf{x}_i;\alpha^*)) \geq \frac{1}{2}.$$

Together with the assumption $\alpha_i^*(\mathbf{y}) > 0$ this contradicts the KKT complementary condition $\alpha_i^*(\mathbf{y})(F(\mathbf{x}_i,\mathbf{y};\alpha^*) + \theta_i^* + \frac{1}{2}) = 0$.

Lemma 2 *Define a matrix $D((\mathbf{x}_i,\mathbf{y}),(\mathbf{x}_j,\bar{\mathbf{y}})) \equiv z_i(\mathbf{y})z_j(\bar{\mathbf{y}})k_{i,j}(\mathbf{y},\bar{\mathbf{y}})$, then $\alpha' D e_i(\mathbf{y}) = z_i(\mathbf{y})F(\mathbf{x}_i,\mathbf{y})$, where $e_i(\mathbf{y})$ refers to the canonical basis vector corresponding to the dimension of $\alpha_i(\mathbf{y})$.*

Proof:

$$\begin{aligned} \alpha' D e_i(\mathbf{y}) &= z_i(\mathbf{y})\sum_{j,\mathbf{y}'}\alpha_j(\mathbf{y}')z_j(\mathbf{y}')k_{i,j}(\mathbf{y},\mathbf{y}') \\ &= z_i(\mathbf{y})F(\mathbf{x}_i,\mathbf{y})\,. \end{aligned}$$

We use a working set approach to optimize over the i-th subspace that adds at most one negative pseudo-example to the working set at a time. We define an objective for the i-th subspace by

$$W_i(\alpha_i; \{\alpha_j : j \neq i\}) \qquad (20)$$

which we propose to maximize over the arguments α_i while keeping all other α_j's fixed. Adopting the proof presented in [9], we prove the following result:

Proposition 2 *Assume a working set $S \subseteq \mathcal{Y}$ with $\mathbf{y}_i \in S$ is given and that a solution for the working set has been obtained, i.e. $\alpha_i(\mathbf{y})$ with $\mathbf{y} \in S$ maximize the objective W_i subject to the constraints that $\alpha_i(\mathbf{y}) = 0$ for all $\mathbf{y} \notin S$. If there exists a negative pseudo-example $(\mathbf{x}_i, \hat{\mathbf{y}})$ with $\hat{\mathbf{y}} \notin S$ such that $-F(\mathbf{x}_i,\hat{\mathbf{y}}) - \theta_i < \frac{1}{2}$, then adding $\hat{\mathbf{y}}$ to the working set $S' \equiv S \cup \{\hat{\mathbf{y}}\}$ and optimizing over S' subject to $\alpha_i(\mathbf{y}) = 0$ for $\mathbf{y} \notin S'$ will yield a strict improvement of the objective function.*

Proof:
Case I: If the training example $(\mathbf{x}_i, \mathbf{y}_i)$ is not a support vector (yet), then all $\alpha_i(\mathbf{y})$ in the working set will be zero, since $\alpha_i(\mathbf{y}_i) = \sum_{\mathbf{y}\neq\mathbf{y}_i}\alpha_i(\mathbf{y}) = 0$. Consider $\bar{\alpha}_i = \alpha_i + \delta e_i(\mathbf{y}_i) + \delta e_i(\hat{\mathbf{y}})$, for some $\delta > 0$. Then, the difference in cost function can be written as:

$$\begin{aligned} & W_i(\bar{\alpha}_i; \{\alpha_j : j \neq i\}) - W_i(\alpha_i; \{\alpha_j : j \neq i\}) \\ &= (\delta e_i(\mathbf{y}_i) + \delta e_i(\hat{\mathbf{y}}_i))'1 - \alpha' D(\delta e_i(\mathbf{y}_i) + \delta e_i(\hat{\mathbf{y}}_i)) \\ &\quad -\frac{1}{2}(\delta e_i(\mathbf{y}_i) + \delta e_i(\hat{\mathbf{y}}_i))' D(\delta e_i(\mathbf{y}_i) + \delta e_i(\hat{\mathbf{y}}_i)) \\ &= 2\delta - \delta\,(F(\mathbf{x}_i,\mathbf{y}_i) - F(\mathbf{x}_i,\hat{\mathbf{y}}_i)) - \mathbf{O}(\delta^2) \geq \delta - \mathbf{O}(\delta^2) \end{aligned}$$

since $F(\mathbf{x}_i, \mathbf{y}_i) - F(\mathbf{x}_i, \hat{\mathbf{y}}_i) < 1$. *By choosing* δ *small enough we can make* $\delta - \mathbf{O}(\delta^2) > 0$.

Case II: If the training example is a support vector, $\alpha_i(\mathbf{y}_i) > 0$ *then there has to be a negative pseudo-example* $\bar{\mathbf{y}}$ *with* $\alpha_i(\bar{\mathbf{y}}) > 0$. *Consider* $\bar{\alpha}_i = \alpha_i + \delta e_i(\hat{\mathbf{y}}_i) - \delta e_i(\bar{\mathbf{y}}_i)$.

$$\begin{aligned} & W_i(\bar{\alpha}_i; \{\alpha_j : j \neq i\}) - W_i(\alpha_i; \{\alpha_j : j \neq i\}) \\ &= (\delta e_i(\hat{\mathbf{y}}) - \delta e_i(\bar{\mathbf{y}}))' 1 - \alpha' D(\delta e_i(\hat{\mathbf{y}}) - \delta e_i(\bar{\mathbf{y}})) - \mathbf{O}(\delta^2) \\ &= \delta(F(\mathbf{x}_i, \hat{\mathbf{y}}) - F(\mathbf{x}_i, \bar{\mathbf{y}})) - \mathbf{O}(\delta^2) \end{aligned}$$

Hence we have to show that $F(\mathbf{x}_i, \hat{\mathbf{y}}) - F(\mathbf{x}_i, \bar{\mathbf{y}}) \geq \epsilon > 0$ *independent of* δ. *From the KKT conditions we know that* $-F(\mathbf{x}_i, \bar{\mathbf{y}}) - \theta_i = \frac{1}{2}$, *while our assumption was that* $-F(\mathbf{x}_i, \hat{\mathbf{y}}) - \theta_i < \frac{1}{2}$. *Setting* $\epsilon = \frac{1}{2} + \theta_i + F(\mathbf{x}_i, \hat{\mathbf{y}})$ *concludes the proof.*

The above proposition justifies the optimization procedure for the coordinate ascent over the i-th subspace, described in Algorithm 2.

Algorithm 2 Working set optimization for HM-SVMs.

1: $S \leftarrow \{\mathbf{y}_i\}$, $\alpha_i = 0$
2: **loop**
3: compute $\hat{\mathbf{y}} = \arg\max_{\mathbf{y} \neq \mathbf{y}_i} F(\mathbf{x}_i, \mathbf{y}; \alpha)$
4: **if** $F(\mathbf{x}_i, \mathbf{y}_i; \alpha) - F(\mathbf{x}_i, \hat{\mathbf{y}}; \alpha) \geq 1$ **then**
5: return α_i
6: **else**
7: $S \leftarrow S \cup \{\hat{\mathbf{y}}\}$
8: $\alpha_i \leftarrow$ optimize W_i over S
9: **end if**
10: **for** $\mathbf{y} \in S$ **do**
11: **if** $\alpha_i(\mathbf{y}) = 0$ **then**
12: $S \leftarrow S - \{\mathbf{y}\}$
13: **end if**
14: **end for**
15: **end loop**

7. Soft Margin HM-SVM

In the non-separable case, one may also want to introduce slack variables to allow margin violations. First, we investigate the case of L_2 penalties.

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{2}\sum_i \xi_i^2 \qquad (21) \\ \text{s.t.} \quad & z_i(\mathbf{y})(\langle \mathbf{w}, \Phi(\mathbf{x}_i, \mathbf{y}) \rangle + \theta_i) \geq 1 - \xi_i \\ & \xi_i \geq 0 \quad \forall i = 1, \ldots, n, \quad \forall \mathbf{y} \in \mathcal{Y} \end{aligned}$$

Notice that we only introduce a single slack variable per training data point and not per pseudo-example, since we want to penalize the strongest margin violation per sequence.

By solving the Lagrangian function for ξ_i, we get

$$\xi_i = \frac{1}{C}\sum_{\mathbf{y}} \alpha_i(\mathbf{y}) \qquad (22)$$

which gives us the following penalty term:

$$\frac{C}{2}\sum_i \xi_i^2 = \frac{1}{C}\sum_i \sum_{\mathbf{y},\mathbf{y}'} \alpha_i(\mathbf{y})\alpha_i(\mathbf{y}') \qquad (23)$$

Similar to the SVM case, this term can be absorbed into the kernel which is effectively changed to

$$\begin{aligned} K_C((\mathbf{x}_i, \mathbf{y}), (\mathbf{x}_i, \bar{\mathbf{y}})) &= K((\mathbf{x}_i, \mathbf{y}), (\mathbf{x}_i, \bar{\mathbf{y}})) \qquad (24) \\ &\quad + \frac{1}{C} z_i(\mathbf{y}) z_i(\mathbf{y}') \end{aligned}$$

and $K_C((\mathbf{x}_i, \mathbf{y}), (\mathbf{x}_j, \mathbf{y}')) = K((\mathbf{x}_i, \mathbf{y}), (\mathbf{x}_j, \mathbf{y}'))$ for $i \neq j$.

Using the more common L_1 penalty, one gets the following optimization problem

$$\begin{aligned} \min \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i \qquad (25) \\ \text{s.t.} \quad & z_i(\mathbf{y})(\langle \mathbf{w}, \Phi(\mathbf{x}_i, \mathbf{y}) \rangle + \theta_i) \geq 1 - \xi_i, \quad \xi_i \geq 0 \\ & \forall i = 1, \ldots, n, \quad \forall \mathbf{y} \in \mathcal{Y} \end{aligned}$$

Again the slack variable ξ_i is shared across all the negative pseudo-examples generated. The Lagrangian function for this case is

$$\begin{aligned} L &= \frac{1}{2}\|\mathbf{w}\|^2 + \sum_i (C - \rho_i)\xi_i \\ &\quad - \sum_{i,\mathbf{y}} \alpha_i(\mathbf{y}) \left[z_i(\mathbf{y}) \left(F(\mathbf{x}_i, \mathbf{y}) + \theta_i \right) - 1 + \xi_i \right] \qquad (26) \end{aligned}$$

with non-negativity constraints on the dual variables $\rho_i \geq 0$ and $\alpha_i(\mathbf{y}) \geq 0$. Differentiating w.r.t. ξ_i gives:

$$\sum_{\mathbf{y}} \alpha_i(\mathbf{y}) = C - \rho_i \leq C \qquad (27)$$

The box constraints on the $\alpha_i(\mathbf{y})$ thus take the following form

$$0 \leq \alpha_i(\mathbf{y}), \quad \text{and} \quad \sum_{\mathbf{y} \in \mathcal{Y}} \alpha_i(\mathbf{y}) \leq C. \qquad (28)$$

In addition, the KKT conditions imply that whenever $\xi_i > 0$, then $\sum_{\mathbf{y} \in \mathcal{Y}} \alpha_i(\mathbf{y}) = C$ which means that $\alpha_i(\mathbf{y}_i) = \sum_{\mathbf{y} \neq \mathbf{y}_i} \alpha_i(\mathbf{y}) = C/2$.

Hence, one can use the same working set approach proposed in Algorithm 2 with different constraints in the quadratic optimization in step 8.

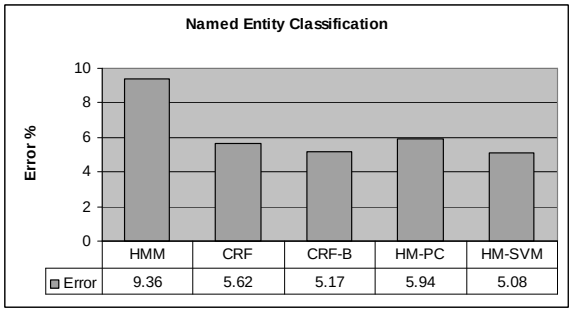

Figure 1. Test error of NER task over a window of size 3 using 5-fold cross validation.

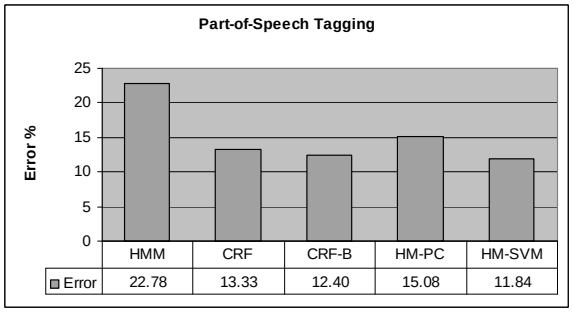

Figure 2. Test error of POS task over a window of size 3 using 5-fold cross validation.

8. Applications and Experiments

8.1. Named Entity Classification

Named Entity Recognition (NER) is an information extraction problem which deals with finding phrases containing person, location and organization names, as well as temporal and number expressions. Each entry is annotated with the *type* of its expression and its *position* in the expression, i.e. the beginning or the continuation of the expression.

We generated a sub-corpus consisting of 300 sentences from the Spanish news wire article corpus which was provided for the Special Session of CoNLL2002 on NER. The expression types in this corpus are limited to person names, organizations, locations and miscellaneous names, resulting in a total of $|\Sigma| = 9$ different labels.

All input features are simple binary features. Most features are indicator functions for a word occurring within a fixed size window centered on the word being labeled. In addition there are features that do not only encode the identity of the word, but also more detailed properties (e.g. spelling features). Notice that these features are combined with particular label indicator functions in the joint feature map framework. Some example features are: "Is the previous word '*Mr.*' and the current tag '*Person-Beginning*'?", "Does the next word end with a dot, and is the current tag '*Non-name*'?", and "Is the previous tag '*Non-name*' and the current tag '*Location-Intermediate*'?".

We compared the performance of HMMs and CRFs with the HM-Perceptron and the HM-SVM with L_2-penalties. Overlapping features with a window of size 3 were used in all experiments. Although in a generative model like an HMM, overlapping features violate the model, we observed that HMMs using the overlapping features described above outperformed ordinary HMMs. For this reason, we only report the results of HMMs with overlapping features. The CRFs have been optimized using a conjugate gradient method which has been reportedly outperformed other techniques for minimizing the CRF loss function [8]. Since optimizing log-loss functions (as is done in CRFs) may result in overfitting, especially with noisy data, we have followed the suggestion of [5] and used a regularized cost function. We refer to this CRF variant as *CRF-B*.

The results summarized in Figure 1 demonstrate the competitiveness of HM-SVMs. As expected, CRFs perform better than the HM-Perceptron algorithm (*HM-PC*), since CRFs use the derivative of the log-loss function at every step, whereas the Perceptron algorithm uses only an approximation of it (cf. [2]). HM-SVMs achieve the best results, which validates our approach of explicitly maximizing a soft margin criterion.

8.2. Part-Of-Speech Tagging

We extracted a corpus consisting of 300 sentences from the Penn TreeBank corpus for the Part-Of-Speech (POS) tagging experiments. The features and experimental setup is similar to the NER experiments. The total number of function tags was $|\Sigma| = 45$. Figure 2 summarizes the experimental results obtained on this task. Qualitatively the behavior of the different optimization methods is comparable to the NER experiments. All discriminative methods clearly outperform HMMs, with HM-SVMs outperforming the other methods.

8.3. Protein Secondary Structure Prediction

Protein secondary-structure prediction is an important step towards solving one of the most challenging prob-

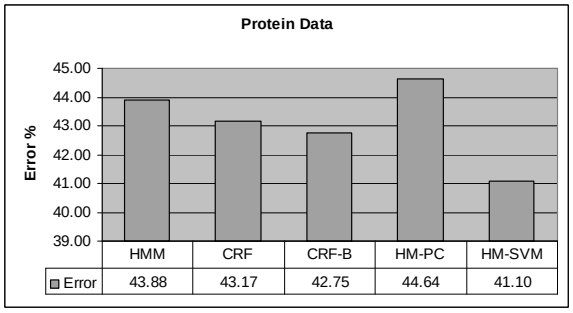

Figure 3. Test error obtained on the RS126 data sets using 7-fold cross validation.

lems in molecular biology: the *protein folding problem*. Simplifying the 3-D problem by projecting the complex structure to a sequence, the task is to classify segments of an aminoacid sequence as *helix* (H), *sheet* (E) or *coil* (C).

We used the RS126 data set, which consists of 126 proteins, sequences over an alphabet of size 20. We represented each protein in an orthogonal representation by converting the 20-valued feature at each residue (i.e. the identity of the amino acid) into 20 binary features, of which exactly one has a value of 1 at a time.

We performed experiments using polynomial kernels of varying degree and different window sizes. The reported results are for a window of size 11 and second order features, implemented via a polynomial kernel in the HM-Perceptron and HM-SVM case. Figure 3 summarizes our results. Again, HM-SVMs obtain the lowest error rate.

9. Conclusion

We presented HM-SVMs, a novel discriminative learning technique for the label sequence learning problem. This method combines the advantages of maximum margin classifier and kernels with the elegance and efficiency of HMMs. Our experiments prove the competitiveness of HM-SVMs in terms of the achieved error rate on three benchmark data sets. HM-SVMs have several advantages over other methods, including the possibility of using a larger number and more expressive features. We are currently addressing the numerical and scalability issues to be able to perform larger scale experiments.

Acknowledgments

This work was sponsored by an NSF-ITR grant, award number IIS-0085940.

References

[1] Y. Altun, T. Hofmann, and M. Johnson. Discriminative learning for label sequences via boosting. In *Advances in Neural Information Processing Systems (NIPS*15)*, 2002.

[2] M. Collins. Discriminative training methods for Hidden Markov Models: Theory and experiments with perceptron algorithms. In *EMNLP*, 2002.

[3] M. Collins and N.l Duffy. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

[4] T. Hofmann, I. Tsochantaridis, and Y. Altun. Learning over structured output spaces via joint kernel functions. In *Advances in Neural Information Processing Systems, Workshop on Kernel Methods*, 2002.

[5] M. Johnson, S. Geman, S. Canon, Z. Chi, and S. Riezler. Estimators for stochastic unification-based grammars. In *Proceedings ACL'99, Maryland.*, 1999.

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

[7] A. McCallum, D. Freitag, and F. Pereira. Maximum Entropy Markov Models for Information Extraction and Segmentation. In *Machine Learning: Proceedings of the Seventeenth International Conference (ICML 2000)*, pages 591–598, Stanford, California, 2000.

[8] T. Minka. Algorithms for maximum-likelihood logistic regression. Technical report, CMU, Department of Statistics, TR 758, 2001.

[9] E. Osuna, R. Freund, and F. Girosi. Training support vector machines:an application to face detection. In *CVPR'97*, 1997.

[10] V. Punyakanok and D. Roth. The use of classifiers in sequential inference. In *NIPS*, pages 995–1001, 2000.

[11] J. Weston, O. Chapelle, A. Elisseeff, B. Schölkopf, and V. Vapnik. Kernel dependency estimation. Technical report, Max Planck Institute for Biological Cybernetics, Tübingen, Germany, August 2002.