Maximum Margin Hidden Markov Models for Sequence Classification

Nikolaus Mutsam, Franz Pernkopf

Graz University of Technology, Laboratory of Signal Processing and Speech Communication, Graz, Austria

ABSTRACT

Discriminative learning methods are known to work well in pattern classification tasks and often show benefits compared to generative learning. This is particularly true in case of model mismatch, i.e. the model cannot represent the true data distribution. In this paper, we derive discriminative maximum margin learning for hidden Markov models (HMMs) with emission probabilities represented by Gaussian mixture models (GMMs). The focus is on single-label sequence classification where the margin objective is specified by the probabilistic gap between the true class and the most competing class. In particular, we use the extended Baum-Welch (EBW) framework to optimize this probabilistic margin embedded in a hinge loss function. Approximations of the margin objective and the derivatives are necessary. In the experiments, we compare maximum margin HMMs to generative maximum likelihood and discriminative conditional log-likelihood (CLL) HMM training. We present results of classifying trajectories of handwritten characters, Australian sign language data, digits of speech data and UCR time-series data. Maximum margin HMMs outperform in many cases CLL-HMMs. Furthermore, maximum margin HMMs achieve a significantly better performance that generative maximum likelihood HMMs.

1. Introduction

The analysis of time-series or sequential data covers a wide field of applications, such as for instance speech analysis [1], financial mathematics [2], and weather forecasts [3]. In a time-series the samples are dependent on previous samples in the sequence. Modelling the dependency on all previous samples is computationally intractable, therefore usually only the neighboring context is modeled. In the simplest case, only the most recent sample is considered leading to the first-order Markov model assumption. One of the simplest and most widely used models for time-series processing in the past decades is the HMM.

There are two classical learning paradigms in the machine learning community: generative learning and discriminative learning [4]. Generative learning aims to recover the data distribution from a finite set of samples. This is achieved by optimizing the data-likelihood or the data-posterior-probability, in the case of a model-prior for regularization. Maximum likelihood estimation (MLE) is usually used to generatively learn the classifier. Generatively optimized models facilitate to generate samples by the model having the same statistical distribution as the training data. Discriminative learning methods such as conditional log-likelihood (CLL) [5,6] or maximum margin learning [7,8] more directly represent aspects that are important for classification accuracy, i.e. a prediction function is optimized, predicting output variables (classes) from a set of input variables (features). As shown in [9], generative learning of naive Bayes models reach faster their asymptotic generalization performance with respect to samples size compared to discriminative training. If the model does not match the true underlying distribution, discriminatively learned models usually obtain better asymptotic performance for a sufficiently large set of training samples.

Discriminative training of HMMs celebrated success over the years in speech processing and in this context the extended
Baum-Welch optimization algorithm [Gopalakrishnan et al. (1991)] has been introduced. In the seminal work of [Bahl et al. (1986)], discriminative HMM parameter learning based on the maximum mutual information (MMI) criterion has been proposed. The goal is to maximize the posterior probability of the transcriptions given the speech utterances. This results in significantly better recognition rates compared to conventional generative MLE learning. A decade later, the minimum classification error (MCE) has been proposed [Juang et al. (1997)] where the aim is to minimize the sentence classification error, i.e., risk, on the training set. The main advantage of MCE is that in MMI training only the posterior distribution is optimized, which does not necessarily result in an optimal classification performance. MCE optimizes the empirical risk and is therefore susceptible to overfitting. The expected risk on unseen test data also depends on the generalization ability of the model. From the SVM literature [Schölkopf & Smola (2001); Burges (1998)] it is well-known that the optimization of the margin leads to good generalization properties cf. Vapnik-Chervonenkis (VC) dimension and PAC bounds [Vapnik (1998)]. Hence, margin optimisation has been proposed for HMMs in [Taskar et al. (2004); Sha & Saul (2007a); Saon & Povey (2008); Heigold et al. (2008, 2010, 2012)]. Most of this work is in the area of speech processing, e.g., phoneme recognition, with focus on multi-label sequence classification, i.e., the observation sequence is assigned to multiple labels, i.e., a label sequence. This usually leads to a margin objective [Sha & Saul (2007b)] composed of two terms:

1. A probabilistic term measuring the gap between log-probabilities of the data of the target label sequence and the most competing sequence;
2. A term measuring the Hamming distance between two label sequences, the target label sequence and the competing label sequence.

In this objective, the gap between the log-probabilities in the first term should be larger or equal to the Hamming distance of the second term. In [Heigold et al. (2010, 2012)], the margin term has been incorporated in a unified training criterion. Special cases of this criterion are MMI or MCE amongst others.

In contrast, we perform single-label classification of sequences, i.e., the observation sequence is assigned to a single class label. We derive discriminative maximum margin learning for HMMs with emission probabilities represented by GMMs for single-label sequence classification. Related to the margin criterion for multi-label classification from above the distance measure between the competing and the target/true label sequences is neglected. Here, we use the probabilistic definition of the margin [Guo et al. (2005); Pernkopf et al. (2012)], i.e., the first term in the objective above. The log-probability of the sequence of the true class should lie at least some distance away from the log-probabilities of the competing classes, i.e., the second term in the objective above is set to a constant. Our probabilistic margin is embedded in a hinge-loss function and optimized by the EBW algorithm. The EBW algorithm uses the derivatives of the objective functions. Robust approximations to the derivatives are discussed. During optimization using the EBW algorithm the sum-to-one constraint of the probability distributions of the HMM is maintained. Hence, the parameters of the HMM are still ’normalized’ probability distributions. This has the advantage that summing over missing variables is still possible as we show for Bayesian network classifiers in [Pernkopf et al. (2012)]. This is in contrast to [Kim & Pavlovic (2011)] where also single-label classification is considered. Furthermore, they approximate the objective to obtain a convex optimization problem solved in a similar way as in [Sha & Saul (2007b)].

Generative pre-training is important in many discriminatively learned models [Erhan et al. (2010); Bahl et al. (1986); Pernkopf et al. (2012)]. This can be seen as a form of regularization of the discriminative learning objective. Therefore, we initialize discriminative HMM training with the MLE solution. Furthermore, we use early stopping during discriminative optimization. Hence, the discriminative parameters are partially reflecting the MLE solution. Experimental results for frequently used time-series data such as handwritten characters, Australian sign language, digits of speech data and UCR time-series are provided. In all cases, maximum margin HMMs lead to competitive performance compared to CLL-HMMs and generative HMMs. Maximum margin HMMs mostly outperform the CLL-HMMs and the generatively optimized ME-HMMs in terms of classification rate in all experiments.

The paper is organized as follows: In Section 2 we shortly review the Bayesian classifier for sequential data and introduce the HMM and the notation. In Section 3 we present conventional parameter estimation techniques for HMMs such as MLE and CLL optimization are summarized. Maximum margin parameter estimation for HMMs is derived in Section 4 using the EBW algorithm. This section also includes approximations of the derivatives necessary for the EBW method. In Section 5, we present experimental results. Section 6 concludes the paper.

2. Bayesian Classifier for Sequential Data

The task of classification is to assign a given observation sequence \( \mathbf{x} = (x_1, \ldots, x_T) \) to a class \( c \in \{1, \ldots, C\} \), where \( C \) is the number of classes, \( x_t \in \mathbb{R}^D \), \( D \) is the number of observations at time \( t \) and \( T \) is the length of the sequence. According to Bayes’ rule, the class posterior \( p(c|x) \) is given by

\[
p(c|x) = \frac{p(x|c)p(c)}{p(x)} = \frac{p(x|c)p(c)}{\sum_{c'} p(x|c')p(c')},
\]

where the likelihood term is assumed to be a parametric model for class \( c \), i.e., \( p(x|c) = p(x|\Theta_c) \). In particular, we use an HMM \( \Theta_c \) for each class. The class label can be determined by the maximum a-posteriori (MAP) estimate, i.e., the most likely class label \( c^* \) is determined using the class posteriors as

\[
c^* = \arg \max_{1 \leq c \leq C} p(c|x) = \arg \max_{1 \leq c \leq C} \{p(x|\Theta_c)p_c\},
\]
where the denominator of \( \frac{1}{N} \) can be neglected since it only scales \( p(x|c) \). The term \( p_c = p(c) \) is the class prior distribution.

The probability \( p(x|\Theta_c) \) can be efficiently determined by using either the forward or the backward procedure [Rabiner (1989)]. If the most probable state sequence \( Q^* = [q_1^*, \ldots, q_T^*] \) producing \( x \) is known or estimated by the Viterbi algorithm, \( p(x|\Theta_c) \) can also be approximated by \( p^*(x|\Theta_c) \), i.e., the product of the prior, the observation, and transition probabilities along the most probable path \( Q^* \) of HMM \( \Theta_c \).

An HMM can be fully described by two stochastic processes. The first is a Markov process that produces a sequence of not directly observable states \( Q = \{q_1, \ldots, q_t, \ldots, q_T\} \) where \( q_t \in \{1, 2, \ldots, S\} \) and \( S \) is the number of states. The second process produces an observation \( x_t \) at every time step \( t \) of the sequence according to a state-dependent observation probability distribution \( b_t(x_t) = p(x_t|q_t = i) \). In a first-order HMM, the state of variable \( q_t \) depends on the state of the previous variable \( q_{t-1} \), i.e., the transition from state \( q_{t-1} = i \) to state \( q_t = j \) occurs with a certain probability denoted by \( a_{i,j} = p(q_t = j|q_{t-1} = i) \), where \( \sum_{j=1}^{S} a_{i,j} = 1, \forall i \in \{1, \ldots, S\} \). The transition matrix \( A \) collects all transition probabilities \( a_{i,j} \). The probability of being in a hidden state \( i \) at the beginning of a sequence \( t = 1 \) is modeled by the state prior distribution \( \pi_i = p(q_1 = i) \). We use a multivariate GMM to model the observation probabilities, i.e., we have a sum of \( M \) weighted Gaussians \( N(x_t; \mu_{i,m}, \Sigma_{i,m}) \).

\[
b_t(x_t) = p(x_t|q_t = i) = \sum_{m=1}^{M} \alpha_{i,m} N(x_t; \mu_{i,m}, \Sigma_{i,m}),
\]

where \( \alpha_{i,m} \) are the weights of each Gaussian component, \( 0 \leq \alpha_{i,m} \leq 1 \) and \( \sum_{m=1}^{M} \alpha_{i,m} = 1 \), and \( \mu_{i,m} \in \mathbb{R}^D \) is the \( D \)-dimensional mean vector and \( \Sigma_{i,m} \) is the \( D \times D \) covariance matrix. We assume a diagonal covariance throughout the paper. An HMM is fully specified by the state prior distribution \( \pi_i \), the transition matrix \( A \) and the emission probability \( b_t(x_t) \). These parameters are collected for HMM of class \( c \) in \( \Theta_c = \{\pi_i, a_{i,j}, b_{i,m}, \mu_{i,m}, \Sigma_{i,m}|i \in \{1, \ldots, S\}, m \in \{1, \ldots, M\}\} \).

3. Conventional Parameter Estimation of HMMs

Commonly, HMM parameters are determined iteratively using MLE [Rabiner (1989); Pernkopf et al. (2014)]. Discriminative parameter optimization using conditional log-likelihood learning, i.e., the class posterior of the model is maximized, has been introduced in [Bahl et al. (1986)] as MMI training. Both objectives for parameter estimation are introduced in the following.

Formally, MLE parameters for the model of class \( c \) are learned as

\[
\Theta_c^{MLE} = \arg \max_{\Theta_c} p(x_c|\Theta_c) = \prod_{n=1}^{N_c} p(x^n|\Theta_c),
\]

where \( x_c = \{x^1, x^2, \ldots, x^{N_c}\} \) is a set of \( N_c \) training sequences belonging to class \( c \). MLE of the HMM parameters leads to an iterative scheme such as the expectation-maximization (EM) algorithm also known as Baum-Welch algorithm for HMMs [Rabiner (1989)].

In contrast to generative methods, discriminative training of an HMM \( \Theta_c \) of class \( c \) involves all training samples \( X = \{X_1, X_2, \ldots, X_C\} \), where \( N = \sum_{c=1}^{C} N_c \). The conditional log-likelihood (CLL) is given as

\[
CLL(X|\Theta_c) = \log \prod_{n=1}^{N} p(x^n|\Theta_c) = \log \prod_{c=1}^{C} \sum_{n=1}^{N} p(x^n|\Theta_c)p_c
\]

and the parameters are determined according to

\[
\Theta_c^{CLL} = \arg \max_{\Theta_c} CLL(X|\Theta_c),
\]

where \( \Theta = \{\Theta_1, \rho_1, \ldots, \Theta_C, \rho_C\} \) and \( c^n \) denotes the class of sequence \( x^n \). Maximizing the CLL criterion is closely related to MMI estimation [Bahl et al. (1986); Normandin & Mgera (1997); Normandin et al. (1994); Woodland & Povey (2002)]. The CLL can be maximized by gradient-based optimization methods, e.g., the EB algorithm.

4. Maximum Margin Parameter Estimation

The multi-class margin [Guo et al. (2005); Pernkopf et al. (2012) of sample \( n \) is

\[
d_m^n = \min_{c \neq c^n} \frac{p(c^n|x^n, \Theta_c)}{p(c|x^n, \Theta_c)} = \min_{c \neq c^n} \frac{p(c^n, x^n|\Theta_c)}{p(c, x^n|\Theta_c)}
\]

\[
\frac{1}{\max_{c \neq c^n} p(x^n|\Theta_c)}
\]

If \( d_m^n > 1 \), then sample \( n \) is correctly classified and vice versa.

We replace the max operator by the differentiable approximation \( \max_{\eta} f(\eta) \approx \left[ \sum f(\eta) \right]^\frac{1}{\eta} \), where \( \eta > 1 \) and \( f(\eta) \) is non-negative. In the limit of \( \eta \rightarrow \infty \) the approximation converges to the maximum operator. Replacing the maximum with its approximation, we obtain

\[
d_m^n = \frac{p(x^n|\Theta_c)}{\left( \sum_{c \neq c^n} p(x^n|\Theta_c) \right)^\frac{1}{\eta}}
\]

Usually, the maximum margin approach maximizes the margin of the sample with the smallest margin, i.e., \( \min_{n=1,\ldots,N} d_m^n \) for a separable classification problem [Schölkopf & Smola (2001)]. We aim to relax this by introducing a soft margin, i.e., we focus on samples with a \( d_m^n \) close to one. Therefore, we consider the hinge loss function according to

\[
J^\kappa(\chi|\Theta) = \prod_{n=1}^{N} \min_{\kappa, d_m^n}
\]

where parameter \( \kappa > 1 \) controls the influence of the margin \( d_m^n \) in the hinge loss \( J^\kappa(\chi|\Theta) \) and is set by cross-validation. Maximizing this function with respect to the parameters \( \Theta \) implicitly means to increase the margin \( d_m^n \) whereas the emphasis is on
samples with a margin $d^n_\Theta < \kappa$, i.e. samples with a large positive margin have no impact on the optimization. Maximizing $J(X(\Theta))$ via EBW or gradient descent is not straightforward due to the discontinuity in the derivative at $d^n_\Theta = \kappa$. Therefore, we propose to use for the hinge function $h(y) = \min\{x, y\}$ a smooth hinge function which enables a smooth transition of the derivative and has a similar shape as $h(y)$. We propose the following function inspired by the Huber loss [Huber, 1964]. In particular, we approximate the discontinuity by a circle segment as

$$h(y) = \begin{cases} 
  y + \frac{1}{2}, & \text{if } y \leq \kappa - 1 \\
  \kappa - \frac{1}{2}(y - \kappa)^2, & \text{if } \kappa - 1 < y < \kappa \\
  \kappa, & \text{if } y \geq \kappa
\end{cases}$$

which requires to divide the data $X$ into three partitions depending on $y = d^n_\Theta$, i.e. $X^1$ contains samples where $d^n_\Theta \leq \kappa - 1$, $X^2$ consists of samples with a margin in the range $\kappa - 1 < d^n_\Theta < \kappa$, and $X^3 = X \setminus \{X^1 \cup X^2\}$. The smooth hinge function is illustrated in [Pernkopf et al., 2012].

Basically, there are other smoothing techniques available for non-smooth convex objectives, e.g. [Nesterov, 2005]. In our case, smoothing of the objective function makes it amenable for gradient-based optimization methods while still approximating the original objective well. Experiments using a similar parametrized smooth hinge function show only a slight influence on performance for maximum margin Bayesian network classifiers [Pernkopf et al., 2012]. Furthermore, a similar approximation of the maximum margin objective outperforms a convex formulation (which requires relaxation of constraints) with respect to computational requirements, while the classification performance is almost identical.

Using the smooth hinge function in (10), our objective function for margin maximization is

$$J(X(\Theta)) = \prod_{n=1}^{N} f^n_{\Theta} = \left\{ \prod_{n \in X^1} \left( d^n_\Theta + \frac{1}{2} \right) \prod_{n \in X^2} \left( \kappa - \frac{1}{2} (d^n_\Theta - \kappa)^2 \right) \right\} \cdot 1^{X^3},$$

4.1. Optimization of the Margin Objective

The EBW algorithm (more details are given in Appendix A) is an iterative procedure which can be used to optimize rational functions [Gopalakrishnan et al., 1991]. We use the EBW framework to optimize the margin objective in (11) for the discrete model parameters $\rho_c, \pi_c, a_c, i, j, a_{c,i,j,m}$. The parameter re-estimation equation of the form

$$\theta^i_j \leftarrow \theta^i_j \left( \frac{\partial \log J(X(\Theta))}{\partial \theta^i_j} + D \right) / \sum_j \theta^i_j \left( \frac{\partial \log J(X(\Theta))}{\partial \theta^i_j} + D \right),$$

is used, where $\theta^i_j \geq 0$, $\sum_j \theta^i_j = 1$, and $j$ indicates a particular discrete variable. EBW requires the partial derivative $\frac{\partial \log J(X(\Theta))}{\partial \theta^i_j}$ and $D$. Both terms are provided in the sequel. Specifically, the derivative $\frac{\partial \log J(X(\Theta))}{\partial \theta^i_j}$ for the re-estimation equation (12) of the EBW algorithm is

$$\frac{\partial \log J(X(\Theta))}{\partial \Theta} = \sum_{n=1}^{N} s^n \frac{\partial \log d^n_\Theta}{\partial \Theta}$$

where $s^n$ denotes a sample dependent weight given as follows:

$$s^n = \begin{cases} 
  \frac{d^n_\Theta}{d^n_\Theta + p_\Theta}, & \text{if } n \in X^1 \\
  \frac{\kappa (d^n_\Theta - \kappa)^2}{\kappa (d^n_\Theta - \kappa)^2}, & \text{if } n \in X^2 \\
  0, & \text{if } n \in X^3
\end{cases}$$

Approximating $p(x|\Theta_c)$ with the probability of the most probable state sequence of the Viterbi algorithm, i.e.

$$p(x|\Theta_c) \approx p^v(x|\Theta_c) = \pi_{c,i,j} b_{c,i,j}(x_1) \prod_{t=2}^{T} a_{c,i,j-1} b_{c,i,j}(x_t),$$

the log of the margin $d^n_\Theta$ of sample $x^n$ in Eq. (8) decomposes to

$$\log d^n_\Theta = \log(p(x^n|\Theta_c, \rho_c)) - \frac{1}{\eta} \sum_{c^n \in \pi_c} (p(c^n|\Theta_c, \rho_c))^\eta,$$

$$= \log p_{c,i,j} \sum_{t=1}^{T} a_{c,i,j} b_{c,i,j}(x_t) \prod_{t=2}^{T} a_{c,i,j-1} b_{c,i,j}(x_t),$$

where $p_{c,i,j}^{\eta}$ is the most probable state of the HMM of class $c^n$ for a sequence $x^n$ at time $t$. The derivative for $\rho_c$ of $\frac{\partial \log d^n_\Theta}{\partial \rho_c}$ in (13) is

$$\frac{\partial \log d^n_\Theta}{\partial \rho_c} = \frac{\|_{c^n \in \pi_c}}{\|_{c^n \in \pi_c}} \|_{c^n \in \pi_c} (p(x^n|\Theta_c, \rho_c))^\eta,$$

$$\frac{\partial \log d^n_\Theta}{\partial \rho_c} = \frac{\|_{c^n \in \pi_c}}{\|_{c^n \in \pi_c}} \|_{c^n \in \pi_c} (p(x^n|\Theta_c, \rho_c))^\eta,$$

$$\frac{\partial \log d^n_\Theta}{\partial \rho_c} = \frac{\|_{c^n \in \pi_c}}{\|_{c^n \in \pi_c}} \|_{c^n \in \pi_c} (p(x^n|\Theta_c, \rho_c))^\eta,$$

Symbol $\|_{c^n \in \pi_c}$ denotes the indicator function (i.e. equals 1 if the Boolean expression $i = j$ is true and 0 otherwise).

Furthermore, the partial derivatives of $\frac{\partial \log d^n_\Theta}{\partial \Theta}$ with respect to $\pi_c, a_{c,i,j}$ and $a_{c,i,j,m}$ are given as follows:

$$\frac{\partial \log d^n_\Theta}{\partial \pi_c} = \frac{1}{\|_{c^n \in \pi_c}} \sum_{c^n \in \pi_c} (p(x^n|\Theta_c, \rho_c))^\eta,$$

$$\frac{\partial \log d^n_\Theta}{\partial \rho_c} = \frac{1}{\|_{c^n \in \pi_c}} \sum_{c^n \in \pi_c} (p(x^n|\Theta_c, \rho_c))^\eta,$$

$$\frac{\partial \log d^n_\Theta}{\partial \rho_c} = \frac{1}{\|_{c^n \in \pi_c}} \sum_{c^n \in \pi_c} (p(x^n|\Theta_c, \rho_c))^\eta.$$
where

\[
\begin{align*}
\eta_{c,t}^n &= \eta_{c,:,j = j_t}^n, \\
\theta_{c,t}^n &= \theta_{c,:,j = j_t}^n, \\
\gamma_{c,t}^n &= \gamma_{c,:,j = j_t}^n
\end{align*}
\]

are small parameter values. Merialdo (1988) observed that low-valued parameters may cause a large magnitude of the gradient and the optimization concentrates on those parameters. However, small parameter values indicate that they are rarely used during the production of an observation sequence. Hence, there is not sufficiently training data available for reliably estimating very low probabilities and concentrating on low-valued parameters is unreliable. Therefore, he suggests to focus on modifying better estimated high-valued parameters during optimization by using an approximation of the gradients. In particular, for gradients of the form

\[
\frac{\partial \log d_{\Theta}}{\partial \theta_i^m} \approx \frac{1}{\theta_i^m} (c_{i,j} - c_{j,i}) \tag{29}
\]

This approximation of the gradients has been used for CLL learning in Normandin & Morgera (1991); Normandin et al. (1994). Unfortunately, approximating the gradient by (29) cannot be applied to the derivatives of the margin, because the approximated gradient disappears for any HMM parameter. Therefore, we suggest an alternative approximation in order to obtain reliable parameter updates. Since the unreliability of the updates is caused by small parameter values due to high values of the gradients Merialdo (1988), normalizing the gradient by a sum-to-one constraint of the absolute gradient values keeps the updates reliable. For gradients of the form

\[
\frac{\partial \log d_{\Theta}}{\partial \theta_i^m} \approx \frac{1}{\theta_i^m} (c_{i,j} - c_{j,i})
\]

we propose to approximate the gradient by

\[
\frac{\partial \log d_{\Theta}}{\partial \theta_i^m} \approx \frac{1}{\theta_i^m} (c_{i,j} - c_{j,i}) \tag{30}
\]

The resulting approximations of the derivatives in (17), (21), (22) and (23) are provided in the algorithm for maximum margin (MM) training of HMMs in Appendix B. As an alternative, Woodland and Povey [Woodland & Povey (2002) proposed an alternative mixture weight update rule using an iterative procedure.

4.3. Approximation for the Gaussians

EBW has been formulated for discrete probability distributions. Normandin and Morgera [Normandin & Morgera (1991)] introduced a discrete approximation of the Gaussian distribution assuming diagonal covariance matrices. This leads to the re-estimation equation for \( \mu_{c,i,m} \) and \( \Sigma_{c,i,m} \) given as

\[
\mu_{c,i,m} = \frac{\sum_{n=1}^{N} s_n^m \sum_{t=1}^{T} \gamma^n_{c,t} (u_{c,j}^n - \bar{u}_{c,j}^d - r_{c,j}^d)^2}{\sum_{n=1}^{N} s_n^m \sum_{t=1}^{T} \gamma^n_{c,t}}
\]

and

\[
\Sigma_{c,i,m} = \frac{\sum_{n=1}^{N} s_n^m \sum_{t=1}^{T} \gamma^n_{c,t} \left( u_{c,i}^n - \bar{u}_{c,i}^d - r_{c,i}^d \right) \left( u_{c,i}^n - \bar{u}_{c,i}^d - r_{c,i}^d \right)^T}{\sum_{n=1}^{N} s_n^m \sum_{t=1}^{T} \gamma^n_{c,t}}
\]

And:

\[
\sum_{n=1}^{N} s_n^m \sum_{t=1}^{T} \gamma^n_{c,t} \left( u_{c,i}^n - \bar{u}_{c,i}^d - r_{c,i}^d \right) \left( u_{c,i}^n - \bar{u}_{c,i}^d - r_{c,i}^d \right)^T
\]

where \( \gamma_{c,t}^n \) and \( \theta_{c,i,m} \) are taken element-wise.

4.4. Implementation of the MM-HMM EBW Algorithm

The EBW algorithm converges to a local optimum of \( J(X; \Theta) \) providing a sufficiently large value for \( D \). Setting the constant \( D \) is not trivial. If it is chosen too large then training is slow and if it is too small the update may fail to increase the objective function. In practical implementations heuristics have been suggested [Woodland & Povey (2002); Klautau et al. (2003); Pernkopf & Wohlfahrt (2010)]. In order to obtain positive covariances, the inequality

\[
\frac{v_{c,i,m} + \eta_{c,i,m}^2 - \gamma_{c,i,m}^2}{v_{c,i,m} + \eta_{c,i,m}^2 - \gamma_{c,i,m}^2} \geq 0
\]

must hold for any covariance \( \sigma_{c,i,m} \) of dimension \( d \in D \), where

\[
h_{c,i,m} = \sum_{n=1}^{N} s_n^m \sum_{t=1}^{T} \gamma^n_{c,t} (u_{c,i}^n - \bar{u}_{c,i}^d - r_{c,i}^d)
\]

and

\[
k_{c,i,m} = \sum_{n=1}^{N} s_n^m \sum_{t=1}^{T} \gamma^n_{c,t} (u_{c,i}^n - \bar{u}_{c,i}^d - r_{c,i}^d)
\]

Rearranging (33) leads to a quadratic inequality with respect to \( D \) [Valkchev et al. (1997)]:

\[
\frac{\sigma_{c,i,m} D^2}{a} + (\sigma_{c,i,m} + \sigma_{c,i,m}^2 + 2 \sigma_{c,i,m} \mu_{c,i,m} h_{c,i,m}) D + (\sigma_{c,i,m} h_{c,i,m} - k_{c,i,m}^2) > 0 \tag{36}
\]

We propose to set

\[
D = F \cdot \max(D_1, D_2, D_3)
\]

(37)
Table 1. Classification rates of MLE-HMMs, CLL-HMMs, and MM-HMMs in [%] on data of the Pendigits characters.

<table>
<thead>
<tr>
<th>S</th>
<th>M</th>
<th>CLL-HMM</th>
<th>MM-HMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>90.3 ± 0.98, 94.3 ± 0.77, 93.0 ± 0.84</td>
<td>96.4 ± 0.62, 96.9 ± 0.57, 97.7 ± 0.50</td>
<td>97.0 ± 0.57, 97.4 ± 0.53, 97.9 ± 0.47</td>
</tr>
<tr>
<td>3</td>
<td>92.5 ± 0.87, 92.5 ± 0.87, 93.9 ± 0.80</td>
<td>97.1 ± 0.56, 96.7 ± 0.59, 97.4 ± 0.52</td>
<td>97.8 ± 0.49, 97.9 ± 0.48, 98.4 ± 0.42</td>
</tr>
<tr>
<td>4</td>
<td>92.5 ± 0.87, 93.9 ± 0.79, 95.6 ± 0.68</td>
<td>97.5 ± 0.52, 96.8 ± 0.58, 98.2 ± 0.44</td>
<td>97.4 ± 0.53, 97.9 ± 0.47, 98.6 ± 0.39</td>
</tr>
<tr>
<td>5</td>
<td>93.7 ± 0.80, 94.9 ± 0.73, 94.2 ± 0.77</td>
<td>96.4 ± 0.62, 97.1 ± 0.56, 97.31 ± 0.53</td>
<td>98.2 ± 0.44, 98.3 ± 0.43, 98.8 ± 0.36</td>
</tr>
<tr>
<td>6</td>
<td>94.7 ± 0.74, 94.3 ± 0.77, 95.7 ± 0.67</td>
<td>97.9 ± 0.48, 97.6 ± 0.50, 98.1 ± 0.46</td>
<td>98.3 ± 0.43, 97.9 ± 0.47, 98.5 ± 0.40</td>
</tr>
</tbody>
</table>

Table 2. Classification rates of MLE-HMMs, CLL-HMMs, and MM-HMMs in [%] on data of the Auslan database.

<table>
<thead>
<tr>
<th>S</th>
<th>M</th>
<th>CLL-HMM</th>
<th>MM-HMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>68.1 ± 7.72, 72.9 ± 7.37, 73.6 ± 7.30</td>
<td>68.3 ± 7.71, 87.1 ± 5.54, 80.0 ± 6.63</td>
<td>81.2 ± 6.47, 85.5 ± 5.84, 84.1 ± 6.07</td>
</tr>
<tr>
<td>3</td>
<td>71.2 ± 7.50, 75.0 ± 7.17, 77.9 ± 6.88</td>
<td>86.7 ± 5.63, 86.4 ± 5.67, 80.2 ± 6.60</td>
<td>84.8 ± 5.95, 86.0 ± 5.76, 87.9 ± 5.41</td>
</tr>
<tr>
<td>4</td>
<td>73.3 ± 7.33, 77.4 ± 6.93, 81.2 ± 6.47</td>
<td>86.2 ± 5.71, 78.6 ± 6.80, 80.7 ± 6.54</td>
<td>86.7 ± 5.63, 78.8 ± 6.77, 87.4 ± 5.50</td>
</tr>
<tr>
<td>5</td>
<td>76.2 ± 7.06, 78.1 ± 6.85, 80.5 ± 6.57</td>
<td>79.0 ± 6.74, 79.5 ± 6.68, 80.7 ± 6.54</td>
<td>80.5 ± 6.57, 82.6 ± 6.28, 80.7 ± 6.54</td>
</tr>
</tbody>
</table>

where

\[ D_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \]

and

\[ D_3 = 1 + \min_{i,j} \frac{\partial \log J(X|\Theta)}{\partial \theta_i^j} \]

\( D_3 \) guarantees a positive parameter after the update in (12) and

\( F > 1 \) regulates the convergence speed of the algorithm.

A large value of \( S \) and \( M \) leads to an HMM with many parameters. For a reliable estimation of many parameters a sufficiently large data set has to be provided. Discriminative training methods were unstable for too low values of the convergence-regulating constant \( F \). The minimum value of \( F \) for convergence has been determined empirically for each data set. For MM-HMM, the margin scaling parameter \( \kappa \in \{0.001, \ldots, 1\} \) is set by 3-fold cross-validation on the training set. Furthermore, the parameter \( \eta \) for the approximation of the maximum in Eq. (3) has limited influence on the performance, i.e. it is set to 2. The objective function does not necessarily increase at each iteration. Reasons are the approximation of the derivatives and a bad choice of \( D \). We used early stopping for training the CLL-HMM and MM-HMM models, i.e. the best number of training iterations is determined on the training set. Numerical underflow might occur during the estimation of very small HMM parameters, leading to unreliable training results.

To overcome this, we set the parameter values to a minimum value. In particular, the values of the covariance matrix and the transition probabilities are set to 0.0001 and 0.001 during optimization, respectively. For DTW, the warping window parameter \( w \in \{1, \ldots, 10\} \) specifies the size of the local neighborhood in DTW, i.e. consider comparing two sequences \( x = \{x_1, \ldots, x_r\} \) and \( y = \{y_1, \ldots, y_r\} \), the distance between \( x_i \) and \( y_j \) is calculated only for indexes \( i \) and \( j \) such that \( |i - j| \leq w \) [Hiroaki & Chiba (1978); Berndt & Clifford (1994)]. \( w \) is selected by cross-validation and the \( L^2 \)-norm is used as distance measure.

5. Experiments

The maximum marginal HMM is compared to the MLE and CLL optimized HMM. We provide results for spoken digit classification using the TIMIT corpus, handwritten character data Australian sign language data, and UCR time-series data. We use the acronym MLE-HMM for generatively learned HMMs and CLL-HMM and MM-HMM for discriminative CLL and maximum margin HMM parameter estimation, respectively. For comparison, we used a 1-nearest neighbor classifier with a similarity measure obtained by dynamic time warping (DTW) [Hiroaki & Chiba (1978); Berndt & Clifford (1994)].

5.1. Experimental Setup

The HMM parameters trained by MLE have been used as initialization for the discriminative methods, i.e. CLL and MM parameter learning. We perform classification using HMMs with varying numbers of mixture components and states. We use up to \( S = 6 \) states and \( M \in \{2, 3, 4\} \) mixture components.

5.2. Handwritten Digit Classification on the Pendigits Database

The Pendigits database contains trajectories of handwritten digits from 0 to 9 from 44 different writers. Some examples

---

The selected \( \kappa \) values for the experiments in the following sections using Auslan, Pendigits, TIMIT, ECG200, OSULeaf, and SwedishLeaf are 0.26, 0.0215, 0.0023, 0.209, 0.209, and 0.209, respectively.
for digit ‘0’, ‘3’, and ‘6’ are shown in Figure 1. The data is divided into a training set of 7494 samples from 30 writers and a test set of 3498 samples from 14 writers, respectively. The sample sequences have a uniform length of 8 time steps. The feature vectors consist of two elements: the absolute pen position in x- and y-direction. Values have been normalized to a range between 0 and 100. We rescaled the values to a range between -1 and 1 and added the first derivatives in x- and y-direction. The derivatives were obtained by numerical differentiation of the coordinates. The classification performances for MLE-HMM, CLL-HMM and MM-HMM of the Pendigits data are shown in Table 1. Best results for each number of states \( S \in \{2, 3, \ldots, 6\} \) are bold.

In this task discriminative MM training clearly outperforms generative MLE-HMMs. MM-HMMs perform mostly slightly better than CLL-HMMs.

### 5.3. Australian Sign Language Classification

The Australian Sign Language (Auslan) data set consists of 6647 samples of 95 different signs from 5 writers. The feature vectors contain 15 values, including the position in x-, y- and \( z' \)-direction, finger bend and more. We ignored the attributes \( 5, 6, \) and 11-14 as advised in the description of the data set. For the experiments, we selected 10 signs that were used in Kim & Pavlovic (2011) and applied a median filter to the data. Furthermore, we compressed the sequences to a fixed length of 10 time steps by taking the means of equally-sized partitions of each sequence. We split the data randomly into 80% of the samples for training and 20% for testing, respectively. The splitting was repeated three times and the average is reported. The classification performances for MLE-HMM, CLL-HMM and MM-HMM of the Auslan data are shown in Table 2. Each result is the mean of three runs with randomly selected partitions for training an testing. Best results for each number of states \( S \in \{2, 3, \ldots, 5\} \) are bold. Discriminative MM training outperforms generative MLE-HMMs. Again, MM-HMMs perform mostly slightly better than CLL-HMMs.

### 5.4. Sequence Classification on the UCR Database

In this experiment, we evaluated HMM classification on three data sets, namely ECG200, OSULeaf, and SwedishLeaf, of the UCR database. The data sets vary in their number of classes, size of training and test set and sequence length. All provided data sets have one single attribute. Furthermore, we compressed the time series to approximately 1/10 of their original length. Only results for the optimal number of states and components are reported.

The classification performances for MLE-HMM, CLL-HMM, MM-HMM and DTW of the UCR data are shown in Table 3. Best results for each dataset are bold. MM-HMMs outperform MLE-HMMs and CLL-HMMs. Fig. 2 shows the convergence of the objective function of CLL-HMMs and MM-HMMs for different values of \( F \) on the SwedishLeaf dataset of the UCR database.

### 5.5. Spoken Digit Classification

A set of spoken numbers from ‘one’ to ‘ten’ has been extracted from the TIMIT corpus (Lamel et al., 1986). The utterances are recorded at a sampling rate of 16 kHz. For each digit a sequence of observation vectors \( x \), consisting of 13 mel-frequency cepstral coefficients (MFCCs) and their first and second derivatives are obtained. The features are whitened at a frame rate of 10ms and a window length of 25 ms. Thus, each observation \( x \) consists of 39 features. Additionally, principal component analysis (PCA) was applied to whiten the data. In total, 165 training and 64 test sequences of spoken digits are available. The classification rates are shown in Tables 3 for MLE, CLL and MM training of the HMM using \( M = 2 \) Gaussian components. Best results for each number of states \( S \) are bold.

In this task, MM training achieves the highest classification rate, i.e. generative MLE-HMMs are outperformed in each case while CLL-HMMs provides the same classification performance for \( S = 5 \) and \( S = 2 \). The classification performance of MLE, CLL and MM decreases with an increasing number of HMM states. Due to the little amount of training samples, these methods are presumably suffering from overfitting, i.e. the number of parameters is too large to be reliably estimated.
### Table 4. Classification rates of MLE-HMMs, CLL-HMMs, and MM-HMMs in [%] on spoken digit classification.

<table>
<thead>
<tr>
<th>Data</th>
<th>MLE-HMM</th>
<th>CLL-HMM</th>
<th>MM-HMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>90.6 ± 7.14</td>
<td>95.3 ± 5.18</td>
<td>95.3 ± 5.18</td>
</tr>
<tr>
<td>3</td>
<td>93.8 ± 5.93</td>
<td>90.6 ± 7.14</td>
<td>96.9 ± 4.26</td>
</tr>
<tr>
<td>4</td>
<td>82.8 ± 9.24</td>
<td>93.8 ± 9.53</td>
<td>98.4 ± 3.04</td>
</tr>
<tr>
<td>5</td>
<td>79.7 ± 9.86</td>
<td>95.3 ± 5.18</td>
<td>95.3 ± 5.18</td>
</tr>
<tr>
<td>6</td>
<td>82.8 ± 9.24</td>
<td>90.6 ± 7.14</td>
<td>96.9 ± 4.26</td>
</tr>
</tbody>
</table>

### Table 5. Classification rates of DTW in [%].

<table>
<thead>
<tr>
<th>Data</th>
<th>Warping window size w</th>
<th>Classification rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pendigits</td>
<td>0</td>
<td>97.8 ± 0.51</td>
</tr>
<tr>
<td>Auslan</td>
<td>1</td>
<td>88.8 ± 5.22</td>
</tr>
<tr>
<td>TIMT</td>
<td>3</td>
<td>79.7 ± 9.86</td>
</tr>
<tr>
<td>ECG200</td>
<td>5</td>
<td>87.0 ± 6.59</td>
</tr>
<tr>
<td>OSULeaf</td>
<td>6</td>
<td>57.9 ± 6.22</td>
</tr>
<tr>
<td>SwedishLeaf</td>
<td>8</td>
<td>89.0 ± 2.46</td>
</tr>
</tbody>
</table>

6. Conclusion

A discriminative maximum margin learning method is derived for hidden Markov models and compared to conditional log-likelihood and maximum likelihood parameter estimation. We formulate the margin of a sample sequence as the ratio of the class posterior of the true class and the most competing class. This sample margin is embedded in a hinge loss function. The derivatives of a smooth approximation of this objective function are used in the extended Baum-Welch algorithm to discriminatively optimize the HMM parameters. In the experiments, we provide results for the tasks of classifying handwritten characters, Australian sign language data, digits of speech data and UCR time-series data. Discriminatively trained HMMs outperform the generative maximum likelihood approach. Maximum margin training outperforms conditional likelihood training in almost all cases. Generative learning.

### Appendix A: EBW Algorithm

In its original form [Baum & Eagon (1967)], the Baum-Eagon inequality has been formulated for domains of discrete probabilities. Consider a domain $E$ of discrete probability values $\Phi = \{\varphi_j\}$, with $\varphi_j \geq 0$, $\sum_j \varphi_j = 1$, and $j = 1, ..., J$. Given a homogeneous polynomial $Q(\Phi)$ with nonnegative coefficients over the domain $E$, the Baum-Eagon inequality offers an iterative method to find local maxima in $Q$. It provides a transformation, $T : E \to E$, such that $Q(T(\Phi)) > Q(\Phi)$, unless $T(\Phi) = \Phi$. This transformation, called growth transform, maps from $\Phi \in E$ to $T(\Phi) = \Phi \in E$, where

$$
\varphi'_j = \frac{\varphi'_j \partial Q(\Phi)}{\partial \varphi'_j} \sum_{i} \varphi'_i \frac{\partial Q(\Phi)}{\partial \varphi'_i}
$$

(40)

For brevity, $\frac{\partial Q(\Phi)}{\partial \varphi'_j}$ denotes the partial derivative $\frac{\partial Q}{\partial \varphi'_j}$ evaluated at point $\Phi$.

In [Gopalakrishnan et al. (1991)], the growth transform is extended3 to rational functions $R(\Phi)$ over $E$:

$$
R(\Phi) = \frac{\text{Num}(\Phi)}{\text{Den}(\Phi)}
$$
This is done by converting $R(\Phi)$ into a polynomial $Q_\Phi(\Phi)$ for a given $\Phi$ such that if $Q_\Phi(T(\Phi)) > Q_\Phi(\Phi)$, then $R(T(\Phi)) > R(\Phi)$, except $T(\Phi) = \Phi$. The polynomial $Q_\Phi(\Phi)$ that fulfills this condition is given in [Gopalakrishnan et al. (1991)] as

$$Q_\Phi(\Phi) = \text{Num}(\Phi) - R(\Phi)\text{Den}(\Phi).$$

To see this, first note that $Q_\Phi(\Phi) = 0$. Thus, if $Q_\Phi(\Phi) > Q_\Phi(\Phi)$, then $\text{Num}(\Phi) > R(\Phi)\text{Den}(\Phi)$, and hence $R(\Phi) > R(\Phi)$.

Unfortunately, the growth transform can not be applied directly to $Q_\Phi(\Phi)$, as it might have negative coefficients. To ensure nonnegativity, the growth transform is instead applied to

$$S_\Phi(\Phi) = Q_\Phi(\Phi) + C(\Phi),$$

where

$$C(\Phi) = \kappa \left( \sum_{j,i} \varphi_i^j + 1 \right).$$

has constant value over $E$, since $\sum_i \varphi_i^j = 1$, and $r$ denotes the maximal order of $Q_\Phi(\Phi)$. Hence, $C(\Phi)$ adds a constant $\kappa$ to every monomial in $Q_\Phi(\Phi)$. This constant $\kappa$ must be chosen such that $S_\Phi(\Phi)$ has nonnegative coefficients for every $\Phi$. Thus, $S_\Phi(\Phi)$ has positive coefficients and still has the same important property as $Q_\Phi(\Phi)$. This polynomial with positive coefficients can now be considered for the growth transform in Eq. (40).

As easily can be verified, the partial derivative of $S_\Phi(\Phi)$ can be expressed in terms of $\frac{\partial \log R(\Phi)}{\partial \varphi_i^j}$, according to

$$\frac{\partial S_\Phi(\Phi)}{\partial \varphi_i^j} = \text{Num}(\Phi) \frac{\partial \log R(\Phi)}{\partial \varphi_i^j} + D,$$

where $D = \kappa r(J + 1)^{-1}$ is the derivative of $C(\Phi)$. Plugging this result into Eq. (40), we finally obtain the extended Baum-Welch re-estimation equation for discrete probability distributions of the form

$$\varphi_i^j \leftarrow \frac{\varphi_i^j \left( \frac{\partial \log R(\Phi)}{\partial \varphi_i^j} + D \right)}{\sum_j \varphi_i^j \left( \frac{\partial \log R(\Phi)}{\partial \varphi_i^j} + D \right)}, \quad (41)$$

where the $\tilde{\varphi}_i^j$ denotes the updated parameters, and constant $D = \kappa r(J + 1)^{-1}$ must be chosen to be sufficiently large.

### Appendix B: MM-HMM EBW Algorithm

The implementation of the EBW algorithm for maximizing the margin, i.e. MM-HMM EBW algorithm, is stated in Algorithm 1.

The E-step of the MM-HMM EBW algorithm using the approximation of $\frac{\partial \log p_h}{\partial \alpha_h}$ (see Eq. (30)) is depicted in Algorithm 2.

In Algorithm 2, the M-step of the MM-HMM EBW algorithm using parameter updates of Eq. (12) is illustrated.

### 7. Acknowledgements

This work was supported by the Austrian Science Fund (Project number P27803-N15).

### References


Cao, L.J, & Tay, F.EH. 2003. Support vector machine with adaptive parameters
M-Step:

\[
\begin{align*}
\hat{\pi}(m) &\leftarrow \sum_{i=1}^{T} \alpha_{i,m}(\mathbf{w}_{i-1},\mathbf{w}_{i}) \gamma_{i,m}(\mathbf{w}_{i}) \\
\hat{\mu}_{j,m} &\leftarrow \frac{1}{\sum_{i=1}^{T} \alpha_{i,m}(\mathbf{w}_{i-1},\mathbf{w}_{i})} \sum_{i=1}^{T} \alpha_{i,m}(\mathbf{w}_{i-1},\mathbf{w}_{i}) \mathbf{w}_{i:j} \\
\hat{\Sigma}_{j,m} &\leftarrow \frac{1}{\sum_{i=1}^{T} \alpha_{i,m}(\mathbf{w}_{i-1},\mathbf{w}_{i}) (\mathbf{w}_{i:j} - \hat{\mu}_{j,m}) (\mathbf{w}_{i:j} - \hat{\mu}_{j,m})^{T}} \sum_{i=1}^{T} \alpha_{i,m}(\mathbf{w}_{i-1},\mathbf{w}_{i}) (\mathbf{w}_{i:j} - \hat{\mu}_{j,m})^{T}
\end{align*}
\]

\[
\pi(z) \leftarrow \hat{\pi}(z) \forall z
\]

Algorithm 3: M-step of the MM-HMM EBW algorithm.

in financial time series forecasting. *IEEE Transactions on Neural Networks* 14(6), 1506–1518.

Ebrahimpour, R., & Hamedi, S. 2009. Hand written digit recognition by mul-
siple classifier fusion based on decision templates approach. *Pages – of: CESSSE.*


Gopalakrishnan, O., Kanevsky, D., Nadas, A., & Nahamoo, D. 1991. An in-
dependence for rational functions with applications to some statistical estima-


Conf. on Machine Learning (ICML).*