Graz University of Technology

Master's Thesis

Maximum Margin Hidden Markov Models

conducted at the
Signal Processing and Speech Communications Laboratory
Graz University of Technology, Austria

by
Nikolaus Mutsam

Supervisor:
Franz Pernkopf

Graz, March 12, 2013

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

date	(signature)

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am	(Unterschrift)

Abstract

Discriminative learning methods are well-approved techniques in machine learning for pattern classification. In particular, the maximum margin approach, known from state-of-the-art support vector machine classifiers, can outperform generative and also discriminative learning schemes under certain conditions. Hidden Markov models (HMMs) are widely used to model data sequences of variable length. It has been shown that generatively as well as discriminatively trained HMMs can yield good performance in sequence classification tasks. In this work, the maximum margin approach is introduced to HMMs. We compare maximum margin HMM training to generative as well as other discriminative HMM training schemes such as maximum mutual information training. In particular, we use the extended Baum-Welch (EBW) algorithm for parameter estimation to maximize the objective functions of discriminative training methods for HMMs. We utilize the Viterbi algorithm to simplify the derivatives of the objective functions required by the EBW algorithm. Furthermore, robust approximations to these derivatives are presented. Experimental results for real-world classification problems are provided, showing the capabilities of maximum margin HMM training. The maximum margin approach is compared with the generative method of maximum likelihood estimation (MLE) and the discriminative training scheme of conditional log-likelihood (CLL) parameter estimation. We present results on the tasks of human speech and handwritten digit classification. The experiments show that maximum margin learning achieves good classification performance and competes with both MLE and CLL training.

Kurzfassung

Diskriminative Lernmethoden sind eine weithin anerkannte Technik maschinellen Lernens zur Mustererkennung. Im Speziellen kann der Maximum Margin Ansatz, bekannt durch die Klassifikationsmethode der Support Vektor Machines, generative und auch diskriminative Lernverfahren unter gewissen Bedingungen übertreffen. Hidden Markov Modelle (HMM) werden gemeinhin zur Modellierung von Datensequenzen unterschiedlicher Länge verwendet. Es wurde gezeigt, dass sowohl generativ als auch diskriminativ trainierte HMM gute Ergebnisse bei der Klassifikation von Sequenzen erzielen können. Diese Arbeit beschäftigt sich mit der Adaptierung des Maximum Margin Ansatzes für HMM. Wir vergleichen Maximum Margin Training mit generativen und anderen diskriminativen Lernverfahren für HMM, wie zum Beispiel Maximum Mutual-Information Training. Im Speziellen verwenden wir den erweiterten Baum-Welch Algorithmus (EBW) zur Parameterschätzung bzw. zur Maximierung der Zielfunktionen von diskriminativen Trainingsmethoden für HMM. Zur Vereinfachung der vom EBW-Algorithmus benötigten Ableitungen dieser Zielfunktionen wird der Vitberbi Algorithmus verwendet. Des Weiteren werden robuste Approximationen dieser Ableitungen vorgestellt. Experimentelle Resultate zu realen Klassifikationsaufgaben zeigen die Leistungsfähigkeit des Maximum Margin Trainings für HMM. Der Maximum Margin Ansatz wird mit der generativen Maximum Likelihood Estimation Methode (MLE) und dem diskriminativen Conditional Log-Likelihood Trainigsschema (CLL) verglichen. Wir präsentieren Resultate zur Klassifikation menschlicher Sprache und handgeschriebener Symbole. Die Experimente zeigen, dass Maximum Margin Training gute Klassifikationsergebnisse erzielen und sich sowohl mit MLE Training als auch mit CLL Training messen kann.

Acknowledgements

I like to thank my supervisor Franz Pernkopf for his invaluable help during the work on this thesis. I thank him for all the productive discussions and for keeping me motivated all the time. Furthermore, I give sincere thanks to all my dear friends and colleagues who have accompanied me during the course of studies.

Contents

1 Introduction

The analysis of sequential data and time series covers a wide field of applications, such as speech analysis [1], financial mathematics [2] and weather forecasts [3], just to name a few of them. The broad variety of possibilities to utilize sequential data analysis yields interesting tasks in terms of machine learning, for example the prediction of future events given a time series, data clustering and pattern classification.
A time series is a set of data samples with a sequential order. Thus, the data samples are correlated and not independent and identically distributed (i.i.d.). In other words, the observations in a time series are dependent on previous samples in the sequence. In order of being able to draw any conclusion of a given time series, it is indeed very helpful to understand the process which produces that given sequence. Therefore, it is an important task in the analysis of time series to find an appropriate statistical model that represents the underlying stochastic process given some data produced by that process. A model where a data sample of an observed sequence depends on all previous samples would be computationally infeasible, especially when the sequence has no limit in its length. To overcome this, the assumption is taken that more recent samples have a greater influence on the present than samples in the farther past. In particular, a *Markov model* restricts the influence on the present to a few events in the most recent past. The number of influencing events in the past determines the order of the Markov Model. In a first-order Markov model, the present event only depends on its predecessor, in a second-order Markov model it depends on the two previous events and so forth.
A *hidden Markov model (HMM)* can be used to describe sequences that cannot be observed directly. At every time step, the Markov model is in a certain *hidden* state and produces an observable data sample. The current state is dependent on the model's state in one or more previous time steps, according to the order of the Markov model. The state space of HMMs is discrete, i.e. they have a finite number of different states [4–6].

1.1 Scope of Research

This thesis concentrates on discriminative training of first-order HMMs with emission probabilities represented by Gaussian Mixture Models (GMMs). The focus of research is on margin maximization in HMMs. A specific definition of the margin is investigated and compared to another criterion for discriminative training, namely the conditional log-likelihood (CLL). To optimize the objective functions of the discriminative training methods, i.e. the conditional

log-likelihood (CLL) and the margin criterion, the extended Baum-Welch (EBW) algorithm is adapted to HMMs. The EBW algorithm uses the derivatives of the objective functions. Robust approximations to the derivatives are discussed. Experimental results are presented for the tasks of classifying human speech and handwritten digits. In particular, discriminatively optimized CLL-HMMs and maximum margin HMMs are compared with state-of-the-art generative HMMs, where the maximum likelihood objective is optimized. In the experiments, three different data sets are used, one for speech classification and two for handwritten digit classification.

1.2 Organization of the Thesis

This thesis is organized as follows: The structure and basic algorithms of HMMs are presented in Chapter (2), as well as their application to classification tasks. Furthermore, the notation is introduced. Chapter (3) is devoted to HMM learning. First traditional ML learning is introduced. Then discriminative training methods using conjugate gradient methods are discussed. In particular, we use the CLL criterion and introduce the margin objective. Implementation issues are also in the scope of this chapter. Chapter (4) presents experimental results of the application of the introduced HMM training methods for sequence classification tasks. Two practical tasks in sequence classification are presented, i.e. the classification of speech on a phonetic and a word level and handwritten digits. Chapter (5) concludes the thesis and provides a perspective on future work.

2
Hidden Markov Models

An HMM can be fully described by two stochastic processes [5,6]. The first is a Markov-process that produces a sequence of states that cannot be directly observed. The second process produces an observation at every time step of the state sequence according to a state-dependent probability distribution. Thus, the observation process is driven by the state process. Fig. (2.1) illustrates a first-order HMM as graphical model [4]. A state sequence $Q = \{q_1, q_2, \ldots, q_t, \ldots, q_T\}$, where each element q_t is drawn from a set $\mathcal{S} = \{1, 2, \ldots, S\}$ of S states. The state of element q_t depends only on the state of its preceding element q_{t-1}. At any time-step, an observation $\mathbf{x}_t$ is produced, giving an observation sequence $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t, \ldots, \mathbf{x}_T\}$, where T denotes the length of the sequence. The observations are drawn either from a discrete or a continuous probability distribution. They depend only on the state in the same time-step. Furthermore, the observation at each time step can be either a vector $\mathbf{x}_t$ or a scalar x_t. Throughout the thesis it is assumed to observe a continuous vector $\mathbf{x}_t \in \mathbb{R}^D$, where D is the number of observations in each time-step.

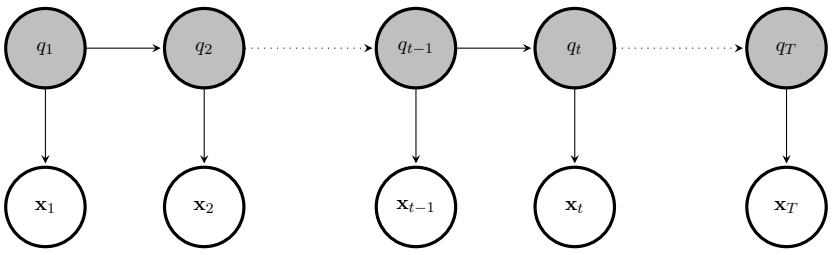


Figure 2.1: A first order Markov chain. The hidden state sequence $Q = \{q_1, \ldots, q_T\}$ is shown by shaded nodes.

At each time step, the variable q_t is in one of $\mathcal{S}$ states, dependent on the state of the previous variable $q_{t-1} \in \mathcal{S}$. Hence, there are S times S possible state transitions in every time-step, where each transition from state i to state j occurs with a certain probability denoted by $a_{i,j} = P(q_t = j | q_{t-1} = i)$. Fig. (2.2) shows the possible state transitions for a model with three different states.

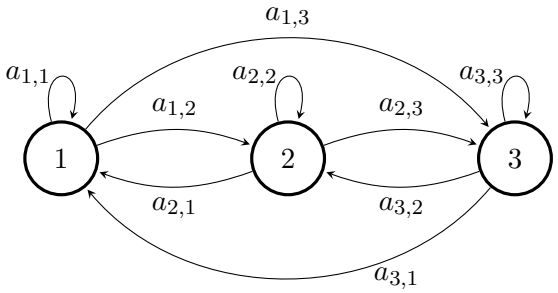


Figure 2.2: A Markov model with three states.

A *transition matrix* $A \in S \times S$ collects all transition probabilities $a_{i,j}$. It is given by

$$A = \begin{bmatrix} a_{1,1} & \cdots & a_{1,j} \\ \vdots & \dots & \vdots \\ a_{i,1} & \cdots & a_{i,j} \end{bmatrix} \quad , \quad \text{where} \qquad \sum_{j=1}^{S} a_{i,j} = 1 \quad \forall i \in \{1, \dots, S\}. \tag{2.1}$$

In an HMM, the state of the model in a particular time-step cannot be observed directly. Instead, the model emits an observation dependent on the model's current hidden state. In each state, the model produces emissions according to a state-dependent probability distribution. The probability of observing a certain emission while the model is in state i at time t is called emission or observation probability $b_i(\mathbf{x}_t) = p(\mathbf{x}_t|q_t = i)$. The Markov model of Fig. (2.2) with the added emission probabilities is shown in Fig. (2.3).

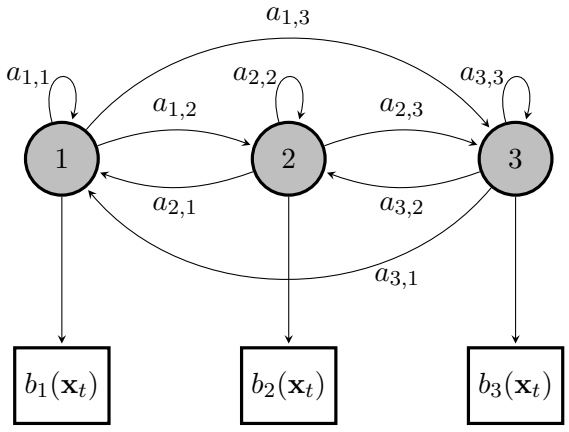


Figure 2.3: Emission probabilities depend on the model's state.

A time series can be modelled as a series of observations that is produced by a HMM while running through a series of state transitions. In the first time step, there is no transition from a previous hidden state. The probability of the model being in hidden state i at the beginning of a sequence $t = 1$ is called the state prior distribution $\pi_i = p(\mathbf{x}_1|q_1 = i)$.

The state prior can be imagined as the transition from an observable start state the model is in with a probability of 1 to one of the hidden states at the beginning of the time series. A hidden Markov model is fully defined by the three parameters discussed above: the state prior distribution π_i, the transition matrix A and the emission probability $b_i(\mathbf{x}_t)$. Fig. (2.4) shows an HMM with all parameters $\mathbf{\Theta} = \{\pi_i, \dots, \pi_S, A, b_i(\mathbf{x}_t), \dots, b_S(\mathbf{x}_t\}$ for modelling a time series.

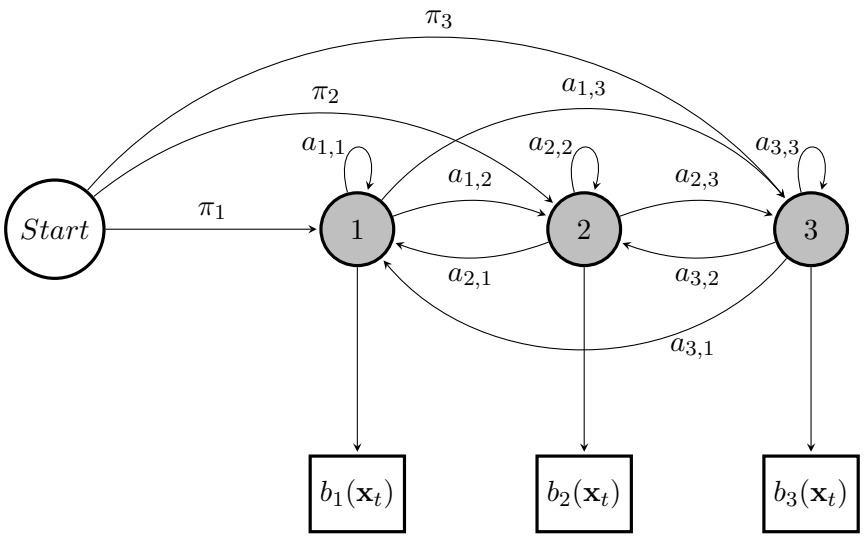


Figure 2.4: An HMM with all parameters. The series of state transitions starts in a defined state named Start.

2.1 HMM Structures

The structure of an HMM is specified by its transition matrix A. State transitions have a probability between 0 and 1. An HMM where all state transitions are nonzero is called a fully connected or ergodic HMM.

2.1.1 Left-Right HMMs

In a left-right HMM, transitions to a state with a lower index than the current index are not possible. An example of such a left-right HMM with three hidden states is depicted in Fig. (2.5).

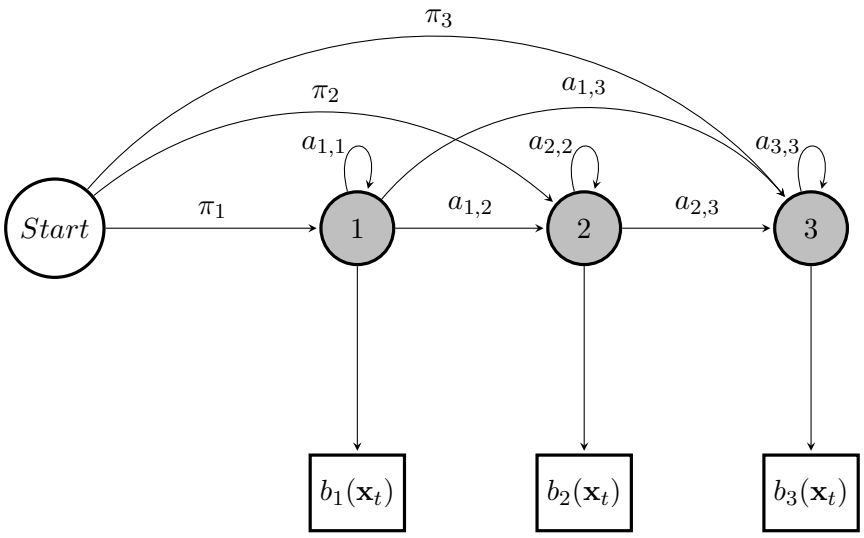


Figure 2.5: A left-right HMM.

The corresponding transition matrix A and state prior π_i probabilities are

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ 0 & a_{2,2} & a_{2,3} \\ 0 & 0 & a_{3,3} \end{bmatrix} \quad \text{and} \quad \pi = \begin{bmatrix} \pi_1 \\ \pi_2 \\ \pi_3 \end{bmatrix} \quad \text{, respectively.}$$

2.1.2 Linear HMMs

A linear HMM allows transitions from a state only to itself and to the state with an index of one higher than itself. Therefore, no state can be omitted in a sufficiently long time series.

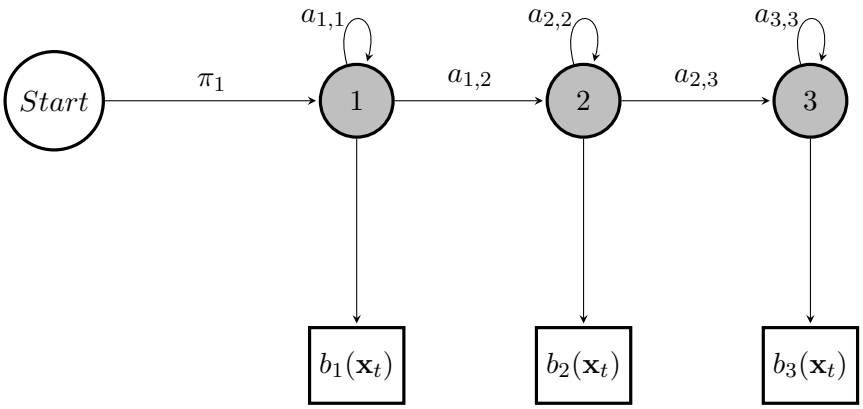


Figure 2.6: A linear HMM.

A linear HMM is depicted in Fig. (2.6) with the associated state prior and transition probabilities

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & 0 \\ 0 & a_{2,2} & a_{2,3} \\ 0 & 0 & a_{3,3} \end{bmatrix} \quad \text{and} \quad \pi = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

2.2 Observation Probabilities

As mentioned above, the observations can be either discrete or continuous. Furthermore, at each time step a single variable x_t or a set of variables $\mathbf{x}_t$ can be observed.

2.2.1 Discrete Observations

If the possible observations are limited to a finite number of K discrete symbols, i.e. $\mathbf{x}_t \in \{1, \ldots, K\}$, the observation probabilities for S hidden states can be stated by an $S \times K$ matrix:

$$B = \begin{bmatrix} b_1(\mathbf{x}_t = 1) & b_1(\mathbf{x}_t = 2) & \cdots & b_1(\mathbf{x}_t = k) \\ b_2(\mathbf{x}_t = 1) & \cdots & \cdots & \vdots \\ \vdots & \cdots & \cdots & \vdots \\ b_S(\mathbf{x}_t = 1) & \cdots & \cdots & b_S(\mathbf{x}_t = k) \end{bmatrix}, \tag{2.2}$$

where $\sum_{k=1}^{K} b_i(\mathbf{x}_t = k) = 1 \quad \forall i \in \{1, \ldots, S\}$.

2.2.2 Continuous Observations

In a continuous observation space, the observation probabilities can be described by one continuous probability density function (pdf) per state. Often the Gaussian or normal distribution is used. It is fully specified by only two parameters, the mean and variance. In the case of multidimensional observations, i.e. $\mathbf{x}_t \in \mathbb{R}^D$, the multivariate Gaussian distribution for a D-dimensional observation vector $\mathbf{x}_t$ of state i is defined as follows:

$$b_i(\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_t | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{\sqrt{(2\pi)^D |\boldsymbol{\Sigma}_i|}} e^{-\frac{1}{2}(\mathbf{x}_t - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_t - \boldsymbol{\mu}_i)}, \tag{2.3}$$

where $\int_{-\infty}^{\infty} b_i(\mathbf{x}_t) d\mathbf{x} = 1$ for all $i \in \{1, \ldots, S\}$.
In Eq. (2.3), $\boldsymbol{\mu}_i$ is the D-dimensional mean vector, $\boldsymbol{\Sigma}_i$ is the $D \times D$ symmetric covariance matrix and $|\boldsymbol{\Sigma}|$ is its determinant.
In case the observations show a multimodal distribution, a multivariate Gaussian mixture model (GMM) can be used. A GMM is a weighted sum of M multivariate Gaussians, where the emission probabilities are defined as follows:

$$b_i(\mathbf{x}_t) = \sum_{m=1}^{M} \alpha_{i,m} \cdot b_{i,m}(\mathbf{x}_t) = \sum_{m=1}^{M} \alpha_{i,m} \cdot \mathcal{N}(\mathbf{x}_t | \boldsymbol{\mu}_{i,m}, \boldsymbol{\Sigma}_{i,m}), \tag{2.4}$$

where $\alpha_{i,m}$ are the weights of each Gaussian component such that $\int_{-\infty}^{\infty} b_i(\mathbf{x}_t) d\mathbf{x} = 1$.

Therefore, $0 \leq a_{i,m} \leq 1$ and $\sum_{m=1}^{M} \alpha_{i,m} = 1$.

2.3 Problems for HMMs

Basically, there are three problems that have to be solved:

- **Evaluation problem**
  Calculating the probability $p(\mathbf{x}|\mathbf{\Theta})$ that a given HMM $\mathbf{\Theta}$ produces a given observation sequence $\mathbf{x}$ is called the evaluation problem. The evaluation problem can be efficiently solved by the *forward procedure* as well as by the *backward procedure* [5, 6].

- **Decoding problem**
  The decoding problem aims at finding the most probable state sequence $Q^* = \{q_1^*, \ldots, q_T^*\}$ given $\mathbf{x}$ and $\mathbf{\Theta}$. The *Viterbi* algorithm provides a solution to this problem [5, 6].

- **Learning problem**
  Adjusting the parameters of a HMM, i.e. $\mathbf{\Theta} = \{\pi_i, \ldots, \pi_S, A, b_i(\mathbf{x}_t), \ldots, b_S(\mathbf{x}_t\}$, to optimize a desired criterion, e.g. to maximize the model likelihood $p(\mathbf{x}|\mathbf{\Theta})$ for given data, is known as learning problem. The learning problem is the most complex of the three basic HMM problems. Various approaches for solving this problem are discussed in Chapter (3).

Before solutions to each of these problems are presented, the *trellis diagram* is introduced.

2.3.1 Trellis Diagram

A trellis diagram visualizes the state transitions of an HMM over the time. At each time-step, each state of the HMM is represented by a node. State transitions are represented by edges. Each path in the trellis diagram is related to a particular state sequence. The trellis diagram in Fig. (2.7) shows all possible state sequences for the left-right HMM of Fig. (2.5) for a sequence of five time-steps. The red path marks a state sequence of $Q = \{1, 1, 2, 3, 3\}$.

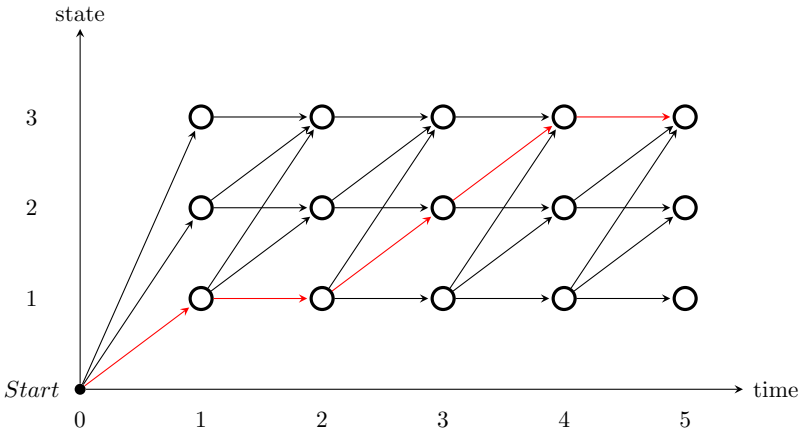


Figure 2.7: Trellis diagram.

For a given HMM, a transition probability can be assigned to each edge in the trellis. Analogously, for a given observation sequence $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$, to each node the observation probability of the current time step can be assigned. This is shown in Fig. (2.8).

The probability of a state sequence $Q = \{q_1, \ldots, q_T\}$ and an observation sequence $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$ is simply the product of the probabilities along the path of Q. For example, the probability of an observation sequence $\mathbf{x}$ and the state sequence $Q = \{1, 1, 2, 3, 3\}$ depicted in Fig. (2.7) given the parameters $\mathbf{\Theta}$ of an HMM is $p(\mathbf{x}, Q|\mathbf{\Theta}) = \pi_1 \cdot b_1(\mathbf{x}_1) \cdot a_{1,1} \cdot b_1(\mathbf{x}_2) \cdot a_{1,2} \cdot b_2(\mathbf{x}_3) \cdot a_{2,3} \cdot b_3(\mathbf{x}_4) \cdot a_{3,3} \cdot b_3(\mathbf{x}_5)$.

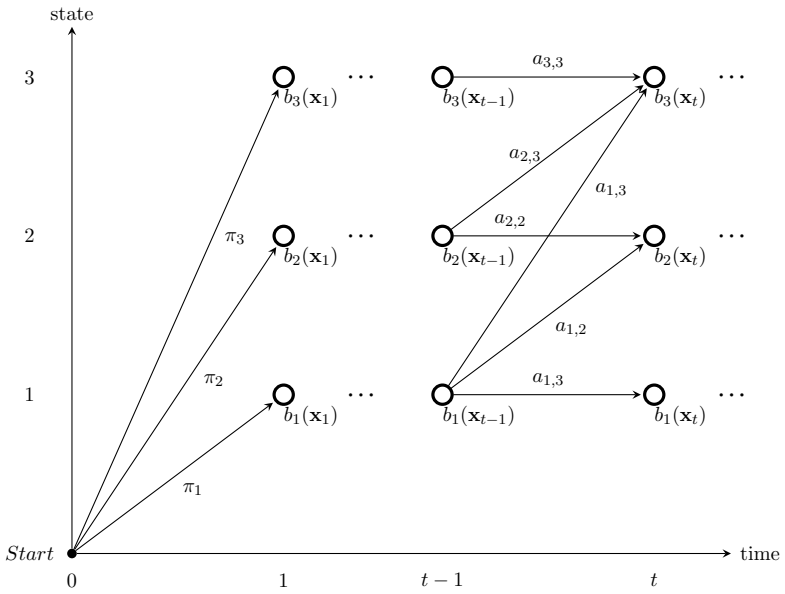


Figure 2.8: Trellis Diagram with transition and observation probabilities.

2.3.2 Forward-Procedure

Supposing a fully connected HMM $\mathbf{\Theta}$ with S hidden states, for an observation sequence Q of length T, there are S^T possible paths through the trellis. For evaluating the probability $p(\mathbf{x}|\mathbf{\Theta})$ of a given observation sequence $\mathbf{x}$, the probabilities of all possible paths, i.e. all possible state sequences, given the observation sequence could be calculated separately and then be summed up, i.e. $p(\mathbf{x}|\mathbf{\Theta}) = \sum_Q p(\mathbf{x}, Q|\mathbf{\Theta})$. Therefore, the computational effort would increase exponentially with the number of observations, i.e. $\mathcal{O}(T \cdot S^T)$.
Fortunately, the computational complexity can be reduced by making use of dynamic programming. The idea is to store intermediate results that can be reused. For a given observation sequence $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t, \mathbf{x}_{t+1}, \ldots, \mathbf{x}_T\}$, let $\omega_{i,t}$ be the *forward probability* of being in state i at time-step t for a partial sequence $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t\}$, starting at $\mathbf{x}_1$ and ending at the present observation $\mathbf{x}_t$:

$$\omega_{i,t} := p(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t, q_t = i|\mathbf{\Theta}) \tag{2.5}$$

To calculate $\omega_{i,t}$, the probabilities of all partial paths that are in state i at time-step t must be summed up. For $t = 1$, there is only one path from the *Start* state to each hidden state:

$$\omega_{i,1} = \pi_i b_i(\mathbf{x}_1), \quad 1 \leq i \leq S \tag{2.6}$$

For the calculation of $\omega_{i,t+1}$, the results of time-step t are reused recursively, i.e.

$$\omega_{j,t+1} = \left[\sum_{i=1}^{S} \omega_{i,t} a_{i,j}\right] b_i(\mathbf{x}_{t+1}), \quad 1 \leq j \leq S, \quad 1 \leq t \leq T-1. \tag{2.7}$$

The sum of $\omega_{i,T} = p(\mathbf{x}, q_T = i|\mathbf{\Theta})$ over all states at time-step T gives the probability of the whole observation sequence, i.e.

$$p(\mathbf{x}|\mathbf{\Theta}) = \sum_{i=1}^{S} \omega_{i,T}. \tag{2.8}$$

This solves the evaluation problem efficiently. Each time-step requires the same number of computations. Thus, using the forward procedure, the computational effort increases linearly with the sequence length, i.e. $\mathcal{O}(S^2 \cdot T)$.

2.3.3 Backward-Procedure

The *backward probability* is defined as the probability of the observations $\{\mathbf{x}_{t+1}, \mathbf{x}_{t+2}, \ldots, \mathbf{x}_T\}$ in the future of time-step t, given the state i at time-step t, i.e.

$$\beta_{i,t} := p(\mathbf{x}_{t+1}, \mathbf{x}_{t+2}, \ldots, \mathbf{x}_T|q_t = i, \mathbf{\Theta}) \tag{2.9}$$

The backward probabilities can be calculated in the same manner as the forward probabilities, just starting at the end of the sequences. Hence, the evaluation problem is solved by the following procedure:

$$\textbf{Initialization:}\quad \beta_{i,T} = 1, \quad 1 \leq i \leq S \tag{2.10}$$

$$\textbf{Recursion:}\quad \beta_{i,t} = \sum_{j=1}^{S} a_{i,j} b_j(\mathbf{x}_{t+1}) \beta_{j,t+1}, \quad 1 \leq i \leq S, \quad T-1 \geq t \geq 1 \tag{2.11}$$

$$\textbf{Termination:}\quad p(\mathbf{x}|\mathbf{\Theta}) = \sum_{i=1}^{S} \pi_i b_i(\mathbf{x}_1) \beta_{i,1} \tag{2.12}$$

2.3.4 Viterbi-Algorithm

Decoding the most probable state sequence Q^* for a given observation sequence $\mathbf{x}$ can be carried out in a similar recursive manner as the evaluation problem. The only difference to the forward procedure is that for each intermediate result, the maximum instead of the sum of the previous results is taken. For a given observation sequence $\{\mathbf{x}_1, \ldots, \mathbf{x}_t\}$, let $\delta_{i,t}$ be the probability of the most probable partial path to state i at time step t:

$$\delta_{i,t} := \max_{q_1, q_2, \ldots, q_{t-1}} p(q_1, q_2, \ldots, q_{t-1}, q_t = i, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t|\mathbf{\Theta}), \quad 1 \leq i \leq S \tag{2.13}$$

As there is only one path from the *Start* state to each hidden state, initialization is the same as for the forward procedure:

$$\delta_{i,1} = \pi_i b_i(\mathbf{x}_1), \quad 1 \leq i \leq S \tag{2.14}$$

$$\psi_{i,1} = 0 \tag{2.15}$$

For the next time-step $t+1$, the maximum of the previous $\delta_{i,t}$ is used. Additionally, for each time step t, $\psi_{j,t}$ indices the preceding state of the most probable partial path to state j. Hence, $\delta_{j,t+1}$ and $\psi_{j,t+1}$ are calculated recursively according to:

$$\delta_{j,t+1} = \max_{1 \leq i \leq S} [\delta_{i,t} a_{i,j}] b_j(\mathbf{x}_{t+1}), \quad 1 \leq j \leq S, \quad 1 \leq t \leq T-1 \tag{2.16}$$

$$\psi_{j,t+1} = \arg \max_{1 \leq i \leq S} [\delta_{i,t} a_{i,j}], \quad 1 \leq j \leq S, \quad 1 \leq t \leq T-1 \tag{2.17}$$

The probability of the most probable path for a given observation sequence is the intermediate result at the last time-step T. It can be determined as

$$p^*(\mathbf{x}|\mathbf{\Theta}) = \max_{1 \leq i \leq S} \delta_{i,T}. \tag{2.18}$$

The most probable path $Q^* = \{q_1^*, \ldots, q_T^*\}$ ends in $q_T^* = \arg\max_{1 \leq i \leq S} \delta_{i,T}$ and can be found by backtracking the indices $\psi_{i,t}$: $q_t^* = \psi_{q_{t+1}^*, t+1}$, where $T-1 \geq t \geq 1$.

2.4 Classification of Time Series Data by HMMs

The task of classification is to assign a given observation sequence $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$ to a class c out of a finite set of $\mathcal{C} = \{1, \ldots, C\}$ classes. For each class, one HMM $\mathbf{\Theta}_c$ is trained. According to Bayes' rule, the class posterior $p(c|\mathbf{x})$ is given by [4]

$$p(c|\mathbf{x}) = \frac{p(\mathbf{x}|c)p(c)}{p(\mathbf{x})} = \frac{p(\mathbf{x}|c)p(c)}{\sum_{c'=1}^{C} p(\mathbf{x}|c')p(c')}, \tag{2.19}$$

where the likelihood is assumed to be a parametric model for class c, i.e. $p(\mathbf{x}|c) = p(\mathbf{x}|\mathbf{\Theta}_c)$. The class label can be determined by the maximum a-posteriori (MAP) estimate, i.e. by assigning a sample $\mathbf{x}$ to the class c^* with the highest class posterior probability $p(c|\mathbf{x})$. It is given by

$$c^* = \arg \max_{1 \leq c \leq C} p(c|\mathbf{x}) = \arg \max_{1 \leq c \leq C} [p(\mathbf{x}|\mathbf{\Theta}_c)\rho_c], \tag{2.20}$$

where the denominator of Eq. (2.19) is obsolete for determining the highest posterior as it scales every class posterior by the same value and therefore does not influence the choice of c^*. Similar to the state prior distribution π_i, the class prior distribution $\rho_c = p(c)$ is the initial probability of a class c. The probability $p(\mathbf{x}|\mathbf{\Theta}_c)$ is determined by the evaluation problem and can be computed by using either the forward or the backward procedure discussed in Section (2.3.2) and (2.3.3). If the most probable state sequence Q^* producing $\mathbf{x}$ is known or estimated by the Viterbi algorithm, $p(\mathbf{x}|\mathbf{\Theta}_c)$ can be approximated by $p^*(\mathbf{x}|\mathbf{\Theta})$, i.e. the product of the probabilities along the path Q^* with the highest probability through the trellis of HMM $\mathbf{\Theta}_c$ (see Section (2.3.1)).

The learning problem, i.e. adjusting the HMM parameters $\mathbf{\Theta}_c$ given training data, is covered in Chapter 3.

3
Training of Hidden Markov Models

If some given data is assumed to be generated from a certain model, it may be desirable to know the model's parameters in order to make further use of it, e.g. generating new data or the classification of samples. Training is the calculation or estimation of the model parameters for the purpose of making the model most appropriate for its designated use with respect to a given set of data samples.
Training a model for sequence classification is to estimate the model parameters such that a sample sequence is assigned to the correct class according to some decision rule. As discussed in Section (2.4), the MAP estimate can be used as decision rule for sequence classification, i.e. a sample sequence $\mathbf{x}$ is assigned to the class c^* with the highest class posterior $p(c|\mathbf{x})$. Thus, using the MAP decision rule, the aim of training is the estimation of the model parameters such that the correct class posterior of a given sample is higher than all other class posteriors. Two training schemes are discussed in this work, namely generative and discriminative training. Generative training fits the model parameters to model the distribution of the training data whereas discriminative training estimates the parameters to model the class posterior probability directly.
The aim of generative training is to determine the model parameters such that samples generated by the model have the same statistical distribution as the training data. When one model per class is fitted to the corresponding class distribution, it is assumed that a given sample is drawn from one of these distributions and therefore the model of the sample's class would give the highest posterior probability. A generative training method usually maximizes the likelihood of the model given training data. The maximum likelihood estimation of HMMs is presented in Section (3.1).
The intention of discriminative training is to optimize the model parameters such that the classification performance is optimized. For instance, in conditional likelihood (CL) learning the class posterior of the model is maximized. In maximum margin optimization, the margin between the class posteriors of the model of the correct class and the most competitive class is maximized. Both discriminative training schemes are derived for HMMs in Section (3.2) and (3.3). Usually, discriminatively learned models result in an increase of classification performance. This is especially true if the model does not fit the underlying distribution well [4].
Throughout this chapter, the variables i and j are used to indicate the hidden states of an HMM and m indicates the mixture components of a GMM. In all equations where these indices are used, $1 \leq i \leq S$, $1 \leq j \leq S$ and $1 \leq m \leq M$, where S is the number of states of an HMM and M is the number of mixture components of a GMM.

3.1 Maximum Likelihood Estimation

Given a set of N_c training sequences $\mathcal{X}_c = \{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^{N_c}\}$ belonging to the same class $c \in \{1, \ldots, C\}$, the likelihood of $\mathcal{X}_c$ given a HMM $\mathbf{\Theta}_c$ is

$$\mathcal{L}(\mathcal{X}_c|\mathbf{\Theta}_c) = p(\mathcal{X}_c|\mathbf{\Theta}_c) = \prod_{n=1}^{N_c} p(\mathbf{x}^n|\mathbf{\Theta}_c) \tag{3.1}$$

where $\mathbf{\Theta}_c = \{\pi_i, \ldots, \pi_S, A, b_i(\mathbf{x}_t), \ldots, b_S(\mathbf{x}_t\}$ is the set of parameters that define the HMM of class c, i.e. state prior, transition matrix and observation probabilities. The aim of maximum likelihood estimation (MLE) is to find a Model $\mathbf{\Theta}_c^*$ such that

$$\mathbf{\Theta}_c^* = \arg\max_{\mathbf{\Theta}_c} \mathcal{L}(\mathcal{X}_c|\mathbf{\Theta}_c). \tag{3.2}$$

The analytical approach to Eq. (3.2) , solving

$$\frac{\partial \mathcal{L}(\mathcal{X}_c|\mathbf{\Theta}_c)}{\partial \mathbf{\Theta}_c} = 0, \tag{3.3}$$

does not provide a closed-form solution [6]. Therefore, an iterative scheme, such as the Baum-Welch algorithm, which is basically an expectation maximization (EM) algorithm, may be applied.

3.1.1 The Baum-Welch Algorithm

The iterative method of the EM algorithm for maximum likelihood estimation consists of two steps. In the expectation step (E-step), the expected values of some hidden quantities are calculated given the current model parameters. In the maximization step (M-step), the model parameters are updated on the basis of the quantities from the E-step. The updated parameters are then used for estimating the hidden quantities in the E-step and so on. Further details to the EM algorithm for HMMs can be found in [7].

- **E-step:**

  In the E-step, two quantities are calculated, given a training set $\mathcal{X}_c$ and the current model $\mathbf{\Theta}_c$. The first is the probability of a state transition from state i to state j given every sequence $\mathbf{x}^n$ at every time step t:

$$\begin{aligned} \xi_{c,i,j,t}^n &:= p(q_t = i, q_{t+1} = j|\mathbf{x}^n, \Theta_c) && (3.4)\\ &= \frac{p(\mathbf{x}^n, q_t = i, q_{t+1} = j|\mathbf{\Theta}_c)}{p(\mathbf{x}^n|\mathbf{\Theta}_c)} \\ &= \frac{\omega_{c,i,t}^n a_{c,i,j} b_{c,j}(\mathbf{x}_{t+1}^n)\beta_{c,j,t+1}^n}{\sum_{i=1}^{S}\sum_{j=1}^{S} \omega_{c,i,t}^n a_{c,i,j} b_{c,j}(\mathbf{x}_{t+1}^n)\beta_{c,j,t+1}^n}. && (3.5) \end{aligned}$$

The numerator in the last term uses the forward and the backward probability introduced in Chapter (2). In particular, $\omega^n_{c,i,t} = p(\mathbf{x}^n_1, \ldots, \mathbf{x}^n_{t-1}, \mathbf{x}^n_t, q_t = i|\mathbf{\Theta}_c)$ and $\beta^n_{c,j,t+1} = p(\mathbf{x}^n_{t+2}, \ldots, \mathbf{x}^n_T|q_{t+1} = j, \mathbf{\Theta}_c)$ leads to $p(q_t = i, q_{t+1} = j|\mathbf{x}^n, \mathbf{\Theta}_c) = \omega^n_{c,i,t} a_{c,i,j} b_{c,j}(\mathbf{x}^n_{t+1}) \beta^n_{c,j,t+1}$, where we add index c to $a_{c,i,j}$ and $b_{c,j}(\mathbf{x}^n_{t+1})$ to denote the parameters of $\mathbf{\Theta}_c$.

The second quantity is the probability of being in state i at time step t given the observation sequence $\mathbf{x}^n$:

$$\begin{aligned}\zeta^n_{c,i,t} &:= p(q_t = i|\mathbf{x}^n, \mathbf{\Theta}_c) \\ &= \frac{p(\mathbf{x}^n, q_t = i|\mathbf{\Theta}_c))}{p(\mathbf{x}^n|\mathbf{\Theta}_c)} = \sum_{j=1}^{S} \xi^n_{c,i,j,t}. \end{aligned} \tag{3.6}$$

The last equality is basically the marginalization of $\xi^n_{c,i,j,t}$. The contribution of each mixture component m of $b_{c,i}(\mathbf{x}^n_t)$ to $\zeta^n_{c,i,t}$ is collected in $\zeta^n_{c,i,m,t}$, i.e.

$$\begin{aligned}\zeta^n_{c,i,m,t} &:= \zeta^n_{c,i,t} \cdot \frac{b_{c,i,m}(\mathbf{x}^n_t)}{b_{c,i}(\mathbf{x}^n_t)} \\ &= \zeta^n_{c,i,t} \cdot \frac{\alpha_{c,i,m} \cdot \mathcal{N}(\mathbf{x}^n_t|\boldsymbol{\mu}_{c,i,m}, \mathbf{\Sigma}_{c,i,m})}{\sum_{m'=1}^{M} \alpha_{c,i,m'} \cdot \mathcal{N}(\mathbf{x}^n_t|\boldsymbol{\mu}_{c,i,m'}, \mathbf{\Sigma}_{c,i,m'})}. \end{aligned} \tag{3.7}$$

This is essentially the probability that at time-step t the HMM is in state $q_t = i$ and that the m^{th} component of $b_{c,i}(\mathbf{x}^n_t)$ produces $\mathbf{x}^n_t$ given the observation sequence $\mathbf{x}^n$ and the model $\mathbf{\Theta}_c$.

- **M-step:**

In the M-step, the estimated quantities from the E-step can be used to update the parameters of the HMM. The updated parameters $\bar{\mathbf{\Theta}}_c$ are given by

$$\bar{\pi}_{c,i} \leftarrow \frac{1}{N_c} \sum_{n=1}^{N_c} \zeta^n_{c,i,1}, \tag{3.8}$$

$$\bar{a}_{c,i,j} \leftarrow \frac{\sum_{n=1}^{N_c} \sum_{t=1}^{T^n - 1} \xi^n_{c,i,j,t}}{\sum_{n=1}^{N_c} \sum_{t=1}^{T^n} \zeta^n_{c,i,t}}, \tag{3.9}$$

$$\bar{\alpha}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N_c} \sum_{t=1}^{T^n} \zeta^n_{c,i,m,t}}{\sum_{n=1}^{N_c} \sum_{t=1}^{T^n} \zeta^n_{c,i,t}}, \tag{3.10}$$

$$\bar{\boldsymbol{\mu}}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N_c}\sum_{t=1}^{T^n}\zeta_{c,i,m,t}^n \mathbf{x}_t^n}{\sum_{n=1}^{N_c}\sum_{t=1}^{T^n}\zeta_{c,i,m,t}^n}, \tag{3.11}$$

$$\bar{\boldsymbol{\Sigma}}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N_c}\sum_{t=1}^{T^n}\zeta_{c,i,m,t}^n (\mathbf{x}_t^n - \boldsymbol{\mu}_{c,i,m})(\mathbf{x}_t^n - \boldsymbol{\mu}_{c,i,m})^T}{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\zeta_{c,i,m,t}^n}. \tag{3.12}$$

3.2 Conditional log-Likelihood Parameter Estimation

In contrast to generative methods, discriminative training of an HMM $\boldsymbol{\Theta}_c$ of class $c \in \{1,\ldots,C\}$ involves all training samples $\mathcal{X} = \{\mathcal{X}_1, \mathcal{X}_2 \ldots, \mathcal{X}_C\}$, where $N = \sum_{c=1}^{C} N_c$. The conditional log-likelihood (CLL) is given by

$$CLL(\mathcal{X}|\boldsymbol{\Theta}) = \log \prod_{n=1}^{N} p(c^n|\mathbf{x}^n) = \log \prod_{n=1}^{N} \frac{p(\mathbf{x}^n|\boldsymbol{\Theta}_{c^n})\rho_{c^n}}{\sum_{c'=1}^{C} p(\mathbf{x}^n|\boldsymbol{\Theta}_{c'})\rho_{c'}}. \tag{3.13}$$

Maximizing the CLL criterion is akin to the problem of maximum mutual information estimation (MMIE) [8,9] and can be maximized by gradient-based optimization methods, e.g. the extended Baum-Welch (EBW) algorithm [10].

3.2.1 The Extended Baum-Welch Algorithm

The EBW algorithm can be used to optimize rational objective functions

$$R(\boldsymbol{\Phi}) = \frac{\mathrm{Num}(\boldsymbol{\Phi})}{\mathrm{Den}(\boldsymbol{\Phi})}, \tag{3.14}$$

where $\boldsymbol{\Phi} = \{\varphi_{i,j}\}$ is a set of discrete probability parameters constrained by

$$\varphi_{i,j} \geq 0 \quad , \qquad \sum_j \varphi_{i,j} = 1 \quad \text{and} \qquad 1 \leq i,j \leq S. \tag{3.15}$$

The EBW update equation for discrete probability parameters is given by [10]

$$\bar{\varphi}_{i,j} \leftarrow \frac{\varphi_{i,j}\left(\frac{\partial \log R(\boldsymbol{\Phi})}{\partial \varphi_{i,j}} + D\right)}{\sum_j \varphi_{i,j}\left(\frac{\partial \log R(\boldsymbol{\Phi})}{\partial \varphi_{i,j}} + D\right)}, \tag{3.16}$$

where D is a constant value. Setting D is not trivial and is discussed in Section (3.2.5). The conditional log-likelihood is a rational function of the form in Eq. (3.14) over the discrete parameters ρ, π, a and α, thus for these parameters the update of Eq. (3.16) can be used. In the further notation, φ is used synonymously for discrete HMM parameters, i.e. $\varphi \in \{\rho, \pi, a, \alpha\}$.

3.2.2 EBW Update Rules for Discrete Parameters

Given the definition of the conditional log-likelihood in Eq. (3.13), and by approximating $p(\mathbf{x}|\mathbf{\Theta}_c)$ with the probability of the most probable state sequence of the Viterbi algorithm, i.e.

$$p(\mathbf{x}|\mathbf{\Theta}_c) \approx p^*(\mathbf{x}|\mathbf{\Theta}_c) = \max_{1 \le i \le S} \delta_{i,T}, \tag{3.17}$$

the CLL can be formulated as

$$\begin{aligned} CLL(\mathcal{X}|\mathbf{\Theta}) &= \log \prod_{n=1}^{N} p(c^n|\mathbf{x}^n) = \log \prod_{n=1}^{N} \frac{p(\mathbf{x}^n|\mathbf{\Theta}_{c^n})\rho_{c^n}}{\sum_{c'=1}^{C} p(\mathbf{x}^n|\mathbf{\Theta}_{c'})\rho_{c'}} \\ &= \sum_{n=1}^{N} \Bigg[\log \pi_{c^n, i^{*,n}_{c^n,1}} + \sum_{t=1}^{T^n} \log b_{c^n, i^{*,n}_{c^n,t}}(\mathbf{x}^n_t) + \sum_{t=2}^{T^n} \log a_{c^n, i^{*,n}_{c^n,t-1}, i^{*,n}_{c^n,t}} + \log \rho_{c^n} \\ &\quad - \log \Bigg[\sum_{c'=1}^{C} \pi_{c', i^{*,n}_{c',1}} \prod_{t=1}^{T^n} b_{c', i^{*,n}_{c',t}}(\mathbf{x}^n_t) \prod_{t=2}^{T^n} a_{c', i^{*,n}_{c',t-1}, i^{*,n}_{c',t}} \rho_{c'} \Bigg] \Bigg], \end{aligned} \tag{3.18}$$

where $i^{*,n}_{c,t}$ is the most probable state of the HMM of class c for a sequence $\mathbf{x}^n$ at time t (corresponds to q^*_t in Chapter (2)).

The derivative of the CLL w.r.t. ρ_c is

$$\begin{aligned} &\frac{\partial CLL(\mathcal{X}|\mathbf{\Theta})}{\partial \rho_c} = \\ &= \sum_{n=1}^{N} \Bigg[\frac{\mathbb{1}_{\{c=c^n\}}}{\rho_c} - \frac{\pi_{c, i^{*,n}_{c,1}} \prod_{t=1}^{T^n} b_{c, i^{*,n}_{c,t}}(\mathbf{x}^n) \prod_{t=2}^{T^n} a_{c, i^{*,n}_{c,t-1}, i^{*,n}_{c,t}}}{\sum_{c'=1}^{C} \pi_{c', i^{*,n}_{c',1}} \prod_{t=1}^{T^n} b_{c', i^{*,n}_{c',t}}(\mathbf{x}^n) \prod_{t=2}^{T^n} a_{c', i^{*,n}_{c',t-1}, i^{*,n}_{c',t}} \rho_{c'}} \Bigg] \\ &= \sum_{n=1}^{N} \Bigg[\frac{\mathbb{1}_{\{c=c^n\}}}{\rho_c} - \frac{p(\mathbf{x}^n|\mathbf{\Theta}_c)}{\sum_{c'=1}^{C} p(\mathbf{x}^n|\mathbf{\Theta}_{c'})\rho_{c'}} \frac{\rho_c}{\rho_c} \Bigg] \\ &= \frac{1}{\rho_c} \sum_{n=1}^{N} \big(\mathbb{1}_{\{c=c^n\}} - p(c|\mathbf{x}^n) \big) \\ &= \frac{1}{\rho_c} \sum_{n=1}^{N} \big(q^n_c - w^n_c \big), \end{aligned} \tag{3.19}$$

where $w_c^n = p(c|\mathbf{x}^n) = \frac{p(\mathbf{x}^n|\mathbf{\Theta}_c)}{\sum_{c'=1}^{C} p(\mathbf{x}^n|\mathbf{\Theta}_{c'})\rho_{c'}}$ and $q_c^n = \mathbb{1}_{\{c=c^n\}}$ is the indicator function

$$\mathbb{1}_{\{statement\}} = \begin{cases} 1, & \text{if } statement \text{ is true,} \\ 0, & \text{if } statement \text{ if false.} \end{cases} \tag{3.20}$$

In this case $q_c^n = 1$ if the sequence $\mathbf{x}^n$ of class $c^n = c$. The partial derivatives of $CLL(\mathcal{X}|\mathbf{\Theta})$ with respect to π, a and α are determined analogously:

$$\frac{\partial CLL(\mathcal{X}|\mathbf{\Theta})}{\partial \pi_{c,i}} = \frac{1}{\pi_{c,i}} \sum_{n=1}^{N} \left[u_{c,i,1}^n - v_{c,i,1}^n \cdot w_c^n \right] \tag{3.21}$$

$$\frac{\partial CLL(\mathcal{X}|\mathbf{\Theta})}{\partial \alpha_{c,i,m}} = \frac{1}{\alpha_{c,i,m}} \sum_{n=1}^{N} \sum_{t=1}^{T^n} \left[\gamma_{c,i,m,t}^n \left(u_{c,i,t}^n - v_{c,i,t}^n \cdot w_c^n \right) \right] \tag{3.22}$$

$$\frac{\partial CLL(\mathcal{X}|\mathbf{\Theta})}{\partial a_{c,i,j}} = \frac{1}{a_{c,i,j}} \sum_{n=1}^{N} \left[y_{c,i,j}^n - z_{c,i,j}^n \cdot w_c^n \right], \tag{3.23}$$

where

$$u_{c,i,t}^n = \mathbb{1}_{\{c=c^n, i=i_{c,t}^{*,n}\}} \tag{3.24}$$

$$v_{c,i,t}^n = \mathbb{1}_{\{i=i_{c,t}^{*,n}\}} \tag{3.25}$$

$$y_{c,i,j}^n = \sum_{t=2}^{T^n} \mathbb{1}_{\{c=c^n, i=i_{c,t-1}^{*,n}, j=i_{c,t}^{*,n}\}} \tag{3.26}$$

$$z_{c,i,j}^n = \sum_{t=2}^{T^n} \mathbb{1}_{\{i=i_{c,t-1}^{*,n}, j=i_{c,t}^{*,n}\}} \tag{3.27}$$

and

$$\gamma_{c,i,m,t}^n = \frac{b_{c,i,m}(\mathbf{x}_t^n)}{b_{c,i}(\mathbf{x}_t^n)} = \frac{\alpha_{c,i,m} \cdot \mathcal{N}(\mathbf{x}_t^n|\boldsymbol{\mu}_{c,i,m}, \mathbf{\Sigma}_{c,i,m})}{\sum_{m'=1}^{M} \alpha_{c,i,m'} \cdot \mathcal{N}(\mathbf{x}_t^n|\boldsymbol{\mu}_{c,i,m'}, \mathbf{\Sigma}_{c,i,m'})}. \tag{3.28}$$

3.2.3 Approximation of the Gradient

In practice, computing the gradient can cause numerical problems. Merialdo discovered in [11] that certain low values of the parameters $\varphi_{i,j}$ in Eq. (3.16) can cause high values of the gradient. Therefore, training would concentrate on low-valued parameters. However, small parameter values indicate that they are rarely used during the production of an observation sequence. Hence, there is not sufficient training data for estimating very low probabilities, and concentrating on low-valued parameters would be unreliable. For gradients of the form $\frac{\partial R(\mathbf{\Theta})}{\partial \varphi_{i,j}} = \frac{1}{\varphi_{i,j}}(c_{i,j} - c'_{i,j})$, as in our case, he suggests to concentrate on high-valued parameters by replacing the gradient by

$$\frac{\partial R(\mathbf{\Theta})}{\partial \varphi_{i,j}} \approx \frac{c_{i,j}}{\sum_j c_{i,j}} - \frac{c'_{i,j}}{\sum_j c'_{i,j}}. \tag{3.29}$$

The derivatives of the conditional log-likelihood with respect to the discrete HMM parameters using the approximation in Eq. (3.29) are

$$\begin{aligned}\frac{\partial CLL(\mathcal{X}|\mathbf{\Theta})}{\partial \rho_c} &\approx \frac{\sum_{n=1}^{N} q_c^n}{\sum_{c'=1}^{C}\sum_{n=1}^{N} q_{c'}^n} - \frac{\sum_{n=1}^{N} w_c^n}{\sum_{c'=1}^{C}\sum_{n=1}^{N} w_{c'}^n} \\ &= \frac{1}{N}\sum_{n=1}^{N}\left[q_c^n - w_c^n\right]\end{aligned} \tag{3.30}$$

$$\begin{aligned}\frac{\partial CLL(\mathcal{X}|\mathbf{\Theta})}{\partial \pi_{c,i}} &\approx \frac{\sum_{n=1}^{N} u_{c,i,1}^n}{\sum_{i'=1}^{S}\sum_{n=1}^{N} u_{c,i',1}^n} - \frac{\sum_{n=1}^{N} v_{c,i,1}^n \cdot w_c^n}{\sum_{i'=1}^{S}\sum_{n=1}^{N} v_{c,i',1}^n \cdot w_c^n} \\ &= \frac{\sum_{n=1}^{N} u_{c,i,1}^n}{\sum_{n=1}^{N} q_c^n} - \frac{\sum_{n=1}^{N} v_{c,i,1}^n \cdot w_c^n}{\sum_{n=1}^{N} q_c^n \cdot w_c^n}\end{aligned} \tag{3.31}$$

$$\begin{aligned}\frac{\partial CLL(\mathcal{X}|\mathbf{\Theta})}{\partial a_{c,i,j}} &\approx \frac{\sum_{n=1}^{N} y_{c,i,j}^n}{\sum_{j'=1}^{S}\sum_{n=1}^{N} y_{c,i,j'}^n} - \frac{\sum_{n=1}^{N} z_{c,i,j}^n \cdot w_c^n}{\sum_{j'=1}^{S}\sum_{n=1}^{N} z_{c,i,j'}^n \cdot w_c^n} \\ &= \frac{\sum_{n=1}^{N} y_{c,i,j}^n}{\sum_{n=1}^{N}\sum_{t=1}^{T^n-1} q_c^n \cdot v_{c,i,t}^n} - \frac{\sum_{n=1}^{N} z_{c,i,j}^n \cdot w_c^n}{\sum_{n=1}^{N}\sum_{t=1}^{T^n-1} v_{c,i,t}^n \cdot w_c^n}\end{aligned} \tag{3.32}$$

$$\frac{\partial CLL(\mathcal{X}|\boldsymbol{\Theta})}{\partial \alpha_{c,i,m}} \approx \frac{\sum_{n=1}^{N}\sum_{t=1}^{T^n} u_{c,i,t}^n \cdot \gamma_{c,i,m,t}^n}{\sum_{m'=1}^{M}\sum_{n=1}^{N}\sum_{t=1}^{T^n} \gamma_{c,i,m',t}^n \cdot u_{c,i,t}^n} - \frac{\sum_{n=1}^{N}\sum_{t=1}^{T^n} v_{c,i,t}^n \cdot \gamma_{c,i,m,t}^n \cdot w_c^n}{\sum_{m'=1}^{M}\sum_{n=1}^{N}\sum_{t=1}^{T^n} v_{c,i,t}^n \cdot \gamma_{c,i,m',t}^n \cdot w_c^n}$$
$$= \frac{\sum_{n=1}^{N}\sum_{t=1}^{T^n} u_{c,i,t}^n \cdot \gamma_{c,i,m,t}^n}{\sum_{n=1}^{N}\sum_{t=1}^{T^n} u_{c,i,t}^n} - \frac{\sum_{n=1}^{N}\sum_{t=1}^{T^n} v_{c,i,t}^n \cdot \gamma_{c,i,m,t}^n \cdot w_c^n}{\sum_{n=1}^{N}\sum_{t=1}^{T^n} v_{c,i,t}^n \cdot w_c^n} \tag{3.33}$$

3.2.4 Approximation of the Gaussian Mixture Model

The EBW algorithm requires discrete probability distributions, arising the need of a discrete approximation of Gaussian distributions. Assuming diagonal covariances, such an approximation exists [12] and leads to the following update rules for $\boldsymbol{\mu}_{c,i,m}$ and $\boldsymbol{\Sigma}_{c,i,m}$:

$$\bar{\boldsymbol{\mu}}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left[\gamma_{c,i,m,t}^n\left(u_{c,i,t}^n - v_{c,i,t}^n \cdot w_c^n\right)\mathbf{x}_t^n\right] + \boldsymbol{\mu}_{c,i,m} \cdot D}{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left[\gamma_{c,i,m,t}^n\left(u_{c,i,t}^n - v_{c,i,t}^n \cdot w_c^n\right)\right] + D} \tag{3.34}$$

and

$$\bar{\boldsymbol{\Sigma}}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left[\gamma_{c,i,m,t}^n\left(u_{c,i,t}^n - v_{c,i,t}^n \cdot w_c^n\right)(\mathbf{x}_t^n)^2\right] + \left(\boldsymbol{\Sigma}_{c,i,m} + (\boldsymbol{\mu}_{c,i,m})^2\right) \cdot D}{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left[\gamma_{c,i,m,t}^n\left(u_{c,i,t}^n - v_{c,i,t}^n \cdot w_c^n\right)\right] + D} - (\bar{\boldsymbol{\mu}}_{c,i,m})^2, \tag{3.35}$$

where the squares of $\mathbf{x}_t^n$ and $\boldsymbol{\mu}_{c,i,m}$ are taken element-wise.

3.2.5 Implementation of the CL-HMM EBW Algorithm

Setting the constant D is crucial. Too high values of D slow down convergence whereas too small values won't lead to updates that increase the objective function [9, 13]. In [14], an iterative scheme is applied. An initial value of $D = 1$ is doubled until all variances in Eq. (3.35) are positive. The result of this procedure is then multiplied by a convergence-regulating constant factor F. We propose to initialize D by a small value that guarantees a positive parameter update in Eq. (3.16), i.e. $D \leftarrow 1 + \left|\min_{i,j} \frac{\partial R(\boldsymbol{\Phi})}{\partial \varphi_{i,j}}\right|$. A detailed pseudo code of CLL training for HMM is given in Appendix (A).

3.3 Maximium Margin Parameter Estimation

The multi-class margin [15] of sample $n \in \{1, \ldots, N\}$ is given by

$$\tilde{d}^n_{\boldsymbol{\Theta}} = \min_{c \neq c^n} \frac{p(c^n|\mathbf{x}^n, \boldsymbol{\Theta})}{p(c|\mathbf{x}^n, \boldsymbol{\Theta})} = \min_{c \neq c^n} \frac{p(c^n, \mathbf{x}^n|\boldsymbol{\Theta})}{p(c, \mathbf{x}^n|\boldsymbol{\Theta})} = \frac{p(c^n, \mathbf{x}^n|\boldsymbol{\Theta})}{\max_{c \neq c^n} p(c, \mathbf{x}^n|\boldsymbol{\Theta})}. \tag{3.36}$$

Hence, a correctly classified time series sample has a margin $\tilde{d}^n_{\boldsymbol{\Theta}} > 1$.
The EBW algorithm requires a differentiable objective function. Therefore, the max-operator is replaced by a differentiable approximation [14] $\max_x f(x) \approx \left[\sum_x \left(f(x)\right)^\eta\right]^{\frac{1}{\eta}}$, where $\eta \geq 1$ and $f(x)$ is non-negative. This leads to the approximation of $\tilde{d}^n_{\boldsymbol{\Theta}}$, i.e.

$$d^n_{\boldsymbol{\Theta}} = \frac{p(c^n, \mathbf{x}^n|\boldsymbol{\Theta})}{\left[\sum\limits_{c \neq c^n} p(c, \mathbf{x}^n|\boldsymbol{\Theta})^\eta\right]^{\frac{1}{\eta}}} = \frac{p(\mathbf{x}^n|\boldsymbol{\Theta}_{c^n})\rho_{c^n}}{\left[\sum\limits_{c \neq c^n} (p(c, \mathbf{x}^n|\boldsymbol{\Theta}_c)\rho_c)^\eta\right]^{\frac{1}{\eta}}}. \tag{3.37}$$

A common way is to maximize the smallest margin, i.e. $\min_n d^n_{\boldsymbol{\Theta}}$ [15, 16]. Pernkopf and Wohlmayr [14] relax this constraint of the smallest margin by applying a *soft margin*. They concentrate on samples with a margin close to one, using the *hinge loss* function. This leads to the objective

$$\tilde{D}(\mathcal{X}|\boldsymbol{\Theta}) = \prod_{n=1}^{N} \min\left[2, (d^n_{\boldsymbol{\Theta}})^\lambda\right], \tag{3.38}$$

where λ is a scaling parameter of the margin and $\tilde{h}(y) = \min[2, y]$ denotes the hinge loss. Setting λ can be done by cross-validation. Due to the discontinuity of Eq. (3.38), they propose to replace the hinge $\tilde{h}(y)$ function by the following differentiable *smooth hinge* function:

$$h(y) = \begin{cases} y + \frac{1}{2}, & \text{if } y \leq 1, \\ 2 - \frac{1}{2}(y-2)^2, & \text{if } 1 < y < 2, \\ 2, & \text{if } y \geq 2. \end{cases} \tag{3.39}$$

Fig. (3.1) shows the dependency of $h((d^n_{\mathbf{\Theta}})^\lambda)$ on λ:

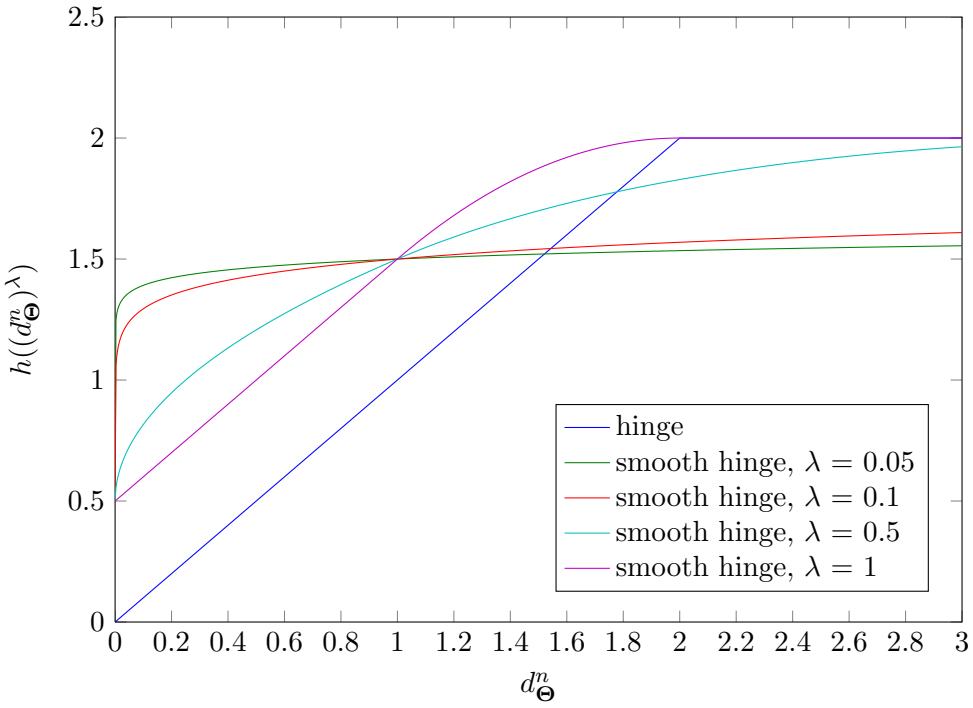


Figure 3.1: Dependency of $h(d^n_{\mathbf{\Theta}})$ on λ.

For the sake of replacing $\tilde{h}(y) = \min\left[2, (d^n_{\mathbf{\Theta}})^\lambda\right]$ in Eq. (3.38) by $h(y)$, the set of samples $\mathcal{X}$ must be divided into three partitions, dependent on the value of $y = (d^n_{\mathbf{\Theta}})^\lambda$. The resulting differentiable objective function is

$$D(\mathcal{X}|\mathbf{\Theta}) = \prod_{n=1}^{N} h\big((d^n_{\mathbf{\Theta}})^\lambda\big) = \left\{ \prod_{n\in\mathcal{X}^1} \left((d^n_{\mathbf{\Theta}})^\lambda + \frac{1}{2} \right) \right\} \left\{ \prod_{n\in\mathcal{X}^2} \left(2 - \frac{1}{2}\big((d^n_{\mathbf{\Theta}})^\lambda - 2\big)^2 \right) \right\} 2^{|\mathcal{X}^3|}. \tag{3.40}$$

3.3.1 EBW Update Rules for Discrete Parameters

The EBW update rule requires the derivative of the $\log D(\mathcal{X}|\boldsymbol{\Theta})$. This derivative with respect to $\boldsymbol{\Theta}$ is

$$\frac{\partial \log D(\mathcal{X}|\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} = \sum_{n=1}^{N} s^n \frac{\partial \log d^n_{\boldsymbol{\Theta}}}{\partial \boldsymbol{\Theta}}, \tag{3.41}$$

where s^n is a weight depending on the partition $\mathcal{X}^p$ of sample $\mathbf{x}^n$ given by

$$s^n = \begin{cases} \frac{\lambda (d^n_{\boldsymbol{\Theta}})^{\lambda}}{(d^n_{\boldsymbol{\Theta}})^{\lambda} + \frac{1}{2}}, & \text{if } \mathbf{x}^n \in \mathcal{X}^1, \\ \frac{\lambda \left(2 - (d^n_{\boldsymbol{\Theta}})^{\lambda}\right)}{2 - \frac{1}{2}(d^n_{\boldsymbol{\Theta}})^{\lambda}}, & \text{if } \mathbf{x}^n \in \mathcal{X}^2, \\ 0, & \text{if } \mathbf{x}^n \in \mathcal{X}^3. \end{cases} \tag{3.42}$$

The log of the margin $d^n_{\boldsymbol{\Theta}}$ of sample $\mathbf{x}^n$ in Eq. (3.37) decomposes to

$$\begin{aligned} \log d^n_{\boldsymbol{\Theta}} &= \log(p(\mathbf{x}^n|\boldsymbol{\Theta}_{c^n})\rho_{c^n}) - \frac{1}{\eta}\log \sum_{c' \neq c^n} (p(c', \mathbf{x}^n|\boldsymbol{\Theta}_{c'})\rho_{c'})^{\eta} \\ &= \log \pi_{c^n, i^{*,n}_{c^n,1}} + \sum_{t=1}^{T^n} \log b_{c, i^{*,n}_{c^n,t}}(\mathbf{x}^n_t) + \sum_{t=2}^{T^n} \log a_{c, i^{*,n}_{c^n,t-1}, i^{*,n}_{c^n,t}} + \log \rho_{c^n} \\ &- \frac{1}{\eta}\log \Bigg[\sum_{c' \neq c^n}^{C} \left(\pi_{c', i^{*,n}_{c',1}} \prod_{t=1}^{T^n} b_{c', i^{*,n}_{c',t}}(\mathbf{x}^n_t) \prod_{t=2}^{T^n} a_{c', i^{*,n}_{c',t-1}, i^{*,n}_{c',t}} \rho_{c'} \right)^{\eta} \Bigg], \end{aligned} \tag{3.43}$$

where again $p(\mathbf{x}|\boldsymbol{\Theta}_c)$ is approximated by $p^*(\mathbf{x}|\boldsymbol{\Theta}_c)$, determined by the Viterbi algorithm, see Eq. (3.17). Analogously to Eq. (3.19), the derivative $\frac{\partial \log d^n_{\boldsymbol{\Theta}}}{\partial \rho_c}$ is

$$\begin{aligned} \frac{\partial \log d^n_{\boldsymbol{\Theta}}}{\partial \rho_c} &= \frac{\mathbb{1}_{\{c=c^n\}}}{\rho_c} - \frac{\mathbb{1}_{\{c \neq c^n\}} (p(\mathbf{x}^n|\boldsymbol{\Theta}_c)\rho_c)^{\eta-1} p(\mathbf{x}^n|\boldsymbol{\Theta}_c)}{\sum_{c' \neq c^n} (p(\mathbf{x}^n|\boldsymbol{\Theta}_{c'})\rho_{c'})^{\eta}} \frac{\rho_c}{\rho_c} \\ &= \frac{1}{\rho_c}\left[\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} \frac{(p(\mathbf{x}^n|\boldsymbol{\Theta}_c)\rho_c)^{\eta}}{\sum_{c' \neq c^n} (p(\mathbf{x}^n|\boldsymbol{\Theta}_{c'})\rho_{c'})^{\eta}} \right] \\ &= \frac{1}{\rho_c}\left[q^n_c - \check{q}^n_c \cdot r^{n,\eta}_c \right], \end{aligned} \tag{3.44}$$

where

$$r^{n,\eta}_c = \frac{(p(\mathbf{x}^n|\boldsymbol{\Theta}_c)\rho_c)^{\eta}}{\sum_{c' \neq c^n} (p(\mathbf{x}^n|\boldsymbol{\Theta}_{c'})\rho_{c'})^{\eta}}, \quad q^n_c = \mathbb{1}_{\{c=c^n\}} \quad \text{and} \quad \check{q}^n_c = \mathbb{1}_{\{c \neq c^n\}}. \tag{3.45}$$

The partial derivatives of $\log d^n_{\boldsymbol{\Theta}}$ with respect to π_i, $a_{c,i,j}$ and $\alpha_{c,i,m}$ are straightforward to determine:

$$\frac{\partial \log d^n_{\boldsymbol{\Theta}}}{\partial \pi_{c,i}} = \frac{1}{\pi_{c,i}} \left[u^n_{c,i,1} - \check{u}^n_{c,i,1} \cdot r_c^{n,\eta} \right] \tag{3.46}$$

$$\frac{\partial \log d^n_{\boldsymbol{\Theta}}}{\partial a_{c,i,j}} = \frac{1}{a_{c,i,j}} \left[y^n_{c,i,j} - \check{y}^n_{c,i,j} \cdot r_c^{n,\eta} \right] \tag{3.47}$$

$$\frac{\partial \log d^n_{\boldsymbol{\Theta}}}{\partial \alpha_{c,i,m}} = \frac{1}{\alpha_{c,i,m}} \sum_{t=1}^{T^n} \left[\gamma^n_{c,i,m,t} \left(u^n_{c,i,t} - \check{u}^n_{c,i,t} \cdot r_c^{n,\eta} \right) \right], \tag{3.48}$$

where

$$u^n_{c,i,t} = \mathbb{1}_{\{c=c^n, i=i^{*,n}_{c,t}\}} \tag{3.49}$$

$$\check{u}^n_{c,i,t} = \mathbb{1}_{\{c\neq c^n, i=i^{*,n}_{c,t}\}} \tag{3.50}$$

$$y^n_{c,i,j} = \sum_{t=2}^{T^n} \mathbb{1}_{\{c=c^n, i=i^{*,n}_{c,t-1}, j=i^{*,n}_{c,t}\}} \tag{3.51}$$

$$\check{y}^n_{c,i,j} = \sum_{t=2}^{T^n} \mathbb{1}_{\{c\neq c^n, i=i^{*,n}_{c,t-1}, j=i^{*,n}_{c,t}\}}. \tag{3.52}$$

3.3.2 Approximation of the Gradient

Unfortunately, approximating the gradient by Eq. (3.29) cannot be applied to the derivatives of the margin, because the approximated gradient would vanish for any HMM parameter. For instance, Eq. (3.53) shows Merialdo's approximation to the derivative of the margin with respect to the HMM parameter $a_{c,i,j}$:

$$\begin{aligned}
\frac{\partial \log d^n_{\boldsymbol{\Theta}}}{\partial a_{c,i,j}} &\approx \frac{y^n_{c,i,j}}{\sum_{j'=1}^{S} y^n_{c,i,j'}} - \frac{y^n_{c,i,j} r_c^{n,\eta}}{\sum_{j'=1}^{S} y^n_{c,i,j'} r_c^{n,\eta}} \\
&= \frac{z^n_{c,i,j} \mathbb{1}_{\{c=c^n\}}}{\sum_{j'=1}^{S} z^n_{c,i,j'} \mathbb{1}_{\{c=c^n\}}} - \frac{z^n_{c,i,j} \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}}{\sum_{j'=1}^{S} z^n_{c,i,j'} \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}} \\
&= \frac{z^n_{c,i,j} \mathbb{1}_{\{c=c^n\}}}{\mathbb{1}_{\{c=c^n\}} \sum_{t=1}^{T^n-1} \mathbb{1}_{\{i=i^{*,n}_{c,t}\}}} - \frac{z^n_{c,i,j} \mathbb{1}_{\{c\neq c^n\}}}{\mathbb{1}_{\{c\neq c^n\}} \sum_{t=1}^{T^n-1} \mathbb{1}_{\{i=i^{*,n}_{c,t}\}}} \\
&= \frac{z^n_{c,i,j}}{\sum_{t=1}^{T^n-1} \mathbb{1}_{\{i=i^{*,n}_{c,t}\}}} - \frac{z^n_{c,i,j}}{\sum_{t=1}^{T^n-1} \mathbb{1}_{\{i=i^{*,n}_{c,t}\}}} = 0
\end{aligned} \tag{3.53}$$

Therefore, an alternative approximation is suggested in order to obtain reliable parameter updates. As the unreliability of the updates is caused by small parameter values due to high values of the gradients [11], normalizing the gradient by a sum-to-one constraint might keep the updates reliable. For gradients of the form $\frac{\partial R(\boldsymbol{\Theta})}{\partial \varphi_{i,j}} = \frac{1}{\varphi_{i,j}}(c_{i,j} - c'_{i,j})$, we propose to approximate the gradient by

$$\frac{\partial R(\boldsymbol{\Theta})}{\partial \varphi_{i,j}} \approx \begin{cases} \frac{\frac{1}{\varphi_{i,j}}(c_{i,j}-c'_{i,j})}{\sum_{j'=1}^{S}\left|\frac{1}{\varphi_{i,j'}}(c_{i,j'}-c'_{i,j'})\right|}, & \text{if } \sum_{j'=1}^{S}\left|\frac{\partial R(\boldsymbol{\Theta})}{\partial \varphi_{i,j'}}\right| > 1 \\ \frac{1}{\varphi_{i,j}}(c_{i,j} - c'_{i,j}), & \text{otherwise} \end{cases} . \tag{3.54}$$

As an alternative to the parameter update rule in Eq. (3.16), for gradients of the form $\frac{\partial R(\boldsymbol{\Theta})}{\partial \varphi_{i,j}} = \frac{1}{\varphi_{i,j}}(c_{i,j} - c'_{i,j})$, Woodland and Povey [9] propose to update the parameters using a constrained nonlinear optimization problem. They suggest to use a generic function-optimization routine, or, as they do in [9], to use an iterative procedure.

3.3.3 Approximation of the Gaussian Mixture Model

For the continuous parameters of the GMMs, the discrete approximation described in Sec. (3.2.4) can be used. Hence, the update rules for $\boldsymbol{\mu}_{c,i,m}$ and $\boldsymbol{\Sigma}_{c,i,m}$ are

$$\bar{\boldsymbol{\mu}}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N} s^n \sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\big(u^n_{c,i,t} - \check{u}^n_{c,i,t}\cdot r_c^{n,\eta}\big)\mathbf{x}^n_t\right] + \boldsymbol{\mu}_{c,i,m}\cdot D}{\sum_{n=1}^{N} s^n \sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\big(u^n_{c,i,t} - \check{u}^n_{c,i,t}\cdot r_c^{n,\eta}\big)\right] + D} \tag{3.55}$$

and

$$\bar{\boldsymbol{\Sigma}}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N} s^n \sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\big(u^n_{c,i,t} - \check{u}^n_{c,i,t}\cdot r_c^{n,\eta}\big)(\mathbf{x}^n_t)^2\right] + \big(\boldsymbol{\Sigma}_{c,i,m} + (\boldsymbol{\mu}_{c,i,m})^2\big)\cdot D}{\sum_{n=1}^{N} s^n \sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\big(u^n_{c,i,t} - \check{u}^n_{c,i,t}\cdot r_c^{n,\eta}\big)\right] + D} - (\bar{\boldsymbol{\mu}}_{c,i,m})^2, \tag{3.56}$$

respectively.

3.3.4 Implementation of the MM-HMM EBW Algorithm

The procedure for setting the constant D of EBW is analogous to the procedure for CL-based training. For details, the reader should refer to Sec. (3.2.4). A detailed-pseudo code of maximum margin (MM) training for HMM is provided in Appendix (B).

4

Application of HMMs to Time Series Data Classification

The generative and discriminative parameter estimations techniques presented in the previous chapter are compared in time series classification tasks. In particular, we perform broad phonetic classification and spoken digit classification using the TIMIT corpus. Furthermore, we provide results for handwritten digit classification. 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.

4.1 Experimental Setup

For the experiments conducted in this section, the HMM parameters trained by MLE have been used as initialization for the discriminative methods, i.e. CLL and maximum margin parameter learning (see Alg. (1)) and (5)). The classification rate is used as performance measure. The classification rate (CR) is the ratio of correctly classified to the total number of test samples in [%], given by

$$CR = \frac{100}{N^T} \sum_{n}^{N^T} \mathbb{1}_{\{c^n = c^*\}} = \frac{100}{N^T} \sum_{n}^{N^T} \mathbb{1}_{\{c^n = \arg\max_{1 \leq c \leq C} p(c|\mathbf{x})\}}, \tag{4.1}$$

where N^T denotes the number of test samples. As an estimate of the accuracy of the CR, we state a confidence interval for every result. According to [17], with a probability of approximately 95%, the true classification rate lies within the confidence interval

$$CR \pm 1.96 \sqrt{\frac{CR(100 - CR)}{N^T}}. \tag{4.2}$$

We perform classification using HMMs with varying numbers of mixture components and states. We use $S \in \{2, 3, \ldots, 7\}$ states and $M \in \{2, 3, 4\}$ mixture components. 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 for the convergence-regulating constant F. The minimum value of F that lead to convergence of the objective functions depends individually on the training data and the selected model, i.e. number of states and number of mixture components.
For MM-HMM, we need the margin scaling parameter λ. This parameter was set by 3-fold cross-validation on the training set. Furthermore, the parameter η used for the approximation of the margin in Eq. (3.37) was set to 2.
As convergence is not guaranteed due to the approximation of the objective's gradients, we propose two methods to promote convergence. The first strategy is simply to let the EBW algorithm run for a sufficiently large number of iterations and record the classification performance on a validation set for every iteration. Then, training is repeated and stopped after the number of iterations with the best classification result of the first run. The second method is to stop training when the objective function begins to decrease and use the model parameters of the last training iteration that lead to an increase of the objective function. The latter method was applied in the experiments of this chapter.
When discriminative HMM training is applied, numerical underflow might occur during the estimation of very small HMM parameters, leading to unreliable training results. To overcome this, we suggest to restrict the parameter values to be greater than a minimum value and rescale the parameters due to the sum-to-one constraint of Eq. (3.15).
In practice, discriminative training is limited due to a long training time, depending on the amount of data and the complexity of the model. As for every class the whole training data is required, the training time increases linearly with the number of different classes.

4.2 Speech Classification

4.2.1 The TIMIT Speech Corpus

The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus has been designed under the cooperation of Texas Instruments (TI), the Massachusetts Institute of Technology (MIT) and the Stanford Research Institute (SRI). The database contains a total number of 6300 sentences in American English. From each of the 630 speakers, 10 sentences have been recorded and categorized by the sex of the speakers and by eight dialect regions within the United States of America [18]. The corpus consists of three kinds of sentences: 450 phonetically-compact, 1890 phonetically-diverse and 2 dialect sentences. The latter category is usually left out for training and testing. The utterances are recorded at a 16 kHz sample rate with a 16 bit resolution [19]. For each utterance, a hand-labelled transcription of the complete orthographic text, of single words and of single phonemes are available. The phonetic transcription is based on 61 phonemes. A partition into training and test samples is provided.
For the experiments conducted in this section, the observation vectors $\mathbf{x}_t$ consist of the first 13 mel-frequency cepstral coefficients (MFCCs), computed with a window length of 25 ms at a frame rate of 10 ms, and their first and second derivatives [5]. Thus, an observation $\mathbf{x}_t$ consists of 39 features. Additionally, principal component analysis (PCA) was applied to whiten the data.

4.2.2 Broad Phonetic Classification

For broad phonetic classification, the 61 phonetic labels are collapsed into broad phonetic groups as proposed by Halberstadt [20]. In particular, we have the following classes: Vowel/Semivowel (VS), Nasal/Flap (NF), Strong Fricative (SF), Weak Fricative (WF), Stop (ST) and Closure (CL). In Table (4.1) the labelling of the TIMIT phones used in this work is shown. The dialect sentences have been omitted for training as well for testing. The resulting training data consisted of 140173 sequences. For testing, the 'core' test set was used [18]. It consists of 7211 sequences. For maximum margin training, the margin scaling parameter λ was set to 0.05. In Table (4.2), we present classification results for MLE-HMMs using different numbers of states and mixture components. The boldface entries in each table point out the best result achieved by the corresponding training method.

Phonetic class	**# TIMIT labels**	**Timit labels**
Vowel/Semivowel (VS)	25	aa ae ah ao aw ax axh axr ay eh er ey ih ix iy ow oy uh uw ux el l r w y
Nasal/Flap (NF)	8	m en eng m n ng nx dx
Strong Fricative (SF)	6	s z sh zh ch jh
Weak Fricative (WF)	6	v f dh th hh hv
Stop (ST)	6	b d g p t k
Closure (CL)	9	bcl dcl gcl pcl tcl kcl epi pau h#

Table 4.1: Mapping of TIMIT labels into six broad phonetic groups.

# HMM states	**# mixture components**		
	2	**3**	**4**
2	86.5 ± 0.79	87.1 ± 0.77	88.2 ± 0.75
3	86.8 ± 0.78	88.0 ± 0.75	88.1 ± 0.75
4	86.3 ± 0.79	88.4 ± 0.74	**88.6 ± 0.73**
5	86.7 ± 0.78	87.9 ± 0.75	88.4 ± 0.74
6	87.4 ± 0.77	87.8 ± 0.75	88.3 ± 0.74
7	87.5 ± 0.76	88.0 ± 0.75	88.4 ± 0.74

Table 4.2: Classification rates of MLE-HMMs in [%] on TIMIT broad phonetic classification.

We encountered problems with setting the constant D of EBW. As discussed in Section (3.2.5), after initialization D was repeatedly doubled in order to obtain covariance matrices with non-negative entries (see Algorithms (3) and (7)). For HMMs with particular numbers of states and mixture components, it turned out that no large enough value for D was found, no matter how often the doubling procedure was repeated. Therefore, discriminative training failed on these HMMs. Nevertheless discriminative training worked for particular HMM settings. Tables (4.3) and (4.4) state the results of conditional likelihood and maximum margin training for HMMs with five states and $m = \{2, 3, 4\}$ mixture components.

# HMM states	# mixture components		
	2	**3**	**4**
5	**91.1 ± 0.66**	86.8 ± 0.78	88.0 ± 0.75

Table 4.3: Classification rates of CLL-HMMs in [%] on TIMIT broad phonetic classification.

# HMM states	# mixture components		
	2	**3**	**4**
5	**87.6 ± 0.76**	86.3 ± 0.79	86.9 ± 0.78

Table 4.4: Classification rates of MM-HMMs in [%] on TIMIT broad phonetic classification.

Having a look at Table (4.2), it turns out that the number of HMM states doesn't have much influence classification performance, especially for $m = 4$ mixture components. This leads to the assumption that observations $\mathbf{x}_t$ according to sequences of the same class do not differ much from each other. Hence, phonetic sequences do not provide significant temporal information in order to estimate the HMM state transition probabilities well, i.e. the observation sequences do not resemble a meaningful pattern to be learned. Therefore, classification performance is dominated by the number of mixture components in this task.

4.2.3 Spoken Digit Classification

For this task, the provided transcription of single words was used. A subset of the data, containing utterances of 10 different numbers from 'one' to 'ten' (no utterance for 'zero' is available) has been taken. This set of spoken digits consisted of 165 training and 64 test sequences in total. In contrast to the broad phonetic task, the full test set has been used, as for some classes no test samples exist in the 'core' test set. Again, λ was set to 0.05. Tables (4.5), (4.6) and (4.7) show the classification results for MLE, CLL and MM training, respectively.

# HMM states	# mixture components		
	2	**3**	**4**
2	90.6 ± 7.14	87.5 ± 8.10	85.9 ± 8.51
3	**93.8 ± 5.93**	84.4 ± 8.90	79.7 ± 9.86
4	82.8 ± 9.24	84.4 ± 8.90	75.0 ± 10.61
5	79.7 ± 9.86	76.6 ± 9.86	70.3 ± 11.19
6	82.8 ± 9.24	68.8 ± 11.36	68.8 ± 11.36
7	76.6 ± 10.38	68.8 ± 11.36	67.0 ± 11.50

Table 4.5: Classification rates of MLE-HMMs in [%] on TIMIT spoken digit classification.

# HMM states	# mixture components		
	2	**3**	**4**
2	95.3 ± 5.18	**96.9 ± 4.26**	93.8 ± 5.93
3	90.6 ± 7.14	93.8 ± 5.93	92.2 ± 6.58
4	93.8 ± 5.93	93.8 ± 5.93	90.6 ± 7.14
5	95.3 ± 5.18	89.1 ± 7.65	90.6 ± 7.14
6	90.6 ± 7.14	95.3 ± 5.18	92.2 ± 6.58
7	92.2 ± 6.58	92.2 ± 6.58	92.2 ± 6.58

Table 4.6: Classification rates of CLL-HMMs in [%] on TIMIT digit classification.

# HMM states	# mixture components		
	2	**3**	**4**
2	95.3 ± 5.18	96.9 ± 4.26	93.8 ± 5.93
3	96.9 ± 4.26	93.8 ± 5.93	92.2 ± 6.58
4	**98.4 ± 3.04**	93.8 ± 5.93	92.2 ± 6.58
5	95.3 ± 5.18	90.6 ± 7.14	89.1 ± 7.65
6	96.9 ±4.26	93.8 ± 5.93	92.2 ± 6.58
7	93.8 ± 5.93	92.2 ± 6.58	92.2 ± 6.58

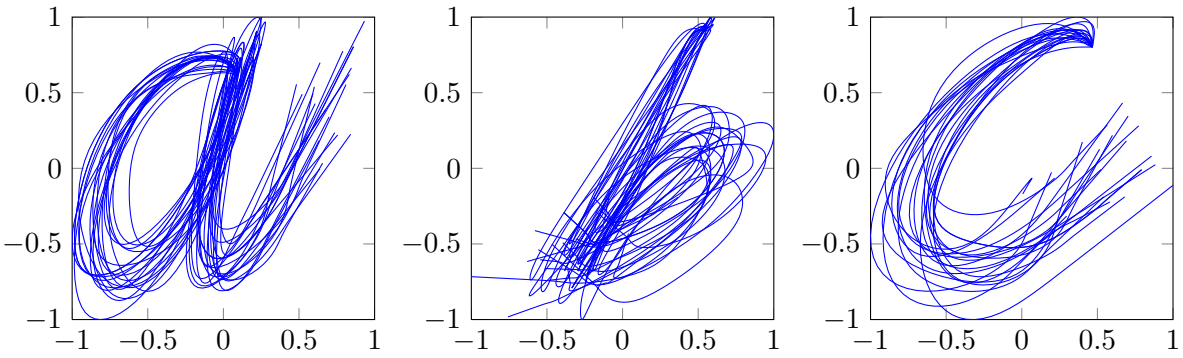
Table 4.7: Classification rates of MM-HMMs in [%] on TIMIT digit classification.

In this task, both discriminative methods perform approximately even well, although MM training achieves the highest classification rate. It is remarkable that the classification performance of MLE, CLL and MM decreases with an increasing number of HMM states and mixture components. 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.

4.3 Handwritten Digit Classification

4.3.1 The Williams Database

The Williams database contains 2858 samples of pen tip trajectories used in [21]. The samples are categorized into 20 different characters and are all from the same writer. The observation vectors $\mathbf{x}_t$ consist of three values: pen velocities in x- and y- direction and pen tip force, captured at a sample rate of 200 Hz. The trajectories have been captured only for characters written in one single stroke [22].
To conduct the experiments, the data was divided into 80% training and 20% test samples. Furthermore, the sample sequences have been compressed to a length of 10 data vectors per trajectory. The compression was done by the following. We partitioned each sample sequence into 10 adjacent sub-sequences and calculated the mean for each sub-sequence. Sample trajectories of the Williams database for the letters 'a', 'b' and 'c' are illustrated in Fig. (4.1). The parameter λ for scaling the margin was set to 0.05. Tables (4.8), (4.9) and (4.10) show the classification performance of MLE-HMM, CLL-HMM and MM-HMM on the Williams database, respectively.



(a) letter 'a' (b) letter 'b' (c) letter 'c'

Figure 4.1: Characters from the Williams database.

# HMM states	# mixture components		
	2	**3**	**4**
2	95.6 ± 1.69	95.8 ± 1.65	96.5 ± 1.51
3	96.1 ± 1.59	97.2 ± 1.36	97.0 ± 1.40
4	96.6 ± 1.48	97.2 ± 1.36	97.3 ± 1.32
5	97.9 ± 1.18	97.7 ± 1.23	97.3 ± 1.32
6	98.2 ± 1.08	98.1 ± 1.13	97.5 ± 1.27
7	98.2 ± 1.08	**98.6 ± 0.97**	97.9 ± 1.18

Table 4.8: Classification rates of MLE-HMMs in [%] on Williams handwritten digit classification.

# HMM states	# mixture components		
	2	**3**	**4**
2	96.5 ± 1.51	97.3 ± 1.32	96.6 ± 1.48
3	98.2 ± 1.08	98.9 ± 0.84	96.6 ± 1.48
4	98.6 ± 0.97	98.6 ± 0.97	99.1 ± 0.77
5	98.9 ± 0.84	98.4 ± 1.03	98.1 ± 1.13
6	99.1 ± 1.13	99.3 ± 0.69	98.8 ± 0.91
7	99.1 ± 0.77	98.8 ± 0.91	**99.5 ± 0.60**

Table 4.9: Classification rates of CLL-HMMs in [%] on Williams handwritten digit classification.

# HMM states	# mixture components		
	2	**3**	**4**
2	91.5 ± 2.28	93.6 ± 2.05	94.9 ± 1.81
3	96.8 ± 1.44	95.4 ± 1.72	95.9 ± 1.62
4	96.6 ± 1.48	96.6 ± 1.48	97.0 ± 1.40
5	97.9 ± 1.18	97.7 ± 1.23	97.3 ± 1.32
6	98.2 ± 1.08	97.5 ± 1.27	98.2 ± 1.08
7	98.2 ± 1.08	98.6 ± 0.07	**98.9 ± 0.84**

Table 4.10: Classification rates of MM-HMMs in [%] on Williams handwritten digit classification.

Although CLL training achieved the best CR on the Williams database, the top performance of all methods lies within a range of 1%. The confidence intervals show that these results are not significantly different. We presume that the true distribution of the given samples closely resembles the distribution modelled by an HMM. In this case, discriminative learning is performing similar to generative learning [4]. The 3-fold cross-validation of the scaling parameter λ for MM training is depicted in Fig. (4.2).

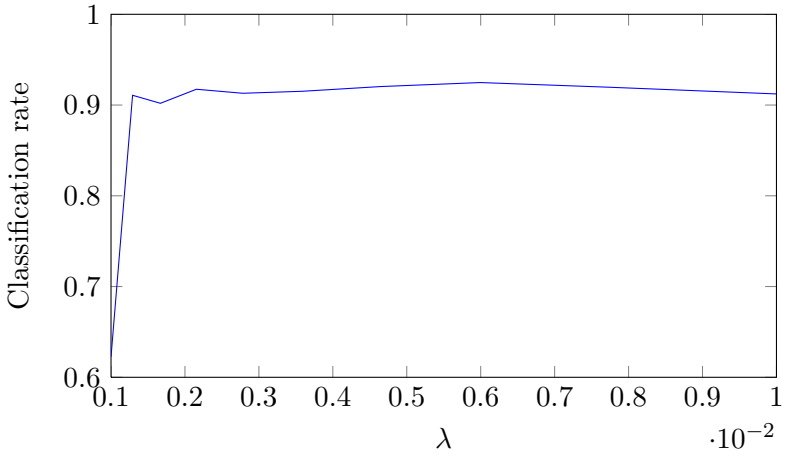
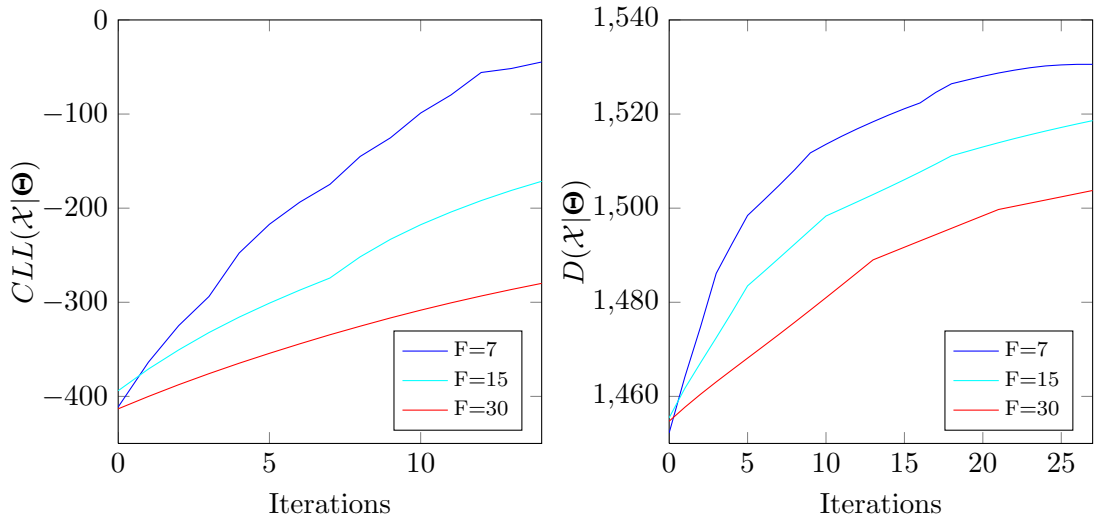


Figure 4.2: 3-fold cross-validation of λ on Williams database.

Fig. (4.3) shows the convergence of the objective function of CLL-HMMs and MM-HMMs for $F = \{7, 15, 30\}$.

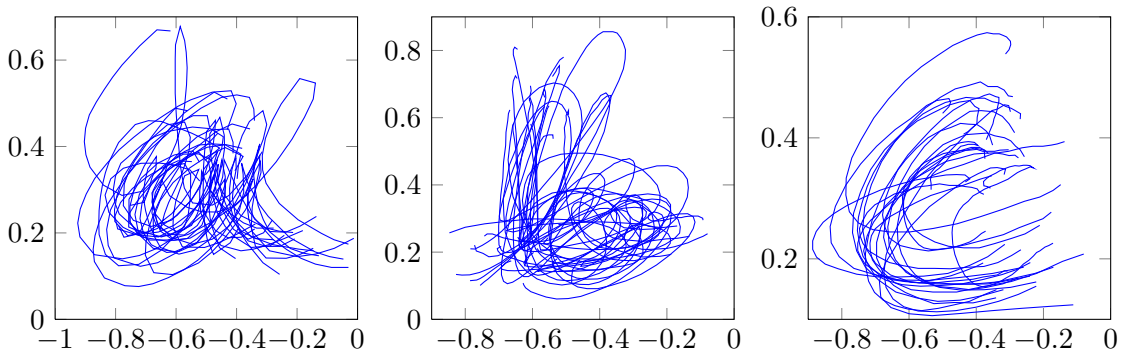


(a) CLL-HMM (b) MM-HMM

Figure 4.3: Convergence of CL-HMM and MM-HMM on Williams database.

4.3.2 The UJI Database

The UJIpenchars2 database contains trajectories of 97 handwritten characters from 60 different writers [23]. The observation vectors $\mathbf{x}_t$ consist of 2 values: absolute x- and y-coordinates. Neither timing information nor pen force was recorded. For each writer, two instances of each symbol were recorded, giving a total number of 11640 samples. A partition into training and test samples is provided.
For the experiments a subset of the data, containing the same 20 characters as the Williams database has been taken. The subset consists of 1830 training and 915 test samples. Additionally, the velocities in x- and y- direction, obtained by numerically differentiating the coordinates, were added to the original data. Thus the observation vectors $\mathbf{x}_t$ used for training and testing consisted of 4 values. For maximum margin training, in this experiment λ was set to 0.1. Sample trajectories of the UJIpenchars2 database for the letters 'a', 'b' and 'c' are illustrated in Fig. (4.4).



(a) letter 'a' (b) letter 'b' (c) letter 'c'

Figure 4.4: Characters from the UJIpenchars2 database.

The classification performances for MLE-HMM, CLL-HMM and MM-HMM are shown in Tables (4.11), (4.12) and (4.13), respectively.

# HMM states	**# mixture components**		
	2	**3**	**4**
2	50.9 ± 3.24	58.5 ± 3.19	65.9 ± 3.07
3	56.8 ± 3.21	64.2 ± 3.11	66.9 ± 3.05
4	58.5 ± 3.19	65.7 ± 3.08	68.9 ± 3.00
5	61.5 ± 3.15	67.3 ± 3.04	69.3 ± 2.99
6	62.8 ± 3.13	**69.5 ± 2.98**	69.4 ± 2.99
7	63.6 ± 3.12	66.4 ± 3.06	68.5 ± 3.01

Table 4.11: Classification rates of MLE-HMMs in [%] on UJIpenchars2 handwritten digit classification.

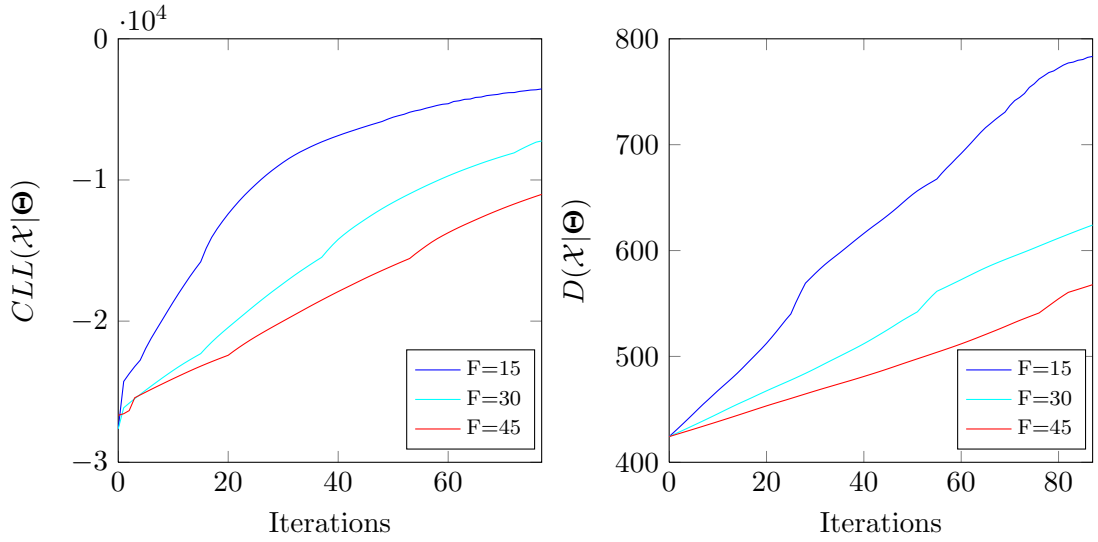
# HMM states	**# mixture components**		
	2	**3**	**4**
2	56.4 ± 3.21	53.6 ± 3.23	67.1 ± 3.04
3	55.7 ± 3.22	63.9 ± 3.11	68.8 ± 3.00
4	66.9 ± 3.05	67.4 ± 3.04	69.8 ± 2.97
5	66.5 ± 3.06	67.5 ± 3.03	70.8 ± 2.95
6	67.9 ± 3.03	**72.2 ± 2.90**	**72.2 ± 2.90**
7	69.2 ± 2.99	71.2 ± 2.94	70.1 ± 2.97

Table 4.12: Classification rates of CLL-HMMs in [%] on UJIpenchars2 handwritten digit classification.

# HMM states	# mixture components		
	2	**3**	**4**
2	59.8 ± 3.18	64.2 ± 3.11	69.1 ± 2.99
3	64.7 ± 3.10	67.7 ± 3.03	69.1 ± 2.99
4	66.5 ± 3.06	66.5 ± 3.06	69.8 ± 2.97
5	67.9 ± 3.03	70.3 ± 2.96	73.1 ± 2.87
6	67.9 ± 3.03	71.9 ± 2.91	**73.3 ± 2.87**
7	68.9 ± 3.00	71.1 ± 2.94	70.4 ± 2.96

Table 4.13: Classification rates of MM-HMMs in [%] on UJIpenchars2 handwritten digit classification.

In this task discriminative training clearly outperforms generative MLE-HMMs. Since there are many different writers the variation is much larger than for the Williams database, as it can be seen in Fig. (4.4) when compared to Fig. (4.1). The great variation of the samples of the same class hardens the challenge of classification for generative MLE-HMMs as well as for discriminative CLL-HMMs and MM-HMMs. Therefore, classification performance on the UJIpenchars2 database is worse for all HMMs compared to the performance on the Williams database. Fig. (4.5) shows the convergence of the objective function of CLL-HMMs and MM-HMMs for $F = \{15, 30, 45\}$.



(a) CLL-HMM (b) MM-HMM

Figure 4.5: Convergence of CL-HMM and MM-HMM on UJIpenchars2 database.

5
Conclusions and Future Work

In this thesis, the discriminative maximum margin learning method of was derived for hidden Markov models and compared to conditional log-likelihood and maximum likelihood parameter estimation for HMMs. In particular, we use the extended Baum-Welch (EBW) framework. The EBW algorithm has been developed to optimize rational functions such as the conditional log-likelihood. We formulate the margin of a sample as the ratio of the class posterior of the true class and the most competing class. This sample margin is embedded into a hinge loss function. Since this objective can not be derived, a smooth approximation has been introduced. The derivatives of this objective function are used in the EBW algorithm to discriminatively optimize the HMM parameters. In the experiments we provided results for the tasks of classifying human speech and handwritten data using maximum likelihood, conditional likelihood and maximum margin parameter learning. Three different data sets were used, showing strengths and weaknesses of the training methods.

The conclusion of this thesis can be summarized by the following:

- Selecting the states and mixture components of an HMM is crucial for a good classification performance.
- If the true distribution of the data to be classified fits the model distribution well, the application of discriminative training does not provide much performance gain compared to using generative learning. This is well known in the machine learning community [4].
- Maximum margin training competes with the conditional likelihood method and slightly outperforms conditional likelihood training in the experiments.
- The implementation of maximum margin training is not trivial, but can achieve good classification performance.

In future work, the following issues are treated:
Currently, the probability of an observation sequence $p(\mathbf{x}|\mathbf{\Theta}_c)$ is approximated by $p^*(\mathbf{x}|\mathbf{\Theta}_c)$, of the Viterbi algorithm. This greatly simplifies the derivatives of the discriminative objective functions, i.e. only one path along the trellis has to be considered instead of the sum of all possible paths. As shown in the experiments of Chapter (4), this simplification seems to lead

to good classification performance of the discriminative methods. Nevertheless, it is of interest to investigate the influence of this approximation on the classification results. Hence, a direct comparison of using $p(\mathbf{x}|\boldsymbol{\Theta}_c)$ and $p^*(\mathbf{x}|\boldsymbol{\Theta}_c)$ in the objective functions are focus of future research. The experiments in this thesis have been conducted by modelling every class by HMMs with the same number of states and mixture components for every class. As different classes might be better modelled by HMMs with individual rather than the same number of states and mixture components, an investigation of this approach is considered in the future.

A

CLL-HMM EBW algorithm

The implementation of the EBW algorithm for maximizing the conditional log-likelihood (CLL-HMM EBW algorithm) is stated in Algorithm (1).

Input: $\{\mathcal{X}_1, \dots \mathcal{X}_C\}$
Output: $\rho_c, \pi_{c,i}, a_{c,i,j}, \{\alpha_{c,i,m}, \boldsymbol{\mu}_{c,i,m}, \boldsymbol{\Sigma}_{c,i,m}\}_{m=1}^{M} \quad \forall c \in \{1, \dots, C\},\ \forall i, j \in \{1, \dots, S\}$
Initialization: For each c, train $\pi_{c,i}, a_{c,i,j}, \{\alpha_{c,i,m}, \boldsymbol{\mu}_{c,i,m}, \boldsymbol{\Sigma}_{c,i,m}\}_{m=1}^{M}$ on $\mathcal{X}_c$, using the EM-algorithm. Set ρ_c to class frequency in $\mathcal{X}$, i.e. $\rho_c \leftarrow \frac{|\mathcal{X}_c|}{|\mathcal{X}|}$
while $CLL(\mathcal{X}|\boldsymbol{\Theta})$ *not converged* **do**
  E-Step (see Algorithm (2))
  Determine D (see Algorithm (3))
  M-Step (see Algorithm (4))
end

Algorithm 1:Discriminative CL-based training of HMMs (CLL-HMM EBW algorithm).

The E-step of the CLL-HMM EBW algorithm using Merialdo's approximation of $\frac{\partial CLL(\mathcal{X}|\boldsymbol{\Theta})}{\partial \varphi}$ (see Eq. (3.29)) is depicted in Algorithm (2).

E-Step:
for $c \leftarrow 1$ **to** C **do**
  $w_c^n \leftarrow \frac{p(\mathbf{x}^n|\boldsymbol{\Theta}_c)\rho_c}{\sum_{c'=1}^{C} p(\mathbf{x}^n|\boldsymbol{\Theta}_{c'})\rho_{c'}} \quad \forall n \in \{1, \dots, N\}$
  $\partial\rho_c \leftarrow \frac{\sum_{n=1}^{N} q_c^n}{\sum_{c'=1}^{C}\sum_{n=1}^{N} q_{c'}^n} - \frac{\sum_{n=1}^{N} w_c^n}{\sum_{c'=1}^{C}\sum_{n=1}^{N} w_{c'}^n}$
  for $i \leftarrow 1$ **to** S **do**
    $\partial\pi_{c,i} \leftarrow \frac{\sum_{n=1}^{N} u_{c,i,1}^n}{\sum_{i'=1}^{S}\sum_{n=1}^{N} u_{c,i',1}^n} - \frac{\sum_{n=1}^{N} v_{c,i,1}^n \cdot w_c^n}{\sum_{i'=1}^{S}\sum_{n=1}^{N} v_{c,i',1}^n \cdot w_{c'}^n}$
    for $j \leftarrow 1$ **to** S **do**
      $\partial a_{c,i,j} \leftarrow \frac{\sum_{n=1}^{N} y_{c,i,j}^n}{\sum_{j'=1}^{S}\sum_{n=1}^{N} y_{c,i,j'}^n} - \frac{\sum_{n=1}^{N} z_{c,i,j}^n \cdot w_c^n}{\sum_{j'=1}^{S}\sum_{n=1}^{N} z_{c,i,j'}^n \cdot w_{c'}^n}$
    end
    for $m \leftarrow 1$ **to** M **do**
      $\gamma_{c,i,m,t}^n \leftarrow \frac{\alpha_{c,i,m} \cdot \mathcal{N}(\mathbf{x}_t^n|\boldsymbol{\mu}_{c,i,m}, \boldsymbol{\Sigma}_{c,i,m})}{b_{c,i}(\mathbf{x}_t^n)} \quad \forall n \in \{1, \dots, C\}$
      $\partial\alpha_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left(u_{c,i,t}^n \cdot \gamma_{c,i,m,t}^n\right)}{\sum_{n=1}^{N}\sum_{t=1}^{T^n} u_{c,i,t}^n} - \frac{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left(v_{c,i,t}^n \cdot \gamma_{c,i,m,t}^n \cdot w_c^n\right)}{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left(v_{c,i,t}^n \cdot w_c^n\right)}$
    end
  end
end

Algorithm 2:E-step of the CLL-HMM EBW algorithm.

Algorithm (3) shows the procedure of setting the constant D:

Determine D:

$D \leftarrow 1 + \left| \min_{c,i,j,m} \left[\partial\rho_c, \partial\pi_{c,i}, \partial a_{c,i,j}, \partial\alpha_{c,i,m} \right] \right|$

for $c \leftarrow 1$ **to** C **do**

  for $i \leftarrow 1$ **to** S **do**

    for $m \leftarrow 1$ **to** M **do**

      repeat

$$\bar{\boldsymbol{\mu}}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - v^n_{c,i,t}w^n_c\right)\mathbf{x}^n_t\right] + \boldsymbol{\mu}_{c,i,m}\cdot D}{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - v^n_{c,i,t}\cdot w^n_c\right)\right] + D}$$

$$\bar{\boldsymbol{\Sigma}}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - v^n_{c,i,t}\cdot w^n_c\right)(\mathbf{x}^n_t)^2\right] + \left(\boldsymbol{\Sigma}_{c,i,m} + (\boldsymbol{\mu}_{c,i,m})^2\right)\cdot D}{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - v^n_{c,i,t}\cdot w^n_c\right)\right] + D} - (\bar{\boldsymbol{\mu}}_{c,i,m})^2$$

$D \leftarrow 2D$

      until *all variances in* $\bar{\boldsymbol{\Sigma}}_{c,i,m}$ *positive;*

    end

  end

end

$D \leftarrow DF$

Algorithm 3 :Procedure to determine the constant D of the CLL-HMM EBW algorithm.

In Algorithm (4), the M-step of the CL-HMM EBW algorithm using parameter updates of Eq. (3.16) is illustrated.

M-Step:

for $c \leftarrow 1$ **to** C **do**

$$\bar{\rho}_c \leftarrow \frac{\rho_c\left(\partial\rho_c + D\right)}{\sum_{c'=1}^{C}\rho_{c'}\left(\partial\rho_{c'} + D\right)}$$

  for $i \leftarrow 1$ **to** S **do**

$$\bar{\pi}_{c,i} \leftarrow \frac{\pi_{c,i}\left(\partial\pi_{c,i} + D\right)}{\sum_{i'=1}^{S}\pi_{c',i}\left(\partial\pi_{c',i} + D\right)}$$

    for $j \leftarrow 1$ **to** S **do**

$$\bar{a}_{c,i,j} \leftarrow \frac{a_{c,i,j}\left(\partial a_{c,i,j} + D\right)}{\sum_{j'=1}^{S}a_{c,i,j'}\left(\partial a_{c,i,j'} + D\right)}$$

    end

    for $m \leftarrow 1$ **to** M **do**

$$\bar{\alpha}_{c,i,m} \leftarrow \frac{\alpha_{c,i,m}\left(\partial\alpha_{c,i,m} + D\right)}{\sum_{m'=1}^{M}\alpha_{c,i,m'}\left(\partial\alpha_{c,i,m'} + D\right)}$$

$$\bar{\boldsymbol{\mu}}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - v^n_{c,i,t}w^n_c\right)\mathbf{x}^n_t\right] + \boldsymbol{\mu}_{c,i,m}\cdot D}{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - v^n_{c,i,t}\cdot w^n_c\right)\right] + D}$$

$$\boldsymbol{\Sigma}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - v^n_{c,i,t}\cdot w^n_c\right)(\mathbf{x}^n_t)^2\right] + \left(\boldsymbol{\Sigma}_{c,i,m} + (\boldsymbol{\mu}_{c,i,m})^2\right)\cdot D}{\sum_{n=1}^{N}\sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - v^n_{c,i,t}\cdot w^n_c\right)\right] + D} - (\bar{\boldsymbol{\mu}}_{c,i,m})^2$$

$\boldsymbol{\mu}_{c,i,m} \leftarrow \bar{\boldsymbol{\mu}}_{c,i,m}$

    end

    $\alpha_{c,i,m} \leftarrow \bar{\alpha}_{c,i,m}\forall m$

    $a_{c,i,j} \leftarrow \bar{a}_{c,i,j}\forall j$

  end

  $\pi_{c,i} \leftarrow \bar{\pi}_{c,i}\forall i$

end

$\rho_c \leftarrow \bar{\rho}_c\forall c$

Algorithm 4 :M-step of the CLL-HMM EBW algorithm.

B

MM-HMM EBW algorithm

The implementation of the EBW algorithm for maximizing the margin (MM-HMM EBW algorithm) is stated in Algorithm (5).

Input: $\{\mathcal{X}_1, \dots \mathcal{X}_C\}$
Output: $\rho_c, \pi_{c,i}, a_{c,i,j}, \{\alpha_{c,i,m}, \boldsymbol{\mu}_{c,i,m}, \boldsymbol{\Sigma}_{c,i,m}\}_{m=1}^{M} \quad \forall c \in \{1, \dots, C\},\ \forall i, j \in \{1, \dots, S\}$
Initialization: For each c, train $\pi_{c,i}, a_{c,i,j}, \{\alpha_{c,i,m}, \boldsymbol{\mu}_{c,i,m}, \boldsymbol{\Sigma}_{c,i,m}\}_{m=1}^{M}$ on $\mathcal{X}_c$, using the EM-algorithm. Set ρ_c to class frequency in $\mathcal{X}$, i.e. $\rho_c \leftarrow \frac{|\mathcal{X}_c|}{|\mathcal{X}|}$
while $D(\mathcal{X}|\boldsymbol{\Theta})$ *not converged* **do**
  Determine: $\mathcal{X}^1$, $\mathcal{X}^2$, $\mathcal{X}^3$ based on $(d_{\Theta}^{n})^{\lambda}$
  Determine: $s^n \quad \forall n \in \{1, \dots, N\}$ based on $\mathcal{X}^1$, $\mathcal{X}^2$, $\mathcal{X}^3$
  E-Step (see Algorithm (6))
  Determine D (see Algorithm (7))
  M-Step (see Algorithm (8))
end

Algorithm 5 : Discriminative Margin-based training of HMMs (MM-HMM EBW algorithm).

The E-step of the MM-HMM EBW algorithm using the sum-to-one approximation of $\frac{\partial \log d^n_{\boldsymbol{\Theta}}}{\partial \varphi}$ (see Eq. (3.54)) is depicted in Algorithm (6).

E-Step:
for $c \leftarrow 1$ **to** C **do**

$$r_c^{n,\eta} \leftarrow \frac{(p(\mathbf{x}^n|\boldsymbol{\Theta}_c)\rho_c)^\eta}{\sum_{c' \neq c^n}^{} (p(\mathbf{x}^n|\boldsymbol{\Theta}_{c'})\rho_{c'})^\eta}$$

$$\frac{\partial log d^n_{\boldsymbol{\Theta}}}{\partial \rho_c} \leftarrow \begin{cases} \frac{\frac{1}{\rho_c}\left[q_c^n - \breve{q}_c^n \cdot r_c^{n,\eta}\right]}{\sum_{c'=1}^{C} \frac{1}{\rho_{c'}}\left[q_c^n - \breve{q}_c^n \cdot r_c^{n,\eta}\right]}, & \text{if } \sum_{c'=1}^{C}\left|\frac{\partial log d^n_{\boldsymbol{\Theta}}}{\partial \rho_{c'}}\right| > 1 \\ \frac{1}{\rho_c}\left[q_c^n - \breve{q}_c^n \cdot r_c^{n,\eta}\right], & \text{else} \end{cases}$$

$$\partial \rho_c \leftarrow \sum_{n=1}^{N} s^n \frac{\partial log d^n_{\boldsymbol{\Theta}}}{\partial \rho_c}$$

for $i \leftarrow 1$ **to** S **do**

$$\frac{\partial log d^n_{\boldsymbol{\Theta}}}{\partial \pi_{c,i}} \leftarrow \begin{cases} \frac{\frac{1}{\pi_{c,i}}\left[u^n_{c,i,1} - \breve{u}^n_{c,i,1} \cdot r_c^{n,\eta}\right]}{\sum_{i'=1}^{S} \frac{1}{\pi_{c,i'}}\left[u^n_{c,i',1} - \breve{u}^n_{c,i',1} \cdot r_c^{n,\eta}\right]}, & \text{if } \sum_{i'=1}^{S}\left|\frac{\partial log d^n_{\boldsymbol{\Theta}}}{\partial \pi_{c,i'}}\right| > 1 \\ \frac{1}{\pi_{c,i}}\left[u^n_{c,i,1} - \breve{u}^n_{c,i,1} \cdot r_c^{n,\eta}\right], & \text{else} \end{cases}$$

$$\partial \pi_{c,i} \leftarrow \sum_{n=1}^{N} s^n \frac{\partial log d^n_{\boldsymbol{\Theta}}}{\partial \pi_{c,i}}$$

for $j \leftarrow 1$ **to** S **do**

$$\frac{\partial log d^n_{\boldsymbol{\Theta}}}{\partial a_{c,i,j}} \leftarrow \begin{cases} \frac{\frac{1}{a_{c,i,j}}\left[y^n_{c,i,j} - \breve{y}^n_{c,i,j} \cdot r_c^{n,\eta}\right]}{\sum_{j'=1}^{S} \frac{1}{a_{c,i,j'}}\left[y^n_{c,i,j'} - \breve{y}^n_{c,i,j'} \cdot r_c^{n,\eta}\right]}, & \text{if } \sum_{j'=1}^{S}\left|\frac{\partial log d^n_{\boldsymbol{\Theta}}}{\partial a_{c,i,j'}}\right| > 1 \\ \frac{1}{a_{c,i,j}}\left[y^n_{c,i,j} - \breve{y}^n_{c,i,j} \cdot r_c^{n,\eta}\right], & \text{else} \end{cases}$$

$$\partial a_{c,i,j} \leftarrow \sum_{n=1}^{N} s^n \frac{\partial log d^n_{\boldsymbol{\Theta}}}{\partial a_{c,i,j}}$$

end

for $m \leftarrow 1$ **to** M **do**

$$\gamma^n_{c,i,m,t} \leftarrow \frac{\alpha_{c,i,m} \cdot \mathcal{N}(\mathbf{x}_t^n|\boldsymbol{\mu}_{c,i,m}, \boldsymbol{\Sigma}_{c,i,m})}{b_{c,i}(\mathbf{x}_t^n)} \quad \forall n \in \{1, \ldots, C\}$$

$$\frac{\partial log d^n_{\boldsymbol{\Theta}}}{\partial \alpha_{c,i,m}} \leftarrow \begin{cases} \frac{\frac{1}{\alpha_{c,i,m}} \sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - \breve{u}^n_{c,i,t} r_c^{n,\eta}\right)\right]}{\sum_{m'=1}^{M} \frac{1}{\alpha_{c,i,m'}} \sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m',t}\left(u^n_{c,i,t} - \breve{u}^n_{c,i,t} r_c^{n,\eta}\right)\right]}, & \text{if } \sum_{m'=1}^{M}\left|\frac{\partial log d^n_{\boldsymbol{\Theta}}}{\partial \alpha_{c,i,m'}}\right| > 1 \\ \frac{1}{\alpha_{c,i,m}} \sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - \breve{u}^n_{c,i,t} r_c^{n,\eta}\right)\right], & \text{else} \end{cases}$$

$$\partial \alpha_{c,i,m} \leftarrow \sum_{n=1}^{N} s^n \frac{\partial log d^n_{\boldsymbol{\Theta}}}{\partial \alpha_{c,i,m}}$$

end

end

end

Algorithm 6: E-step of the MM-HMM EBW algorithm.

Algorithm (7) shows the procedure of setting the constant D:

Determine D:

$$D \leftarrow 1 + \left|\min_{c,i,j,m}\left[\partial \rho_c, \partial \pi_{c,i}, \partial a_{c,i,j}, \partial \alpha_{c,i,m}\right]\right|$$

for $c \leftarrow 1$ **to** C **do**

for $i \leftarrow 1$ **to** S **do**

for $m \leftarrow 1$ **to** M **do**

repeat

$$\bar{\boldsymbol{\mu}}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N} s^n \sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - \breve{u}^n_{c,i,t} \cdot r_c^{n,\eta}\right)\mathbf{x}_t^n\right] + \boldsymbol{\mu}_{c,i,m} \cdot D}{\sum_{n=1}^{N} s^n \sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - \breve{u}^n_{c,i,t} \cdot r_c^{n,\eta}\right)\right] + D}$$

$$\bar{\boldsymbol{\Sigma}}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N} s^n \sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - \breve{u}^n_{c,i,t} \cdot r_c^{n,\eta}\right)(\mathbf{x}_t^n)^2\right] + \left(\boldsymbol{\Sigma}_{c,i,m} + (\boldsymbol{\mu}_{c,i,m})^2\right) \cdot D}{\sum_{n=1}^{N} s^n \sum_{t=1}^{T^n}\left[\gamma^n_{c,i,m,t}\left(u^n_{c,i,t} - \breve{u}^n_{c,i,t} \cdot r_c^{n,\eta}\right)\right] + D} - (\bar{\boldsymbol{\mu}}_{c,i,m})^2$$

$D \leftarrow 2D$

until *all variances in* $\bar{\boldsymbol{\Sigma}}_{c,i,m}$ *positive;*

end

end

end

$D \leftarrow DF$

Algorithm 7: Procedure to determine the constant D of the MM-HMM EBW algorithm.

In Algorithm (8), the M-step of the MM-HMM EBW algorithm using parameter updates of Eq. (3.16) is illustrated.

M-Step:
for $c \leftarrow 1$ **to** C **do**
  $\bar{\rho}_c \leftarrow \frac{\rho_c(\partial\rho_c + D)}{\sum_{c'=1}^{C} \rho_{c'}(\partial\rho_{c'} + D)}$
  for $i \leftarrow 1$ **to** S **do**
    $\bar{\pi}_{c,i} \leftarrow \frac{\pi_{c,i}(\partial\pi_{c,i} + D)}{\sum_{i'=1}^{S} \pi_{c',i}(\partial\pi_{c',i} + D)}$
    for $j \leftarrow 1$ **to** S **do**
      $\bar{a}_{c,i,j} \leftarrow \frac{a_{c,i,j}(\partial a_{c,i,j} + D)}{\sum_{j'=1}^{S} a_{c,i,j'}(\partial a_{c,i,j'} + D)}$
    end
    for $m \leftarrow 1$ **to** M **do**
      $\bar{\alpha}_{c,i,m} \leftarrow \frac{\alpha_{c,i,m}(\partial\alpha_{c,i,m} + D)}{\sum_{m'=1}^{M} \alpha_{c,i,m'}(\partial\alpha_{c,i,m'} + D)}$
      $\bar{\boldsymbol{\mu}}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N} s^n \sum_{t=1}^{T^n} \left[\gamma_{c,i,m,t}^{n}\left(u_{c,i,t}^{n} - \breve{u}_{c,i,t}^{n} \cdot r_c^{n,\eta}\right)\mathbf{x}_t^n\right] + \boldsymbol{\mu}_{c,i,m} \cdot D}{\sum_{n=1}^{N} s^n \sum_{t=1}^{T^n} \left[\gamma_{c,i,m,t}^{n}\left(u_{c,i,t}^{n} - \breve{u}_{c,i,t}^{n} \cdot r_c^{n,\eta}\right)\right] + D}$
      $\boldsymbol{\Sigma}_{c,i,m} \leftarrow \frac{\sum_{n=1}^{N} s^n \sum_{t=1}^{T^n} \left[\gamma_{c,i,m,t}^{n}\left(u_{c,i,t}^{n} - \breve{u}_{c,i,t}^{n} \cdot r_c^{n,\eta}\right)(\mathbf{x}_t^n)^2\right] + (\boldsymbol{\Sigma}_{c,i,m} + (\boldsymbol{\mu}_{c,i,m})^2) \cdot D}{\sum_{n=1}^{N} s^n \sum_{t=1}^{T^n} \left[\gamma_{c,i,m,t}^{n}\left(u_{c,i,t}^{n} - \breve{u}_{c,i,t}^{n} \cdot r_c^{n,\eta}\right)\right] + D} - (\bar{\boldsymbol{\mu}}_{c,i,m})^2$
      $\boldsymbol{\mu}_{c,i,m} \leftarrow \bar{\boldsymbol{\mu}}_{c,i,m}$
    end
    $\alpha_{c,i,m} \leftarrow \bar{\alpha}_{c,i,m} \forall m$
    $a_{c,i,j} \leftarrow \bar{a}_{c,i,j} \forall j$
  end
  $\pi_{c,i} \leftarrow \bar{\pi}_{c,i} \forall i$
end
$\rho_c \leftarrow \bar{\rho}_c \forall c$

Algorithm 8 :M-step of the CL-HMM EBW algorithm.

Bibliography

[1] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[2] L. Cao and F. Tay, "Support vector machine with adaptive parameters in financial time series forecasting," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1506–1518, 2003.

[3] S. D. Campbell and F. X. Diebold, "Weather forecasting for weather derivatives," *Journal of the American Statistical Association*, vol. 100, no. 469, pp. 6–16, 2005.

[4] C. M. Bishop, *Pattern Recognition and Machine Learning.* Springer, 2006.

[5] B. Pfister and T. Kaufmann, *Sprachverarbeitung: Grundlagen und Methoden der Sprachsynthese und Spracherkennung.* Springer, 2008.

[6] L. Rabiner and J. Biing-Hwang, *Fundamentals of Speech Recognition.* Prentice hall, 1993.

[7] J. A. Bilmes *et al.*, "A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models," *International Computer Science Institute*, vol. 4, no. 510, p. 126, 1998.

[8] L. Bahl, P. Brown, P. de Souza, and R. Mercer, "Maximum mutual information estimation of hidden Markov model parameters for speech recognition," pp. 49–52, 1986.

[9] P. C. Woodland and D. Povey, "Large scale discriminative training of hidden Markov models for speech recognition," *Computer Speech and Language*, vol. 16, pp. 7–16, 2002.

[10] P. S. Gopalakrishnan, D. Kanevsky, A. Nadas, and D. Nahamoo, "An inequality for rational functions with applications to some statistical estimation problems," *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 107–113, 1991.

[11] B. Merialdo, "Phonetic recognition using hidden Markov models and maximum mutual information training," pp. 111–114, 1988.

[12] Y. Normandin and S. Morgera, "An improved MMIE training algorithm for speaker-independent, small vocabulary, continuous speech recognition," pp. 537–540 vol.1, 1991.

[13] A. Klautau, N. Jevtic, and A. Orlitsky, "Discriminative Gaussian mixture models: A comparison with kernel classifiers," *International Conference on Machine Learning*, pp. 353–360, 2003.

[14] F. Pernkopf and M. Wohlmayr, "Large margin learning of Bayesian classifiers based on Gaussian mixture models," pp. 50–66, 2010.

[15] Y. Guo, D. Wilkinson, and D. Schuurmans, "Maximum margin bayesian networks," *International Conference on Uncertainity in Artificial Intelligence*, 2012.

[16] B. Schölkopf and A. Smola, *Learning with kernels: Support Vector Machines, regularization, optimization, and beyond*. MIT Press, 2001.

[17] T. M. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

[18] "The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus," [Online; accessed 25-February-2013]. Available: http://www.ldc.upenn.edu/Catalog/readme_files/timit.readme.html

[19] C. Lopes and F. Perdigão, "Phone Recognition on the TIMIT Database," *Speech Technologies/Book*, vol. 1, pp. 285–302, 2011.

[20] A. Halberstadt, "Heterogeneous acoustic measurements and multiple classifiers for speech recognition," Ph.D. dissertation, 1998.

[21] B. Williams, "Extracting Motion Primitives from Natural Handwriting Data," Ph.D. dissertation, 2008.

[22] B. H. Williams, "UCI Machine Learning Repository: Character Trajectories Data Set," 2008, [Online; accessed 21-February-2013]. Available: http://archive.ics.uci.edu/ml/datasets/Character+Trajectories

[23] F. Prat, M. Castro, D. Llorens, A. Marzal, and J. Vilar, "UCI Machine Learning Repository: UJI Pen Characters (Version 2) Data Set," 2009, [Online; accessed 21-February-2013]. Available: http://archive.ics.uci.edu/ml/datasets/UJI+Pen+Characters+(Version+2)