Discriminatively Trained Markov Model for Sequence Classification

Oksana Yakhnenko, Adrian Silvescu, Vasant Honavar
Artificial Intelligence Research Laboratory
Center for Computational Intelligence Learning and Discovery
Computer Science Department
Iowa State University
Ames, Iowa 50010
Email: {oksayakh, silvescu, honavar}@cs.iastate.edu

***Abstract*— In this paper, we propose a discriminative counterpart of the directed Markov Models of order $k-1$, or MM($k-1$) for sequence classification. MM($k-1$) models capture dependencies among neighboring elements of a sequence. The parameters of the classifiers are initialized to based on the maximum likelihood estimates for their generative counterparts. We derive gradient based update equations for the parameters of the sequence classifiers in order to maximize the conditional likelihood function. Results of our experiments with data sets drawn from biological sequence classification (specifically protein function and subcellular localization) and text classification applications show that the discriminatively trained sequence classifiers outperform their generative counterparts, confirming the benefits of discriminative training when the primary objective is classification. Our experiments also show that the discriminatively trained MM($k-1$) sequence classifiers are competitive with the computationally much more expensive Support Vector Machines trained using k-gram representations of sequences.**

I. Introduction

Sequence classification is an important problem that arises in many real-world applications: protein function prediction, text classification, speech recognition, intrusion detection, among others. Given a sequence (constructed from letters drawn from a finite alphabet; for instance, 20-letter alphabet of amino acids in the case of protein function classification; a vocabulary of English words in the case of text classification), the task of a sequence classifier is to assign a class label (typically drawn from a finite set of mutually exclusive class labels) to the sequence. Machine learning algorithms offer one of the most cost effective approaches to designing sequence classifiers when a training set of labeled sequences is available. Of particular interest are sequence classifiers based on probabilistic generative models of sequence data. Given a generative model, a new sequence can be assigned the most probable class label. A Naïve Bayes sequence classifier is based on one of the simplest generative models of sequence data - namely one that assumes that each element (letter) of the sequence is *independent* of other elements of the sequence *given* the class. Training a Naïve Bayes sequence classifier from data simply involves estimating the probabilities for each letter of the alphabet conditioned on the class. Naïve Bayes classifiers, because of their simplicity and low computation cost, are often used in applications ranging from text classification [14] to biological sequence classification [7] [24] with varying degrees of success (as estimated by standard measures of classifier performance - e.g., classification accuracy). However, the independence assumption of Naïve Bayes - which is tantamount to ignoring the sequential nature of the data - is often violated in practice. Hence, generative models of sequence data that relax the strong independence assumption of the Naïve Bayes model, the associated sequence classifiers, and efficient algorithms for learning accurate sequence classifiers from data are of significant interest.

The dependencies among the neighbouring elements of a sequence can be modeled by using k-grams [6]. One might consider a Naïve Bayes classifier whose inputs consist of k-grams as opposed to single letters. However, because successive k-gram in a sequence have $k-1$ elements in common, the use of k-grams as input features for a Naïve Bayes classifier consistently and systematically violates the Naïve Bayes assumption - that the inputs to the classifier are independent given the class. By exploiting the well-known Hammersley-Clifford theorem [8], it is possible to construct a generative Markov Model of order $k-1$, or Markov Model($k-1$) (or MM($k-1$) for short) [16] or equivalently, the so-called Naïve Bayes(k) (or NB(k) for short) [1] that models the dependencies among k neighboring elements of a sequence. This model was shown to have consistent improvement in accuracy relative to Naïve Bayes in sequence-based protein function classification [1], as well as text classification [16] with the performance of the classifier improving with increasing values of k (when sufficient training data are available).

Generative models (such as Naïve Bayes or Naïve Bayes(k)) model the probability distribution of the process generating the data from each class. Classification is performed by examining the likelihood of each class producing the observed features in the data (e.g., letters in the sequence) and assigning the sequence to the most likely class (i.e., one with the largest likelihood). In contrast, *discriminative models* directly compute class membership probabilities (or model class boundaries), without modeling the underlying class feature densities. For example, classifiers trained using logistic regression are the discriminative counterparts of the Naïve Bayes generative models [21]. Discriminative models can often outperform generative models on classification tasks [21] [15]. One of the reasons for choosing discriminative models over generative models for classification was pointed out by Vapnik [23]: “One should solve the [classification] problem directly and never solve a more general problem as an intermediate step” (such as modeling the probability distribution of features in the data for each class). Consequently, there has been significant recent interest in the training of discriminative counterparts of generative models for classification tasks. For example, Ng and Jordan [15] have compared Naïve Bayes and Logistic regression classifiers; Bouchard and Triggs [5] have explored the trade-offs between generative and discriminative classifiers; Raina et. al. [18] have studied a hybrid classifier that combines generative and discriminative models for text classification.

Against this background, in this paper, we explore the discriminative counterpart of the Naïve Bayes(k) generative model for sequence classification. We derive a gradient based learning algorithm for training sequence classifiers in a discriminative setting. Results of our experiments on data sets drawn from biomolecular sequences and text classification applications show that the discriminatively trained MM($k-1$) classifiers outperform their generative counterparts and compete favorably with Support Vector Machines trained using k-gram representations of sequences.

The rest of the paper is organized as follows: Section 2 describes the preliminaries of a generative Markov Model, and shows how it can be trained discriminatively using gradient ascent. Section 3 presents the algorithm for discriminative training of MM($k-1$) sequence classifiers. Section 4 describes experimental set-up and results. Section 5 concludes with discussion of related work and an outline of some directions for further research on this topic.

II. Generative and Discriminative Markov Models

We start by outlining the general classification strategy for the generative setup. We then present the Markov Model of order $k-1$ and apply the general strategy to build a classifier from this model, and show how it can be trained in a discriminative setting.

A. Generative Probabilistic Model

A model α for sequences defined over some alphabet Σ specifies for any sequence $S \in \Sigma^*$ the probability $P_\alpha(S)$. Given a set of sequences, each instance is associated with a class from the set of mutually disjoint class labels $C = \{c_1...c_n\}$. A classifier can be constructed in the following fashion: First, a probabilistic model $\alpha(c_j)$ is trained for each class using the sequences belonging to c_j; and second, in order to predict a class for a novel sequence S' Bayes rule is used to obtain $class(S') = \arg\max \frac{P_{\alpha(c_j)}(S'|c_j)P(c_j)}{P_{\alpha(c_j)}(S')}$

B. MM($k-1$): Markov Model of order $k-1$

Markov Models for sequence classification have been used with success by many researchers in a variety of applications [6] [25] [1]. We start with a brief review of the basic ideas behind MM($k-1$).

Let S be a sequence, s_i be the value of an element of S at the position i, and Σ be the alphabet over which the sequence values range.

A sequence can be modeled as a graph in which each sequence element is represented by a node, and a direct dependency between two neighboring elements is represented by an edge in the graph. Generally, it is the case that two or more neighboring elements in the sequence will be dependent on each other. Figure 1 shows several directed dependency models for: a) Naïve Bayes which corresponds to assuming no dependence between neighboring elements of a sequence, b) a dependency model of the first order i.e., one that considers dependencies between pairs of neighboring elements of a sequence ($k=2$) and c) a dependency model of the second order i.e., one that considers dependencies among three adjacent elements of a sequence. More generally, Markov Models of order $k-1$, capture the dependency between the current element s_k and its $k-1$ preceding elements $[s_{k-1}...s_1]$ in a sequence. MM($k-1$) family of generative models offer the needed expressive power to model increasingly complex dependencies among neighboring elements of a sequence.

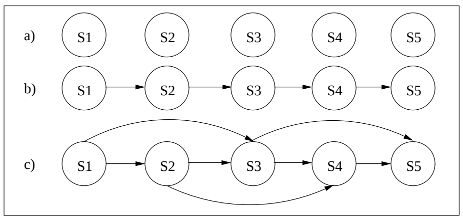

Fig. 1. Markov Model representation of dependencies of order $k-1$ dependency. a) Naïve Bayes model; b) Markov Model of order 1 c) Markov Model of order 2

The joint probability distribution for the MM($k-1$) follows directly from the Junction Tree Theorem [8] and the definition of conditional probability:

$$\begin{aligned} P(S = s_1 s_2 ... s_n, c_j) &= \frac{\prod_{i=1}^n P(S = s_i...s_{i+k-1}, c_j)}{\prod_{i=1}^n P(S = s_i...s_{i+k-2}, c_j)} \\ &= P(S = s_1...s_{k-1}, c_j) \\ &\quad \prod_{i=k}^{n} P(S = s_i | s_{i-1}...s_{i-k+1}, c_j) \end{aligned}$$

The probabilities $P(S = s_i|s_{i-1}...s_{i-k+1}, c_j)$ can be readily estimated from data using the counts of the subsequences $s_i...s_{i-k+1}, c_j$ and $s_{i-1}...s_{i-k+1}, c_j$ as sufficient statistics. With the generative model in place, a sequence S to be classified is assigned to the most likely class based on the generative models for each class. That is, $class(S = s_1...s_n) = \arg\max_{c_j \in C} P(S = s_1...s_{k-1}, c_j) \prod_{i=k}^n P(S = s_i|s_{i-1}...s_{i-k+1}, c_j)$. Markov Models of order $k-1$ (where $k>1$) have been shown to have consistent improvement in accuracy over Naïve Bayes (which is equivalent to a Markov Model of order 0), with the classification accuracy typically increasing with the increase in k (until we run out of data to reliably estimate the increased number of parameters) [1].

C. DMM($k-1$): Discriminatively Trained Markov Models of order $k-1$

As noted earlier, when the primary objective is classification, discriminatively trained classifiers often outperform their generative counterparts. Discriminative training of classifiers entails maximizing the conditional likelihood function [21].

In the case of MM($k-1$), the conditional likelihood function for a sequence $S = s_1 ... s_n$ belonging to class $c = c_q$ is given by:

$$P(c = c_q | S = s_1 ... s_n) = \frac{P(S = s_1 ... s_n, c_q)}{\sum_{c_j \in C} P(S = s_1 ... s_n, c_j)}$$

To minimize the notational clutter, we will use $P_{i_1 ... i_k, c_q}$ to denote $P(S = s_i | s_{i-1} ... s_{i-k+1}, c_j)$, and $P_{i_1 ... i_{k-1}, c_q}$ to denote $P(S = s_i, s_{i-1} ... s_{i-k}, c_j)$. Note that the elements of the sequence take values from the alphabet Σ and class labels take values from C. These parameters collectively represent the corresponding model. We will use P to denote this collection of parameters that specify the model.

Consider a data set $D = \{d^1 ... d^m\}$ of labeled strings where the strings consist of elements from a finite alphabet Σ and the class labels are drawn from a finite set of class labels C. Thus, $d^l = (S^l, c^l), 1 \leq l \leq m; n_l = |S^l|$ is length of S^l and $(\forall l \forall k \leq n_l)\ s^l_k \in \Sigma$ and $c^l \in C$.

Let $CL(d^l : P)$ denote the conditional likelihood of the sequence d^l under the model P. Let $CL(D : P)$ denote the conditional likelihood of the data set D under the model P. Assuming independently identically distributed samples, we have:

$$CL(D : P) = CL(D = \{d^1 ... d^m\} : P) = \prod_{l=1}^{m} CL(d^l : P)$$

Letting $CLL(d^l : P)$ denote $\log CL(d^l : P)$ and $CLL(D : P)$ denote $\log CL(D : P)$ respectively, we can write:

$$\begin{aligned} CLL(D : P) &= \sum_{l=1}^{m} CLL(d^l : P) \\ &= \sum_{l=1}^{m} \log P(C^l = c^l | S^l = s^l_1 ... s^l_{n_l}) \end{aligned}$$

Training the model P in the discriminative setting is equivalent to maximizing the conditional likelihood function $CL(D : P)$, with respect to the parameters P which in turn is equivalent to maximizing the conditional log-likelihood $CLL(D : P)$. We use gradient ascent to find parameters that maximize $CLL(D : P)$. We start with an initial estimate $P(0)$ of P (more on this initialization in Section 3). Denoting by $P(t)$ and $P(t+1)$ the estimates of P at iterations t and $t+1$ respectively, the update equation for each parameter $P_{i_1 ... i_k c_q}$ at the iteration t is given by

$$P_{i_1 ... i_k c_q}(t+1) \leftarrow P_{i_1 ... i_k c_q}(t) + \alpha \left. \frac{\partial CLL(D : P)}{\partial P_{i_1 ... i_k c_q}} \right|_{P = P(t)}$$

where $0 < \alpha \leq 1$ is the learning rate. Thus, all parameters in P are iteratively updated until a desired termination condition is satisfied.

It is convenient to reparametrize $P_{i_1 ... i_k, c_q}$ in terms of $w_{i_1 ... i_k, c_q}$ by setting $P_{i_1 ... i_k, c_q} = \frac{1}{Z_w} e^{w_{i_1 \cdots i_k, c_q}}$, where $Z_w = \sum_{i_1 \in \Sigma} e^{w_{i_1 \cdots i_k, c_q}}$ is the normalization factor for each parameter. This reparameterization is needed in order to maintain the probabilities within the $[0, 1]$ interval. Note that since we are normalizing over the first entry in the k-gram, the normalized result still corresponds to conditional probability of an element given its $k-1$ predecessors $P(S = s_i | s_{i-1} ... s_{i-k+1}, c_q)$. The second parameter, $P_{i_1 ... i_{k-1}, c_q}$, is reparameterized in terms of $u_{i_1 ... i_{k-1}, c_q}$ by setting $P_{i_1 ... i_{k-1}, c_q} = \frac{1}{Z_u} e^{u_{i_1 \cdots i_{k-1}, c_q}}$ and $Z_u = \sum_{i_1 ... i_{k-1} \in \Sigma; c_q \in C} e^{u_{i_1 \cdots i_{k-1}, c_q}}$ is the normalization factor over all possible values of $i_1 ... i_{k-1}, c_q$.

Let W denote the counterpart of P after this reparameterization. Now $CLL(d^l : W)$ for a data point is given by:

$$CLL(c_j | S^l) = \log \frac{\gamma_j}{Z_\gamma}$$

where $\gamma_j = \frac{e^{u_{s^l_1 \cdots s^l_{k-1}, c_j}}}{Z_u} \prod_{r=k}^{n^l} \frac{e^{w_{s^l_r \cdots s^l_{r-k+1}, c_j}}}{Z_w}$, and $Z_\gamma = \sum_{c_j \in C} \gamma_j$ corresponds to normalization over all classes.

The update equation for the resulting parameters after the above reparameterization is given by:

$$w_{i_1 ... i_k, c_q}(t+1) \leftarrow w_{i_1 ... i_k, c_q}(t) + \alpha \left. \frac{\partial CLL(D : P)}{\partial w_{i_1 ... i_k, c_q}} \right|_{W = W(t)}$$

The parameters $u_{i_1 ... i_{k-1}, c_q}$ are updated in the similar fashion.

In what follows, we present the formula for the gradient of the conditional log-likelihood function for an individual instance d^l in the data set D. Since the gradients commute with summation, we can compute the gradient for the entire dataset D by simply summing up the gradients for the individual samples from D. Furthermore, if online update is used, we can update the parameters based on individual samples from D.

Let $count[i_1 ... i_k : S^l]$ be the number of times a subsequence $s_1 .. s_k$ appears in the sequence S^l, and let $\delta(c_j, c_q)$ be 1 if $c_j = c_q$ and be 0 otherwise. The update equation for $w_{i_1 ... i_k, c_q}$ is obtained by taking the derivative of $CLL(d^l : W)$ with respect to $w_{i_1 ... i_k, c_q}$:

$$\begin{aligned} \frac{\partial CLL(d^l : W)}{\partial w_{i_1 ... i_k, c_q}} = \ & count[i_1, ..., i_{k-1} : S^l] \\ & \left(\frac{count[i_1, ..., i_k : S^l]}{count[i_1, ..., i_{k-1} : S^l]} - \frac{e^{w_{i_1 \cdots i_k, c_q}}}{Z_w} \right) \\ & \left(\delta(c_j, c_q) - \frac{\gamma(s^l_1, ..., s^l_{n_l}, c_q)}{Z_\gamma(s^l_1, ..., s^l_{n_l})} \right) \end{aligned}$$

Similarly, the update equation for $u_{i_1 ... i_{k-1}, c_q}$ is obtained by taking the derivative of the $CLL(d^l : W)$ with respect to $u_{i_1 ... i_{k-1}, c_q}$:

$$\frac{\partial CLL(d^l : W)}{\partial u_{i_1 \ldots i_{k-1}, c_q}} = (\delta([i_1, ..., i_{k-1}], [s^l_1, ..., s^l_{k-1}]) - \frac{e^{u_{i_1 \ldots i_{k-1}, c_q}}}{Z_u}) \left(\delta(c_j, c_q) - \frac{\gamma(s^l_1, ..., s^l_{n_l}, c_q)}{Z_\gamma(s^l_1, ..., s^l_{n_l})}\right)$$

A closer look at the update equations reveals that the term $\delta(c_j, c_q) - \frac{\gamma(s^l_1,...,s^l_{n_l},c_q)}{Z_\gamma(s^l_1, ..., s^l_{n_l})}$ is the penalty for the parameters that are associated with a class that is not the true class of the sequence, or the reward for the parameters associated with the true class. Since at the maxima or minima, the derivative of a function is 0, it follows that $\delta(c_j, c_q) - \frac{\gamma(s_1,...,s_n,c_q)}{Z_\gamma(s_1, ..., s_n)}$ should be 0, which is the case when the probability of a sequence given its true class is 1.

III. Training Procedure

Training of DMM($k-1$) (discriminative Markov Model of order $k-1$) proceeds as follows.

The parameters of MM($k-1$), i.e., each $w_{i_1 \ldots i_k, c_j}$ and each $u_{i_1 \ldots i_{k-1}, c_j}$ is initialized with the logarithm of counts of subsequences associated with the parameters. This corresponds to initializing P (equivalently W), with the parameters of the generative MM($k-1$). The parameters are then iteratively updated using the gradient ascent procedure until a desired termination criterion is satisfied (more on this in the next section). The momentum modification was used to speed up convergence [17]. Once we have trained the model, we can classify an input sequence by assigning it to the most probable class under the model. The pseudocode for the algorithm is shown in Figure 2.

IV. Experimental Setup and Results

This section describes the datasets, experimental setup used to test our model and provides direct comparison of a DMM($k-1$) with its generative counterpart MM($k-1$), as well as with a Support Vector Machine using a data representation with the same representational power as that of MM($k-1$) models.

A. Datasets

The experiments were conducted on data sets drawn from two application domains: biomolecular sequence classification and text classification. In the case of biomolecular sequence classification, we have experimented with data sets used in the protein function and in the subcellular localization studies. In the case of text classification, we have experimented with the widely used Reuters Text Categorization data set.

1) Protein Function Data: The first biomolecular sequence dataset is based on the families of human kinases. The Kinase dataset contains a total of 290 protein sequences belonging to four functional classes containing 1, 10, 69, 210 sequences respectively. The goal is to assign a given kinase sequence to one of the four kinase subfamilies. This dataset was previously used in the protein function studies by Andorf et. al [1].

Training

```
Input: training data $D = \{d^1 ... d^m\}$
       model order $k$, learning rate $\alpha$, momentum $\lambda$
Output: $W$ (parameters of the trained model)
1. Initialize $w_{i_1 \ldots i_k, c_q} \leftarrow \log count[i_1 ... i_k, c_q]$
2. Initialize $u_{i_1 \ldots i_{k-1}, c_q} \leftarrow \log count[i_1 ... i_{k-1}, c_q]$
3. For each $d^l = (S^l, c^l) \in D$
4. do until termination criterion is met
   {
5.   For each $i_1 ... i_{k-1} \in S^l$
6.     For each $v \in \Sigma$
7.       For each $c_j \in C$
8.         $w_{v, i_1 \ldots i_{k-1}, c_j} \leftarrow w_{v, i_1 \ldots i_{k-1}, c_j} + \alpha \frac{\partial CLL(d^l : W)}{\partial w_{v, i_1 \ldots i_{k-1}, c_j}} + \lambda w_{v, i_1 \ldots i_{k-1}, c_j}$
10.      For each $c_j \in C$
11.        $u_{i_1 \ldots i_{k-1}, c_j} \leftarrow u_{i_1 \ldots i_{k-1}, c_j} + \alpha \frac{\partial CLL(d^l : W)}{\partial u_{i_1 \ldots i_{k-1}, c_j}} + \lambda u_{i_1 \ldots i_{k-1}, c_j}$
   }
Output($W$)
```

Classification

```
Input: model order $k$, sequence to be classified $S = s_1 s_2 ... s_n$
1. $c \leftarrow \arg\max_{c_j \in C} \frac{u_{1..k-1, c_j}}{Z_u} \prod_{i=k}^{n} \frac{e^{w_{i \ldots i-k, c_j}}}{Z_w}$
2. output $c$
```

Fig. 2. DMM($k-1$): Training and Classification

2) Subcellular Localization Data: In the subcellular localization task [19], the goal is to predict the subcellular localization of a protein from its sequence. The data sets used in our experiments are the same as those explored in the previous studies on predicting cellular localization of protiens by Bhasin et. al. [4] and Hua et. al. [11]. These data sets are based on localization data for prokaryotic and eukaryotic proteins derived from Swiss-Prot database (release 33.0) [2]. The first dataset comprises a total of 997 prokaryotic proteins. It includes the proteins from the following three different subcellular locations: cytoplasmic, periplasmic, and extracellular. The second data set consists of a total of 2427 eukaryotic proteins. It includes proteins from four different subcellular locations: nuclear, cytoplasmic, mitochondrial, and extracellular. Sequence identity within a class is below 90%. No transmembrane proteins were included. Table I summarizes some characteristics of these two datasets.

3) Text Data: We used the Reuters-21578 Text Categorization datasets[1] with the 10 categories (acquisition, corn, crude, earn, grain, interest, money-fx, ship, trade and wheat) that have the highest number of positive training examples. However, because of the large size of the vocabulary, we performed feature selection based on mutual information [9] to reduce the vocabulary size to 300 words. Even with this reduced vocabulary, the estimation of parameters MM($k-1$) or SVM classifiers becomes infeasible for values of k greater

[1]http://www.daviddlewis.com/resources/testcollections/reuters21578/

TABLE I

NUMBER OF SEQUENCES FOR EACH SUBCELLULAR LOCALIZATION CATEGORIES OF THE DATASETS

Species	Subcellural localization	Number of sequences
Prokaryotic	Cytoplasmic	688
	Periplasmic	202
	Extracellular	107
Eukaryotic	Nuclear	1097
	Cytoplasmic	684
	Mitochondrial	321
	Extracellular	325

than 2 because of the sparsity of the data as well as memory requirements unless additional steps are taken to reduce the size of the models. Hence, we limit the experiments reported in this paper experiments to values of k no greater than 2.

B. Experiments Exploring the Conditional-Log-Likelihood vs. Accuracy on Training and Validation Sets

Dicriminatively trained classifiers have been reported to be more susceptible to overfitting than their generative counterparts [15]. It is therefore interesting to explore whether the DMM($k-1$) models suffer from overfitting. Hence, we start with an examination of the behavior of the DMM($k-1$) model by observing the relationship between maximizing conditional likelihood versus accuracy on a validation set (data not used in training of the classifier). We show the behavior of the class-conditional log-likelihood, accuracy on the training data, and accuracy on the previously unseen data in the case of the Eukaryotic subcellular localization datasets (Figure 3). The datasets were split into training set and validation set in the ratio 90%-10%. The values of the negative conditional log-likelihood function and the accuracies on the training set and on the validation set were recorded at each iteration of the algorithm, and repeated for the values of $k = 2, 3, 4$. The values of the negative CLL are plotted on the right y-axis, and the values of accuracy were plotted on the left y-axis, with the number of iterations plotted on the x-axis.

In the case of the Eukaryotic dataset, when k=2 for Cytoplasmic and Nuclear classes, accuracy on training data goes from 80% to 89% and 82% to 92% respectively after which the training accuracy begins to drop even though negative CLL contintues to decrease. Accuracy on the validation data grows from 78% to 82% and 80% to 87% respectively in several iterations and then starts to decrease suggesting the possibility that the model is overfitting the parameters. For Extracelluajar and Mitochondrial classes, the accuracy on the training data increases with the number of iterations of the algorithm, and so does the accuracy on the validation data for Extracellular class. For the Mitochondrial class we again observe increase in accuracy after the first several iterations, followed by a decrease in accuracy even though negative CLL continues to decrease. For $k = 3$ in Cytoplasmic, Mitochondrial and Nuclear classes, the accuracy on validation set decreases and then increases to its starting value, even though accuracy on the training set increases. In the case of the Extracellular class,

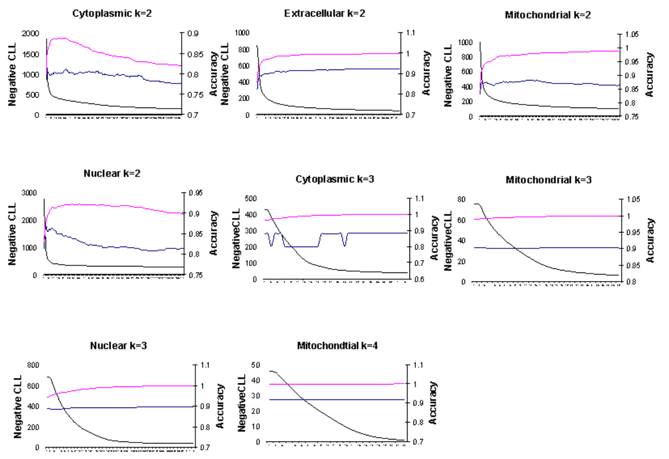

Fig. 3. Negative Conditional Log-Likelihood versus accuracy on training and validation set for Eukaryotic set for classes Cytoplasmic, Extracellular, Mitochondrial and Nuclear for k=2; Cytoplasmic, Mitochondrial and Nuclear for k=3; Mitochondrial for k=4

the plots are not shown for $k = 3$ because the negative CLL was found to be close to 0 at initialization. Finally, for $k = 4$ even though the negative CLL decreases from 50 to almost 0, the accuracy on training and validation sets fail to show improvement as training progresses.

The observation that in a few instances, accuracy on the training data drops as the the conditional log likelihood score continues to increase, at first glance, is somewhat puzzling. Once possible explanation is that the conditional log likelihood tends to favor short sequences over longer sequences (by virtue of the fact that probabilities associated with k-grams get multiplied in computing likelihood) whereas classification accuracy measures the percentage of sequences in the data set that are correctly labeled, regardless of their length. This suggests a possible direction for improving our current algorithm - normalizing the conditional likelihood score for each sequence by the length of the sequence

It is clear that the gradient achieves the highest accuracy on the validation data within several iterations, and most of the time with maximization of the likelihood function, the accuracy on the unseen data drops, which suggests overfitting. Therefore, in our experiments we choose the parameters when the accuracy on the validation set was at its highest value in the first 30 iterations.

C. Experimental Comparison of DMM($k-1$) model with MM($k-1$) and SVM

For all the experiments we have used 10-fold cross-validation to evaluate our approach and to compare it with generatively trained model and the SVM. Furthermore, 10% of the training data during each round of cross-validation was withheld for the validation purposes. After 30 iterations the parameters that yield the highest accuracy on the validation set were used in the resulting classifier. Note that the classifier is tested on the data not used for training or validation. The parameters α and λ were set to those that yield the fastest

TABLE II

ACCURACY ON KINASE DATASET OBTAINED BY 10-FOLD CROSS-VALIDATION, DMM($k-1$), MM($k-1$) $k=2,3,4$, AND SVM FOR $k=2,3$ ($k=4$ IS COMPUTATIONALLY INFEASIBLE FOR SVM)

k	DMM	MM	SVM
2	**89.01±5.89**	87.11±7.2	90.7
3	91.55±5.82	91.13±5.5	90.3
4	78.48±6.5	78.32±8.8	x

TABLE III

ACCURACY, SENSITIVITY, SPECIFICITY AND CORRELATION COEFFICIENT FOR DMM($k-1$), MM($k-1$), AND SVM ON THE PROKARYOTIC DATASET OBTAINED BY 10-FOLD CROSS-VALIDATION

		Accuracy			***Specificity***		
Class	k	DMM	MM	SVM	DMM	MM	SVM
Peri	2	82.91	84.07	**90.17**	87.77	87.67	**92.7**
	3	86.78	87.2	**88.26**	97.74	97.61	97.23
	4	82.57	82.13	x	85.56	85.1	x
Extr	2	**95.05**	94.01	93.68	**98.52**	96.09	97.64
	3	94.78	94.88	94.88	99.17	99.08	99.44
	4	86.84	86.44	x	88.4	87.78	x
Ctpl	2	90.01	90.59	90.17	70.61	76.89	81.55
	3	91.35	91.21	**92.47**	75.47	75.92	**79.94**
	4	86.36	86.02	x	81.36	81.36	x
		Sencitivity			***Correlation Coefficient***		
	k	DMM	MM	SVM	DMM	MM	SVM
Peri	2	63.76	69.99	**94.04**	0.5	0.54	**0.76**
	3	43.66	46.23	**62.97**	0.54	0.55	**0.6**
	4	70.79	70.36	x	0.52	0.51	x
Extr	2	66.17	**78.5**	60.74	**0.72**	0.71	0.64
	3	58.32	**60**	57.01	0.7	0.7	0.7
	4	73.84	75.33	x	0.5	0.5	x
Ctpl	2	**98.78**	96.74	94.04	0.77	0.77	0.77
	3	**98.49**	98.08	91.11	0.79	0.79	**0.82**
	4	88.6	88.1	x	0.68	0.68	x

convergence of the gradient and were set (based on some exploratory runs) to 0.01 and 0.5 respectively. When training the SVM, we have used the counts of the k-grams of the sequences as the input featues and linear kernel, which is equivalent to using the sequence kernel proposed by Leslie et. al. [12]. The results for SVM 4-grams are not presented because in this case we need 20^4 input features leading to excessively slow run times.

1) Kinase dataset: The results for the Kinase dataset are summarized in Table II.

For $k=2$ there is a 2% gain in accuracy for the DMM($k-1$) model over MM($k-1$), with neither outperforming the other for $k=3,4$. The performance of DMM($k-1$) in this case is comparable with that of the SVM, and for $k=3$, both DMM($k-1$) and MM($k-1$) outperform the SVM.

2) Subcellular Localization dataset: In our experiments with the subcellular localization classification task, to facilitate direct comparisons with previous studies [4] [11] the task is set up as that of learning a set of binary classifiers (one for each class).

Tables III and IV summarize the performance of the generative and discriminative models and the SVM.

With $k=2$, for DMM($k-1$), we observe a small (1%) improvement in accuracy for Extracellular class over

TABLE IV

ACCURACY, SENSITIVITY, SPECIFICITY AND CORRELATION COEFFICIENT ON EUKARYOTIC DATASET OBTAINED BY 10-FOLD CROSS-VALIDATION, OF THE DISCRIMINATIVE, GENERATIVE AND SVM APPROACHES

		Accuracy			***Specificity***		
Class	k	DMM	MM	SVM	DMM	MM	SVM
Ctpl	2	77.72	73.84	**81.87**	77.22	69.76	**90.99**
	3	86	85.9	**88.01**	91.28	90.76	**93.2**
	4	89.29	88.96	x	97.76	97.41	x
Extr	2	92.13	89.7	**92.45**	**98.76**	93.82	98.47
	3	94.19	94.15	**95.49**	99.95	99.8	99.33
	4	95.59	95.67	x	99.8	99.7	x
Mit	2	82.37	75.44	**88.46**	83.76	76.83	**97.86**
	3	89.74	89.67	**91.14**	**98.2**	98.2	97.86
	4	90.44	90.56	x	98.6	98.6	x
Nclr	2	80.55	83.48	**85.74**	71.88	90.45	90.45
	3	87.52	87.97	**89.25**	92.4	92.48	92.03
	4	88.55	88.13	x	87.07	87.7	x
		Sencitivity			***Correlation Coefficient***		
	k	DMM	MM	SVM	DMM	MM	SVM
Ctpl	2	78.94	**84.21**	58.63	**0.52**	0.49	**0.53**
	3	72.51	73.68	73.78	0.65	0.65	**0.69**
	4	67.69	67.4	x	0.73	0.72	x
Extr	2	49.23	**63.08**	53.54	0.56	0.45	**0.63**
	3	56.92	57.53	**70.77**	0.72	0.72	**0.79**
	4	70.77	69.54	x	0.81	0.8	x
Mit	2	**73.2**	66.36	26.79	**0.45**	0.32	0.37
	3	34.27	33.01	**47.04**	0.46	0.45	**0.56**
	4	36.76	37.69	x	0.5	0.51	x
Nclr	2	**91.07**	75.23	**80.04**	0.63	0.67	**0.71**
	3	81.59	82.5	**85.87**	**0.75**	0.75	0.71
	4	**90.33**	88.61	x	0.77	0.76	x

MM($k-1$), and 1.5% gain in accuracy over the SVM 2-grams. In other cases the performance of DMM($k-1$) is similar to that of the SVM with respect to different performance measures. In the case of Eukaryotic data set, DMM($k-1$) substantially outperforms MM($k-1$): For $k=2$, DMM($k-1$) has 4%, 2.5% and 7% higher accuracy (respectively) than MM($k-1$) on Cytoplasmic, Extracellular and Mitochondrial classes. In the case of the Cytoplasmic class, SVM outperforms DMM($k-1$) by about 4% in accuracy. However the correlation coefficient is similar in both cases (0.52 and 0.53) and MM($k-1$) has better sensitivity than both DMM($k-1$) and SVM. In the case of Extracellular class, the accuracy and specificity of both SVM and DMM($k-1$) are comparable, with MM($k-1$) having higher sensitivity and SVM higher correlation coefficient. On the Mitochondrial class, the accuracy of SVM is 6% higher that that of the DMM($k-1$), however DMM($k-1$) substantially outperforms SVM in terms of correlation coefficient (0.45 versus 0.37) and sensitivity (73.2% versus 26.79%). Finally, on the Nuclear class, SVM has higher accuracy by 2%, and higher correlation coefficient (0.71 versus 0.63 for the DMM($k-1$) model). In summary, on the subcellualr localization task, the DMM($k-1$) model shows better overall performance relative MM($k-1$) and DMM($k-1$) is competitive with SVM.

3) Reuters data: The results of generative and discriminative MM($k-1$) are presented in Table V for $k=2$. (We have not included the results for the SVM with $k=2$ since the number of 2-gram counts needed to be input to SVM is 300^2 which makes both space and run time needs of

TABLE V

ACCURACY, SENSITIVITY, SPECIFICITY AND CORRELATION COEFFICIENT ON REUTERS 10 CATEGORIES FOR MM($k-1$) AND DMM($k-1$)

	Accuracy		***Sencitivity***	
data	DMM	MM	DMM	MM
acq	95.27	**95.82**	**89.53**	88.01
corn	98.21	98.21	64.29	67.86
crude	**97.35**	97.13	**90.86**	**89.24**
earn	96.59	**97.86**	97.87	95.83
grain	97.49	97.5	**89.86**	85.81
interest	97.5	97.4	**51.1**	47.32
money-fx	97.07	97.34	**82.12**	79.89
ship	98.86	98.9	**85.06**	79.31
trade	97.5	98.01	**80.17**	75.86
wheat	98.13	98.25	70.42	**76.06**
	Specificity		***Correlation Coefficient***	
data	DMM	MM	DMM	MM
acq	96.87	**98**	0.86	**0.87**
corn	98.91	98.7	0.55	**0.56**
crude	97.75	97.6	0.79	0.77
earn	95.96	**98.87**	0.92	**0.95**
grain	97.85	**98.1**	**0.76**	0.75
interest	99.4	99.5	**0.63**	0.61
money-fx	97.93	**98.35**	0.74	0.75
ship	99.24	99.49	0.8	0.8
trade	98.13	**98.8**	0.69	**0.72**
wheat	98.75	98.75	0.62	**0.65**

our current SVM implementation rather prohibitive). As seen from Table V, the accuracies of DMM($k-1$) and MM($k-1$) are comparable, with the MM($k-1$) slightly outperforming the DMM($k-1$). However, in terms of other performance measures e.g., sensitivity, we observe that for most of the categories DMM($k-1$) outperforms MM($k-1$), and that for categories "interest" and "grain" DMM($k-1$) has a higher correlation coefficient as well.

V. SUMMARY AND DISCUSSION

A. Summary and Conclusion

In this paper, we have described a discriminative approach to the training of sequence classifiers and provided empirical exploration of generative and discriminative approaches for classification in biological and text domains. The proposed classifier – DMM($k-1$) – is trained to maximize the conditional likelihood function of the data based on the likelihood function of Markov Model of order $k-1$. The algorithm for training a DMM($k-1$) classifier initializes the parameters of the model based on the parameters of its generative counterpart – MM($k-1$) – and iteratively updates the parameters using gradient ascent to maximize the conditional likelihood. We compared the performance of DMM($k-1$), MM($k-1$), and SVM sequence classifiers on protein function prediction, subcellular localization prediction, and text categorization tasks.

As the experimental results show, DMM($k-1$) marginally outperforms MM($k-1$) in the human kinase data set in terms of accuracy. On the Eukaryotic protein cellular localization task, DMM($k-1$) outperforms MM($k-1$) by a much wider margin (2.4% to 7%) in terms of accuracy on several classes. In these cases, the performance of the DMM($k-1$) is comparable to that of SVM. We have also observed cases in which MM($k-1$) outperforms DMM($k-1$) (e.g., for Nuclear class) or SVM outperforms DMM($k-1$) (e.g., for the Mitochondrial class). However, SVM is outperformed by DMM($k-1$) in terms of sensitivity by a wide margin on the Mitochondrial class. In summary, on the subcellualr localization task, the DMM($k-1$) shows better overall performance relative MM($k-1$) and DMM($k-1$) is competitive with SVM. On the text classification task, generative and discriminative MM($k-1$) models had similar accuracy. But it is worth noting that the discriminative model exhibited higher sensitivity, whereas the generative model had higher specificity. In complex classification tasks, no single performance measure captures all relevant aspects of classifier performance. Hence it is useful to consider multiple performance criteria in evaluating learning algorithms [3]. Furthermore, in specific applications, there often arises a need to trade off some performance measures against others.

MM($k-1$) model has the advantage of training a classifier in one pass through the training data. It also lends itself to incremental update as new training data become available - without the need to revisit previously processed training data. In contrast, DMM($k-1$) requires several passes through the training data and is less amenable to incremental update although in an online mode of training, incremental update of parameters is feasible although optimality of the parameter estimates (with respect to correspondence with local maximum of the conditional likelihood function) cannot be guaranteed without the ability to revisit previously processed training data. SVM algorithm relies on computationally expensive optimization that does not lend itself to incremental update without revisiting previously processed training data. On the other hand, SVM is more sophisticated than MM($k-1$) model in terms of controlling the complexity of the classifier to avoid overfitting. The results presented in this paper suggest that a relatively small number of iterations in discriminative training of DMM($k-1$) model initialized with the parameters of the corresponding MM($k-1$) can often result in performance that is comparable to that of SVM on several sequence classification tasks.

B. Related Work

The sequence classifiers introduced in this paper are the discriminative counterparts of generative models - namely, Markov Model($k-1$) introduced by Peng et. al. [16] which are equivalent to Naïve Bayes(k) model introduced by Andorf et. al. [1] and shown to outperform Naïve Bayes in sequence-based protein function classification [1], as well as text classification [16], when sufficient training data are available.

Rubinstein and Hastie [21] presented a general approach for obtaining a discriminative model given a generative model. They have also analyzed Naïve Bayes and its discriminative counterpart Generalized Additive Model.

Ng and Jordan [15] analyzed the dependence between the performance of generative and discriminative model and the size of the data used for training. They used Naïve Bayes and Linear Regression to support the results obtained by theoretical analysis. They have observed that with limited data, the discriminative models may have lower accuracy than the generative models.

McCallum [13] described discriminative probabilistic graphical models in the form of Conditional Random Fields (CRFs) for speech recognition. Taskar et. al. [22] presented Discriminative Probabilistic Models for relational data by using conditional Markov network built over relational data.

Grossman and Domingos [10] have described an approach for learning parameters that maximize conditional likelihood function in Bayesian Networks instead of the likelihood function. They showed that the resulting model is more accurate than its generative counterpart. They noted that optimization of the conditional likelihood function is computationally infeasible due to the complexity of structure search.

Roose et. al. [20] have recently shown that for special cases of network structures, the discriminative training of Bayesian Networks is equivalent to logistic regression problem. Markov Model of order $k-1$ examined in this paper is a special case of a Bayesian Network, with additional topological structure. This eliminates the need for structure search, yielding gradient based algorithm for optimizing the parameters of the resulting sequence classifier introduced in this paper. The discriminative sequence classifier is initialized with the parameters obtained from the generative model (which can be estimated in a single pass through the training data). Further training tunes the parameters to improve the conditional likelihood.

Finally, there is a growing interest in hybrid models. Hybrid models are the models which combine generative and discriminative learning. Bouchard and Triggs [5] have proposed a Generative-Discriminative Trade-off method (GTD) as a way of combining general classes of generative and discriminative classifiers. Raina et. al. [18] proposed a model that combines Naïve Bayes Multinomial model and Logistic Regression, and showed that their hybrid model has lower error rate than either Naïve Bayes or Logistic Regression.

C. Future Work

Ongoing research is aimed at:

1) Development of sophisticated regularization methods (e.g., maximizing the margin of separation between classes) to avoid overfitting on the training data in discriminative training.
2) Exploration of ways to synergistically exploit the strengths of discriminative as well as generative models for sequence classification.
3) Exploration of extensions of the algorithms proposed in this paper to higher dimensional dependency structures (e.g. 2-dimensional images as opposed to 1-dimensional strings).
4) Applications of the resulting sequence classification methods in bioinformatics and related applications.

Acknowledgment

Research supported in part by grant from the National Institutes of Health (GM 066387) to Vasant Honavar.

References

[1] C. Andorf, A. Silvescu, D. Dobbs, and V. Honavar. Learning classifiers for assigning protein sequences to gene ontology (GO) functional families. In *Proceedings of the Fifth International Conference On Knowledge Based Computer Systems (KBCS)*, 2004.

[2] R. D. Appel, A. Bairoch, and D. F. Hochstrasser. A new generation of information retrieval tools for biologists: the new example of ExPASy www server. *Trenchs Biochem. Sci*, 19:258–260, 1994.

[3] P. Baldi and S. Brunak. *Bioinformatics: The Machine Learning Approach*. MIT Press, Cambridge, MA, 1998.

[4] M. Bhasin and G. P. S. Raghava. ESLpred: SVM-based method for subcellular localization of eukaryotic proteins using dipeptide composition and PSI-BLAST. *Nucleic Acids Research*, 32, 2004.

[5] G. Bouchard and B. Triggs. The trade-off between generative and discriminative classifiers. In *Proceedings to IASC International Symposium on Computational Statistics (CompStat)*, 2004.

[6] E. Charniak. *Statistical Language Learning (Language, Speech, and Communication)*. MIT Press, 1996.

[7] B. Y. M. Cheng, J. G. Carbonell, and J. Klein-Seetharaman. Protein classification based on text document classification techniques. *Proteins*, 1(58):955–70, 2005.

[8] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. Probabilistic networks and expert systems. *Springer*, 1999.

[9] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *CIKM '98: Proceedings of the seventh international conference on Information and knowledge management*, pages 148–155. ACM Press, 1998.

[10] D. Grossman and P. Domingos. Learning bayesian network classifiers by maximizing conditional likelihood. In *Proceedings to the 21st International Conference On Machine Learning (ICML)*, 2004.

[11] S. Hua and Z. Sun. Support vector machine approach for protein subcellular localization prediction. *Bioinformatics*, 17(8):721–8, 2001.

[12] C. Leslie, E. Eskin, J. Weston, and W. Noble. Mismatch string kernels for SVM protein classification. In *Advances in Neural Information Processing Systems (NIPS) 15*, 2002.

[13] A. McCallum. Efficiently inducing features of conditional random fields. In *Proc. of Conference on Uncertainty in Artificial Intelligence (UAI)*, 2003.

[14] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *AAAI-98 Workshop on "Learning for Text Categorization"*, 1998.

[15] A. Ng and M. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. 2002.

[16] F. Peng, D. Shuurmans, and S. Wang. Augmenting naive bayes classifier using statistical n-gram language modeling. *Information Retrieval*, 7(3-4):317–345, 2004.

[17] N. Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12:145–151, 1999.

[18] R. Raina, Y. Shen, A. Ng, and A. McCallum. Classificatin with hybrid generatve/discriminative models. In *Advances in Neural Information Processing Systems (NIPS) 16*, 2004.

[19] A. Reinhardt and T. Hubbard. Using neural networks for prediction of the subcellular location of proteins. *Nucleic Acids Research*, 26(9):2230–2236, May 1998.

[20] T. Roos, H. Wettig, P. Grünwald, P. Mullymäki, and H. Tirri. On discriminative bayesian network classifiers and logistic regression. *Machine Learning*, (59):267–296, June, 2005.

[21] D. Rubinstein and T. Hastie. Discriminative vs informative learning. In *Proceedings of Knowledge Discovery and Data Mining (KDD)*, 1997.

[22] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proc. of Eighteenth Conference On Uncertainty in Artificial Intelligence (UAI02)*, Edmonton, Canada, 2003.

[23] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, New York, 1995.

[24] C. Yan, D. Dobbs, V. Honavar, and D. Dobbs. A two-stage classifier for identification of proteinprotein interface residues. *Bioinfromatics*, 20(S1):i371–i378, 2004.

[25] Z. Yuan. Prediction of protein subcellular locations using Markov chain models. *FEBS Letters*, 451(1):23–6, 1999.