Large Margin Learning of Bayesian Classifiers Based on Gaussian Mixture Models*

Franz Pernkopf and Michael Wohlmayr

Graz University of Technology, Inffeldgasse 16c, A-8010 Graz, Austria
{pernkopf,michael.wohlmayr}@tugraz.at

Abstract. We present a discriminative learning framework for Gaussian mixture models (GMMs) used for classification based on the extended Baum-Welch (EBW) algorithm [1]. We suggest two criteria for discriminative optimization, namely the class conditional likelihood (CL) and the maximization of the margin (MM). In the experiments, we present results for synthetic data, broad phonetic classification, and a remote sensing application. The experiments show that CL-optimized GMMs (CL-GMMs) achieve a lower performance compared to MM-optimized GMMs (MM-GMMs), whereas both discriminative GMMs (DGMMs) perform significantly better than generatively learned GMMs. We also show that the generative discriminatively parameterized GMM classifiers still allow to marginalize over missing features, a case where generative classifiers have an advantage over purely discriminative classifiers such as support vector machines or neural networks.

1 Introduction

In statistical learning theory [2], the PAC bound on the expected risk for unseen data depends on the empirical risk on training data and a measure for the generalization ability of the empirical model which is directly related to the Vapnik-Chervonenkis (VC) dimension. One of the most successful discriminative classifiers, namely the support vector machine (SVM) [3], finds a decision boundary which maximizes the margin between samples of distinct classes resulting in good generalization properties of the classifier. In contrast, conventional discriminative training methods relying on the conditional likelihood (CL) optimize only the empirical risk which is suboptimal. Taskar et al. [4] observed that undirected graphical models can be efficiently trained to maximize the margin. More recently, Guo et al. [5] introduced the maximization of the margin to Bayesian networks. Unlike in undirected graphical models, the main difficulty for Bayesian networks is the normalization constraint of the local conditional probabilities. In [5], this constraint is relaxed to obtain a convex optimization

* This work was supported by the Austrian Science Fund (Project number P22488-N23) and (Project number S10604-N13).

J.L. Balcázar et al. (Eds.): ECML PKDD 2010, Part III, LNAI 6323, pp. 50–66, 2010.

problem, whereby conditions on the graph structure are given where the relaxed problem matches the normalized network [6]. In [7], margin optimization has been applied to GMMs, but similar as above, the normalization constraint has been neglected leading to a *convex* optimization problem. Since then, different margin-based training algorithms have been proposed for HMMs in [8,9] and references therein.

Compared to [5,8], we aim to follow a quite different approach in this paper to maximize the margin in GMM-based classifiers. We keep the sum-to-one constraint which maintains the probabilistic interpretation of the network, e.g. marginalization over missing variables is still possible (as we show in this paper). However, we no longer have a convex optimization problem in general. Convex optimization requires convex loss function, whereas we can also use differentiable non-convex loss functions. Collobert et al. [10] show that the optimization of non-convex loss functions in SVMs can lead to sparse solutions (lower number of support vectors) and accelerated training performance. They conclude that the sacrosanct popularity of convex approaches should not anticipate the exploration of alternative techniques, since they may offer computational advantages. Similar observations are reported in [9].

In this paper, we derive a discriminative training method for GMM-based Bayesian classifiers. The algorithm is based on the EBW parameter re-estimation method [1]. In [11] it is shown that the EBW algorithm resembles the gradient descent algorithm for discriminative GMM optimization using a particular choice of step size in the gradient descent method. Nevertheless, EBW offers an *EM-like* parameter update, whereas the gradient descent method requires additional prudence, e.g. line search or learning rate. We suggest to either optimize the conditional likelihood (CL) or to maximize the margin (MM).[1] The CL criterion is related to the maximum mutual information (MMI) criterion which is popular in speech processing [12,13]. In [14], EBW has been applied to optimize Gaussian mixture models with respect to CL. However, they neglect to optimize the class prior. In the experiments, we depict the differences of the decision boundary for generatively and discriminatively learned GMMs for classification using synthetic data. Furthermore, we show results for broad phonetic classification [15] and compare discriminative GMM classifiers to SVMs and neural networks (NNs). Moreover, one of the key advantages of generative models over discriminative ones (such as SVMs or NNs) is that it is still possible to marginalize over missing features. We provide empirical results showing that the performance advantage of discriminatively learned GMMs for classification can be maintained for a low number of missing values. This is also shown for a remote sensing application on hyperspectral data.

The paper is organized as follows: In Section 2, we shortly review the Bayesian classifier and generative learning of GMMs, respectively. Additionally, notation is introduced. In Section 3, we derive a discriminative learning method for CL-GMMs based on the EBW algorithm used for classification. Margin-based

[1] Both algorithms are implemented in Matlab and can be downloaded at: `http://www.spsc.tugraz.at/people/franz_pernkopf/`

GMM learning is presented in Section 4. We report experimental results on synthetic and real-world data in Section 5. Finally, Section 6 concludes the paper.

2 Bayesian Classifier

The Bayesian classifier [16] relies on the Bayes rule to determine the class posterior probability according to

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

where $c \in \{1,\ldots,C\}$, and C is the number of classes. The posterior probability $p\left(c|\mathbf{x}^n\right)$ models the probability of c given the feature vector of the n^{th} sample $\mathbf{x}^n$. We predict the class label using the MAP (maximum posterior) estimate, i.e. the most likely class label c^* is determined using the class posteriors as

$$c^* = \arg \max_{c\in\{1,\ldots,C\}} p\left(c|\mathbf{x}^n\right) = \arg \max_{c\in\{1,\ldots,C\}} p\left(\mathbf{x}^n|c\right)p\left(c\right), \tag{2}$$

where the denominator of Eq. (1) can be neglected since it only scales $p\left(c|\mathbf{x}^n\right)$ and does not alter the decision in Eq. (2). This equation is a solution of the Bayesian risk minimization problem with the 0/1-loss function. The term $p\left(c\right)$ is known as class prior distribution. We use GMMs to model the term $p\left(\mathbf{x}^n|c\right)$, i.e. for each class c we have a GMM $p\left(\mathbf{x}^n|\mathbf{\Theta}_c\right)$. A Gaussian mixture model $p\left(\mathbf{x}^n|\mathbf{\Theta}_c\right)$ is the weighted sum of $M > 1$ Gaussian components $\mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}_c^m\right)$ in $\mathbb{R}^d$, $p\left(\mathbf{x}^n|\mathbf{\Theta}_c\right) = \sum\limits_{m=1}^{M} \alpha_c^m \mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}_c^m\right)$, where α_c^m corresponds to the weight of each component $m \in \{1,\ldots,M\}$. These weights are constrained to be positive $\alpha_c^m \geq 0$ and $\sum_{m=1}^{M} \alpha_c^m = 1$. The GMM is specified by the set of parameters $\mathbf{\Theta}_c = \left\{\alpha_c^1, \alpha_c^2, \ldots, \alpha_c^M, \boldsymbol{\theta}_c^1, \boldsymbol{\theta}_c^2, \ldots, \boldsymbol{\theta}_c^M\right\}$, where the Gaussians are specified by the mean vector $\boldsymbol{\mu}_c^m$ and the covariance matrix $\boldsymbol{\Sigma}_c^m$, i.e. $\boldsymbol{\theta}_c^m = \{\boldsymbol{\mu}_c^m, \boldsymbol{\Sigma}_c^m\}$. The EM algorithm [16,17] commonly used for learning GMMs consists of an *expectation* step (E-step) and a *maximization* step (M-step) which are alternately used until the $\log p\left(\mathcal{X}_c|\mathbf{\Theta}_c\right) = \log \prod_{n=1}^{N_c} p\left(\mathbf{x}^n|\mathbf{\Theta}_c\right)$ converges to a local optimum, where $\mathcal{X}_c = \left\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^{N_c}\right\}_c$ are N_c i.i.d. samples belonging to class c. $\mathcal{X}$ contains samples of all classes $\mathcal{X} = \{\mathcal{X}_1, \ldots, \mathcal{X}_C\}$ where N denotes the size of $\mathcal{X}$, i.e. $N = |\mathcal{X}| = \sum_{c=1}^{C} N_c$. The performance of the EM algorithm depends strongly on the choice of the initial parameters.

3 Discriminative CL-Based Learning of GMMs in Bayesian Classifiers

Optimizing CL is tightly connected to good classification performance. Hence, we want to learn parameters of the GMM-based Bayesian classifier so that CL is maximized. Unfortunately, CL does not decompose. The objective function of the conditional log likelihood (CLL) using GMMs in Eq. (1) is

$$CLL\left(\mathcal{X}|\mathbf{\Theta}\right) = \log \prod_{n=1}^{N} p\left(c^n|\mathbf{x}^n\right) = \sum_{n=1}^{N} \log \frac{p\left(\mathbf{x}^n|\mathbf{\Theta}_{c^n}\right)\rho_{c^n}}{\sum\limits_{c'=1}^{C} p\left(\mathbf{x}^n|\mathbf{\Theta}_{c'}\right)\rho_{c'}} =$$
$$\sum_{n=1}^{N}\left[\log\left[\left(\sum_{m=1}^{M}\alpha_{c^n}^{m}\mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}_{c^n}^{m}\right)\right)\rho_{c^n}\right] - \log\sum_{c'=1}^{C}\left[\left(\sum_{m=1}^{M}\alpha_{c'}^{m}\mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}_{c'}^{m}\right)\right)\rho_{c'}\right]\right], \tag{3}$$

where, c^n is the class of $\mathbf{x}^n$, $\rho_{c^n} = p\left(c^n\right)$ is the class prior of the n^{th} sample, $0 < \rho_{c^n} < 1$, and $\sum_{c=1}^{C}\rho_{c^n} = 1$. The set of parameters $\mathbf{\Theta}$ is composed of $\mathbf{\Theta} = \{\mathbf{\Theta}_1, \ldots, \mathbf{\Theta}_C, \rho_1, \ldots, \rho_C\}$.

The EBW algorithm (more details are given in Appendix A) is an iterative procedure which can be used to optimize rational functions [1]. Clearly, the CL criterion in Eq. (3) is a rational function over the discrete model parameters ρ_c and α_c^m and the parameter re-estimation equation of the form

$$\theta_i^j \leftarrow \frac{\theta_i^j\left(\frac{\partial CLL(\mathcal{X}|\mathbf{\Theta})}{\partial\theta_i^j} + D\right)}{\sum\limits_{l}\theta_l^j\left(\frac{\partial CLL(\mathcal{X}|\mathbf{\Theta})}{\partial\theta_l^j} + D\right)}, \tag{4}$$

is used, where $\theta_i^j \geq 0$, $\sum_i \theta_i^j = 1$, and j indicates a particular discrete variable. EBW requires the partial derivative $\frac{\partial CLL(\mathcal{X}|\mathbf{\Theta})}{\partial\theta_i^j}$ and D. Both terms are provided in the sequel. Specifically,

$$\frac{\partial CLL\left(\mathcal{X}|\mathbf{\Theta}\right)}{\partial\rho_c} = \sum_{n=1}^{N}\left[\frac{\mathbb{1}_{\{c=c^n\}}}{\rho_c} - \frac{p\left(\mathbf{x}^n|\mathbf{\Theta}_c\right)\rho_c}{\sum_{c'=1}^{C} p\left(\mathbf{x}^n|\mathbf{\Theta}_{c'}\right)\rho_{c'}}\frac{1}{\rho_c}\right] = \frac{1}{\rho_c}\sum_{n=1}^{N}\left(\mathbb{1}_{\{c=c^n\}} - w_c^n\right), \tag{5}$$

where $w_c^n = p\left(c|\mathbf{x}^n\right)$ (same as Eq. (1)) and $\mathbb{1}_{\{i=j\}}$ is the indicator function (i.e. equals 1 if $i = j$ and 0 if $i \neq j$).

Further, the derivative for the parameters α_c^m is

$$\frac{\partial CLL\left(\mathcal{X}|\mathbf{\Theta}\right)}{\partial\alpha_c^m} = \sum_{n=1}^{N}\left[\frac{\gamma_c^{n,m}}{\alpha_c^m}\left(\mathbb{1}_{\{c=c^n\}} - w_c^n\right)\right], \tag{6}$$

where

$$\gamma_c^{n,m} = \frac{\alpha_c^m\mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}_c^m\right)}{\sum_{m'=1}^{M}\alpha_c^{m'}\mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}_c^{m'}\right)}. \tag{7}$$

Considering the derivative in Eq. (6) (similar for Eq. (5)) in the re-estimation Eq. (4) we obtain

$$\alpha_c^m \leftarrow \frac{\sum_{n=1}^{N}\left[\gamma_c^{n,m}\left(\mathbb{1}_{\{c=c^n\}} - w_c^n\right)\right] + \alpha_c^m D}{\sum_{m'=1}^{M}\sum_{n=1}^{N}\left[\gamma_c^{n,m'}\left(\mathbb{1}_{\{c=c^n\}} - w_c^n\right)\right] + D}.$$

The derivatives (Eq. (5) and (6)) are sensitive to small parameter values. Merialdo [18] observed that low-valued parameters ρ_c and α_c^m in Eq. (5) and (6) may cause a large magnitude of the gradient. Hence, the optimization concentrates on those parameters which are usually unreliably estimates due to lack of data. Therefore, he suggests to focus on modifying better estimated high-valued parameters by using an approximation for the derivative in Eq. (6) (similar for Eq. (5))

$$\frac{\partial CLL\left(\mathcal{X}|\boldsymbol{\Theta}\right)}{\partial \alpha_c^m} \approx \frac{\sum_{n=1}^{N} \gamma_c^{n,m} \mathbb{1}_{\{c=c^n\}}}{\sum_{m'=1}^{M} \sum_{n=1}^{N} \gamma_c^{n,m'} \mathbb{1}_{\{c=c^n\}}} - \frac{\sum_{n=1}^{N} \gamma_c^{n,m} w_c^n}{\sum_{m'=1}^{M} \sum_{n=1}^{N} \gamma_c^{n,m'} w_c^n}. \tag{8}$$

EBW has been formulated for discrete probability distributions. Normandin and Morgera [19] introduced a discrete approximation of the Gaussian distribution assuming diagonal covariance matrices. This leads to the re-estimation equation for $\bar{\boldsymbol{\mu}}_c^m$ and $\bar{\boldsymbol{\Sigma}}_c^m$ given as

$$\bar{\boldsymbol{\mu}}_c^m \leftarrow \frac{\sum_{n=1}^{N} \left[\gamma_c^{n,m} \left(\mathbb{1}_{\{c=c^n\}} - w_c^n\right) \mathbf{x}^n\right] + D\boldsymbol{\mu}_c^m}{\sum_{n=1}^{N} \left[\gamma_c^{n,m} \left(\mathbb{1}_{\{c=c^n\}} - w_c^n\right)\right] + D}$$

and

$$\bar{\boldsymbol{\Sigma}}_c^m \leftarrow \frac{\sum_{n=1}^{N} \left[\gamma_c^{n,m} \left(\mathbb{1}_{\{c=c^n\}} - w_c^n\right) \left(\mathbf{x}^n\right)^2\right] + D\left(\boldsymbol{\Sigma}_c^m + \left(\boldsymbol{\mu}_c^m\right)^2\right)}{\sum_{n=1}^{N} \left[\gamma_c^{n,m} \left(\mathbb{1}_{\{c=c^n\}} - w_c^n\right)\right] + D} - \left(\bar{\boldsymbol{\mu}}_c^m\right)^2, \tag{9}$$

where the squares of the vectors $\mathbf{x}^n$ and $\boldsymbol{\mu}_c^m$ are element-wise.

The EBW algorithm converges to a local optimum of $CLL\left(\mathcal{X}|\boldsymbol{\Theta}\right)$ providing a sufficiently large value for D. Setting the constant D is not trivial. If it is chosen too large then training is slow and if it is too small the update may fail to increase the objective function. In practical implementations heuristics have been suggested [13,14]. We initialize $D = 1$ and double D until all variances in Eq. (9) are positive in the re-estimation step. Next, we multiply the obtained D with a global factor F (In Section 5.1, we empirically show the dependency of F on the convergence of EBW.). Value D is adapted in each iteration of the algorithm. The parameters $\boldsymbol{\Theta}_c$ for discriminative learning are initialized to the ML estimates of the GMM determined by the EM algorithm (see Section 2). The class prior is set to the normalized class frequency in $\mathcal{X}$, i.e. $\rho_c = \frac{N_c}{N}$.

4 Discriminative Margin-Based Learning of GMMs in Bayesian Classifiers

The multi-class margin [5] of sample n is

$$d_{\boldsymbol{\Theta}}^n = \min_{c \neq c^n} \frac{p\left(c^n|\mathbf{x}^n, \boldsymbol{\Theta}\right)}{p\left(c|\mathbf{x}^n, \boldsymbol{\Theta}\right)} = \min_{c \neq c^n} \frac{p\left(c^n, \mathbf{x}^n|\boldsymbol{\Theta}\right)}{p\left(c, \mathbf{x}^n|\boldsymbol{\Theta}\right)} = \frac{p\left(c^n, \mathbf{x}^n|\boldsymbol{\Theta}\right)}{\max_{c \neq c^n} p\left(c, \mathbf{x}^n|\boldsymbol{\Theta}\right)}. \tag{10}$$

If $d_{\boldsymbol{\Theta}}^n > 1$, then sample n is correctly classified and vice versa. We replace the max operator by the differentiable approximation $\max_x f(x) \approx [\sum_x (f(x))^\eta]^{\frac{1}{\eta}}$, where $\eta \geq 1$ and $f(x)$ is non-negative. In the limit of $\eta \rightarrow \infty$ the approximation converges to the max operation. Replacing the max with its approximation, we obtain

$$d_{\boldsymbol{\Theta}}^n = \frac{p(c^n, \mathbf{x}^n|\boldsymbol{\Theta})}{\left[\sum_{c \neq c^n} (p(c, \mathbf{x}^n|\boldsymbol{\Theta}))^\eta\right]^{\frac{1}{\eta}}}.$$

Usually, the max margin approach maximizes the margin of the sample with the smallest margin, i.e. $\min_{n=1,\ldots,N} d_{\boldsymbol{\Theta}}^n$ for a separable classification problem [3]. We aim to relax this by introducing a soft margin, i.e. we focus on samples with a $d_{\boldsymbol{\Theta}}^n$ close to one. Therefore, we consider the *hinge* loss function according to

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

using the margin. Maximizing this function with respect to the parameters $\boldsymbol{\Theta}$ implicitly means to increase the margin $d_{\boldsymbol{\Theta}}^n$ whereas the emphasis is on samples with a margin $(d_{\boldsymbol{\Theta}}^n)^\lambda < 2$, i.e. samples with a large positive margin have no impact on the optimization. The parameter $\lambda > 0$ scales the margin and is set by cross-validation. Maximizing $\tilde{D}(\mathcal{X}|\boldsymbol{\Theta})$ via EBW or gradient descent is not straight forward due to the discontinuity in the derivative at $(d_{\boldsymbol{\Theta}}^n)^\lambda = 2$. Therefore, we propose to use for the *hinge* function $h(y) = \min[2, y]$ a *smooth hinge* function which enables a smooth transition of the derivative and has a similar shape as $h(y)$. We propose the following *smooth hinge* function

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

which requires to divide the data $\mathcal{X}$ into three partitions depending on $y = (d_{\boldsymbol{\Theta}}^n)^\lambda$, i.e. $\mathcal{X}^1$ contains samples where $(d_{\boldsymbol{\Theta}}^n)^\lambda \leq 1$, $\mathcal{X}^2$ consists of samples with a margin in the range $1 < (d_{\boldsymbol{\Theta}}^n)^\lambda < 2$, and $\mathcal{X}^3 = \mathcal{X} \setminus \{\mathcal{X}^1 \cup \mathcal{X}^2\}$. Hence, our objective function for margin maximization is

$$D(\mathcal{X}|\boldsymbol{\Theta}) = \prod_{n=1}^{N} h((d_{\boldsymbol{\Theta}}^n)^\lambda) = \left\{\prod_{n \in \mathcal{X}^1} \left((d_{\boldsymbol{\Theta}}^n)^\lambda + \frac{1}{2}\right)\right\}\left\{\prod_{n \in \mathcal{X}^2} \left[2 - \frac{1}{2}\left((d_{\boldsymbol{\Theta}}^n)^\lambda - 2\right)^2\right]\right\} 2^{|\mathcal{X}^3|}$$

using the smooth hinge function. The λ for scaling the margin is usually selected as fraction number leading to *fractional polynomials* in the numerator and denominator of $d_{\boldsymbol{\Theta}}^n$. The growth transform of the EBW algorithm (see [1]) extends to fractional polynomials and we can use the EBW algorithm for maximizing $D(\mathcal{X}|\boldsymbol{\Theta})$. Therefore, the derivative $\frac{\partial \log D(\mathcal{X}|\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}$ for the re-estimation equation (see Eqn. 4) of the EBW algorithm is

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

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

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

Introducing GMMs in Eq. 10 and using the log gives

$$\log d^n_{\boldsymbol{\Theta}} = \log \left[\left(\sum_{m=1}^{M} \alpha^m_{c^n} \mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}^m_{c^n}\right) \right) \rho_{c^n} \right] - \frac{1}{\eta} \log \sum_{c' \neq c^n} \left[\left(\sum_{m=1}^{M} \alpha^m_{c'} \mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}^m_{c'}\right) \right) \rho_{c'} \right]^\eta.$$

Similar as in Eq. 5 (Section 3), the partial derivative of $\log d^n_{\boldsymbol{\Theta}}$ for the parameters ρ_c is

$$\frac{\partial \log d^n_{\boldsymbol{\Theta}}}{\partial \rho_c} = \frac{\mathbb{1}_{\{c=c^n\}}}{\rho_c} - \mathbb{1}_{\{c \neq c^n\}} \frac{\left[p\left(\mathbf{x}^n|\boldsymbol{\Theta}_c\right)\rho_c\right]^\eta}{\left[\sum_{c' \neq c^n} p\left(\mathbf{x}^n|\boldsymbol{\Theta}_{c'}\right)\rho_{c'}\right]^\eta} \frac{1}{\rho_c} = \frac{1}{\rho_c}\left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r^{n,\eta}_c\right),$$

where we introduced

$$r^{n,\eta}_c = \frac{\left[p\left(\mathbf{x}^n|\boldsymbol{\Theta}_c\right)\rho_c\right]^\eta}{\left[\sum_{c' \neq c^n} p\left(\mathbf{x}^n|\boldsymbol{\Theta}_{c'}\right)\rho_{c'}\right]^\eta}.$$

Furthermore, the derivative for the parameters α^m_c is

$$\frac{\partial \log d^n_{\boldsymbol{\Theta}}}{\partial \alpha^m_c} = \frac{\mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}^m_c\right)}{\sum_{m'=1}^{M} \alpha^{m'}_c \mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}^{m'}_c\right)}\left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r^{n,\eta}_c\right) = \frac{\gamma^{n,m}_c}{\alpha^m_c}\left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r^{n,\eta}_c\right),$$

where $\gamma^{n,m}_c$ is given in Eq. 7. For the Gaussian distributions we use again the discrete approximation proposed in [19] assuming diagonal covariance matrices. This leads to the re-estimation equation for $\bar{\boldsymbol{\mu}}^m_c$ and $\bar{\boldsymbol{\Sigma}}^m_c$ given as

$$\bar{\boldsymbol{\mu}}^m_c \leftarrow \frac{\sum_{n=1}^{N}\left[s^n \gamma^{n,m}_c \left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r^{n,\eta}_c\right)\mathbf{x}^n\right] + D\boldsymbol{\mu}^m_c}{\sum_{n=1}^{N}\left[s^n \gamma^{n,m}_c \left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r^{n,\eta}_c\right)\right] + D}$$

and

$$\bar{\boldsymbol{\Sigma}}^m_c \leftarrow \frac{\sum_{n=1}^{N}\left[s^n \gamma^{n,m}_c \left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r^{n,\eta}_c\right)\left(\mathbf{x}^n\right)^2\right] + D\left(\boldsymbol{\Sigma}^m_c + \left(\boldsymbol{\mu}^m_c\right)^2\right)}{\sum_{n=1}^{N}\left[s^n \gamma^{n,m}_c \left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r^{n,\eta}_c\right)\right] + D} - \left(\bar{\boldsymbol{\mu}}^m_c\right)^2,$$

where the squares of the vectors are element-wise. Furthermore, the value D is determined in a similar manner as in Section 3. The EBW algorithm to discriminatively optimize the margin of GMM-based classifiers is summarized in Algorithm 1. Again, we use a more robust approximation for the derivatives of ρ_c and α_c^m as suggested in Section 3.

5 Experimental Results

First we show the differences in the decision boundaries of generatively and discriminatively trained GMM-based Bayesian classifiers using synthetic data. Then, we provide classification results for a remote sensing and broad phonetic classification task.

5.1 Synthetic Data

We have two classes where each class is represented by a spiral. For class 1, sample $\mathbf{x} \in \mathbb{R}^2$ is determined according to $\mathbf{x} = [t\cos(4\pi t) + \epsilon_1 \quad t\sin(4\pi t) + \epsilon_2]^T$, where ϵ_1 and ϵ_2 are independent samples from a zero-mean Gaussian noise process with $\sigma = 1$, and t is sampled from an uniform distribution. Likewise, samples for class 2 are obtained by using $\mathbf{x} = [-t\cos(4\pi t) + \epsilon_1 \quad -t\sin(4\pi t) + \epsilon_2]^T$. For each class we draw $N_c = 5000$ and $N_c = 1000$ samples for training and testing, respectively. Figure 1 shows various cases of generatively and discriminatively learned GMM-based Bayesian classifiers using $M = 12$ components per class, i.e. (a) decision boundary of generative GMM, (b) decision boundary of CL-GMM,

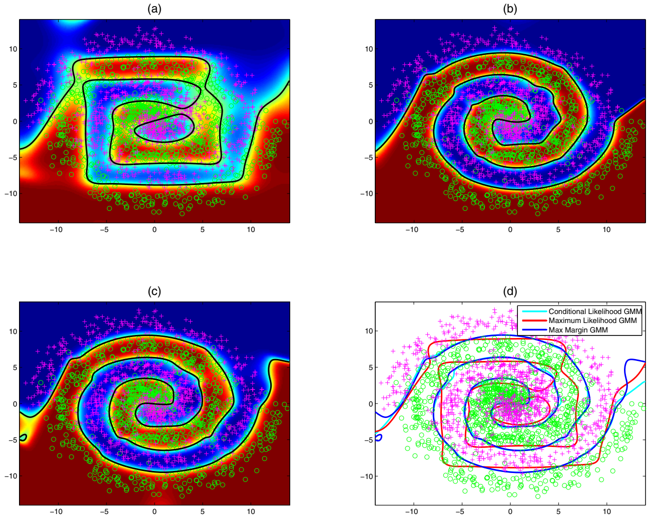
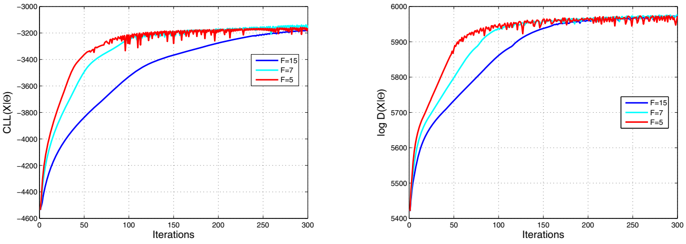


Fig. 1. Synthetic data: (a) generative GMM, (b) CL-GMM, (c) MM-GMM, and (d) decision boundary of all learning approaches

```
Input: $\mathcal{X} = \{\mathcal{X}_1, \ldots, \mathcal{X}_C\}, \eta, \lambda, F$
Output: $\rho_c, \{\alpha_c^m, \boldsymbol{\mu}_c^m, \boldsymbol{\Sigma}_c^m\}_{m=1}^M \quad \forall c \in \{1, ..., C\}$
Initialization: For each $c$, train $\{\alpha_c^m, \boldsymbol{\mu}_c^m, \boldsymbol{\Sigma}_c^m\}_{m=1}^M$ on $\mathcal{X}_c$, using the EM-algorithm. Set $\rho_c$
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
    $d_{\boldsymbol{\Theta}}^n = \frac{\left(\sum_{m=1}^M \alpha_{c^n}^m \mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}_{c^n}^m\right)\right)\rho_{c^n}}{\left[\sum_{c' \neq c^n}\left[\left(\sum_{m=1}^M \alpha_{c'}^m \mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}_{c'}^m\right)\right)\rho_{c'}\right]^\eta\right]^{\frac{1}{\eta}}} \quad \forall n \in \{1, ..., N\}$
    Determine: $\mathcal{X}^1, \mathcal{X}^2, \mathcal{X}^3$ based on $(d_{\boldsymbol{\Theta}}^n)^\lambda$
    Determine: $s^n \quad \forall n \in \{1, ..., N\}$ based on $\mathcal{X}^1, \mathcal{X}^2, \mathcal{X}^3$
    E-step:
    for $c \leftarrow 1$ to $C$ do
        $r_c^{n,\eta} \leftarrow \frac{\left[p\left(\mathbf{x}^n|\boldsymbol{\Theta}_c\right)\rho_c\right]^\eta}{\left[\sum_{c' \neq c^n} p\left(\mathbf{x}^n|\boldsymbol{\Theta}_{c'}\right)\rho_{c'}\right]^\eta} \quad \forall n \in \{1, ..., N\}$
        $\partial\rho_c \leftarrow \frac{\sum_{n=1}^N s^n \mathbb{1}_{\{c=c^n\}}}{\sum_{c'=1}^C \sum_{n=1}^N s^n \mathbb{1}_{\{c'=c^n\}}} - \frac{\sum_{n=1}^N s^n \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}}{\sum_{c'=1}^C \sum_{n=1}^N s^n \mathbb{1}_{\{c'\neq c^n\}} r_{c'}^{n,\eta}}$
        for $m \leftarrow 1$ to $M$ do
            $\gamma_c^{n,m} \leftarrow \frac{\alpha_c^m \mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}_c^m\right)}{\sum_{m'=1}^M \alpha_c^{m'} \mathcal{N}\left(\mathbf{x}^n|\boldsymbol{\theta}_c^{m'}\right)} \quad \forall n \in \{1, ..., N\}$
            $\partial\alpha_c^m \leftarrow \frac{\sum\limits_{n=1}^N s^n \gamma_c^{n,m} \mathbb{1}_{\{c=c^n\}}}{\sum\limits_{m'=1}^M \sum\limits_{n=1}^N s^n \gamma_c^{n,m'} \mathbb{1}_{\{c=c^n\}}} - \frac{\sum\limits_{n=1}^N s^n \gamma_c^{n,m} \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}}{\sum\limits_{m'=1}^M \sum\limits_{n=1}^N s^n \gamma_c^{n,m'} \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}}$
        end
    end
    Determine D: $D \leftarrow \frac{1}{2}$
    for $c \leftarrow 1$ to $C$ do
        for $m \leftarrow 1$ to $M$ do
            repeat
                $D \leftarrow 2\,D$
                $\bar{\boldsymbol{\mu}}_c^m \leftarrow \frac{\sum_{n=1}^N\left[s^n \gamma_c^{n,m}\left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}\right)\mathbf{x}^n\right] + D\boldsymbol{\mu}_c^m}{\sum_{n=1}^N\left[s^n \gamma_c^{n,m}\left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}\right)\right] + D}$
                $\boldsymbol{\Sigma}_c^m \leftarrow$
                $\frac{\sum_{n=1}^N\left[s^n \gamma_c^{n,m}\left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}\right)\left(\mathbf{x}^n\right)^2\right] + D\left(\boldsymbol{\Sigma}_c^m + \left(\boldsymbol{\mu}_c^m\right)^2\right)}{\sum_{n=1}^N\left[s^n \gamma_c^{n,m}\left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}\right)\right] + D} - (\bar{\boldsymbol{\mu}}_c^m)^2$
            until all variances in $\bar{\boldsymbol{\Sigma}}_c^m$ positive ;
        end
    end
    $D \leftarrow DF$
    M-step:
    for $c \leftarrow 1$ to $C$ do
        $\bar{\rho}_c \leftarrow \frac{\rho_c(\partial\rho_c + D)}{\sum_{c'=1}^C \rho_{c'}\left(\partial\rho_{c'} + D\right)}$
        for $m \leftarrow 1$ to $M$ do
            $\bar{\alpha}_c^m \leftarrow \frac{\alpha_c^m\left(\partial\alpha_c^m + D\right)}{\sum_{m'=1}^M \alpha_c^{m'}\left(\partial\alpha_c^{m'} + D\right)}$
            $\bar{\boldsymbol{\mu}}_c^m \leftarrow \frac{\sum_{n=1}^N\left[s^n \gamma_c^{n,m}\left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}\right)\mathbf{x}^n\right] + D\boldsymbol{\mu}_c^m}{\sum_{n=1}^N\left[s^n \gamma_c^{n,m}\left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}\right)\right] + D}$
            $\boldsymbol{\Sigma}_c^m \leftarrow \frac{\sum_{n=1}^N\left[s^n \gamma_c^{n,m}\left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}\right)\left(\mathbf{x}^n\right)^2\right] + D\left(\boldsymbol{\Sigma}_c^m + \left(\boldsymbol{\mu}_c^m\right)^2\right)}{\sum_{n=1}^N\left[s^n \gamma_c^{n,m}\left(\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c\neq c^n\}} r_c^{n,\eta}\right)\right] + D} - (\bar{\boldsymbol{\mu}}_c^m)^2$
            $\boldsymbol{\mu}_c^m \leftarrow \bar{\boldsymbol{\mu}}_c^m$
        end
        $\alpha_c^m \leftarrow \bar{\alpha}_c^m \quad \forall m \in \{1, ..., M\}$
    end
    $\rho_c \leftarrow \bar{\rho}_c \quad \forall c \in \{1, ..., C\}$
end
```

Algorithm 1. Discriminative Margin-based training of GMMs (MM-GMM Algorithm).



(a) (b)

Fig. 2. Convergence of CL-GMM and MM-GMM depending on F. The x-axis denotes the number of iterations. (a) $CLL\,(\mathcal{X}|\mathbf{\Theta})$, (b) $\log D\,(\mathcal{X}|\mathbf{\Theta})$.

Table 1. Classification results in [%] on the synthetic training and test data

	GMM	CL-GMM	MM-GMM
Train Data	79.48 ± 0.40	86.47 ± 0.34	86.58 ± 0.34
Test Data	80.05 ± 0.89	85.80 ± 0.78	86.05 ± 0.77

(c) decision boundary of MM-GMM, and (d) shows the decision boundary of all learning approaches. The decision boundary of the DGMM classifiers is smoother and better approximates the original spiral data. Discriminative learning is able to change the decision boundary to improve the classification rate (see Table 1).

Furthermore, we show the evolution of both the conditional log likelihood $CLL\,(\mathcal{X}|\mathbf{\Theta})$ and the margin $\log D\,(\mathcal{X}|\mathbf{\Theta})$ depending on F over the iterations of the algorithms (see Figure 2(a) and (b)). As mentioned above, the rate of convergence of EBW strongly depends on the value of F. Additionally, the performances do not increase at each iteration. One reason is the approximation of the derivative in Eq. (8) as suggested in [18]. In [20], they experimentally observed that this approximation substantially improves convergence, although it is not guaranteed at each iteration.

5.2 Broad Phonetic classification

We use the TIMIT speech corpus [21] for broad phonetic classification. Therefore, we employ the standard NIST sets of 462 speakers and 168 speakers for training and testing, respectively. We perform frame-by-frame phone classification. We conduct experiments with only four classes and six classes using 1691462 and 1886792 samples, respectively. Moreover, we perform classification experiments on data of male speakers (Ma), female speakers (Fe), and both genders (Ma+Fe). More details about the experimental setup and the features can be found in [15]. We use the following classifiers:

- GMM: Generatively trained GMM with $M = 100$ components.
- CL-GMM: Discriminative CL-based trained GMM classifier using $M = 100$ components.

- MM-GMM: Discriminative margin-based trained GMM classifier using $M = 100$ components.
- NN-100: Neural network (multi-layered perceptron) with one hidden layer. The number of units in the input and output layer is set to the number of features and the number of classes, respectively. In the hidden layer we use 100 neurons with a hyperbolic tangent sigmoid transfer function. Levenberg-Marquardt backpropagation is used for training and the transfer functions in the output layer are linear.
- SVM-1-0.1: The support vector machine with the radial basis function (RBF) kernel uses two parameters, namely C^* and σ, where C^* is the penalty parameter for the errors of the non-separable case and σ is the parameter for the RBF kernel. We set the values for these parameters to $C^* = 1$ and $\sigma = 0.1$.

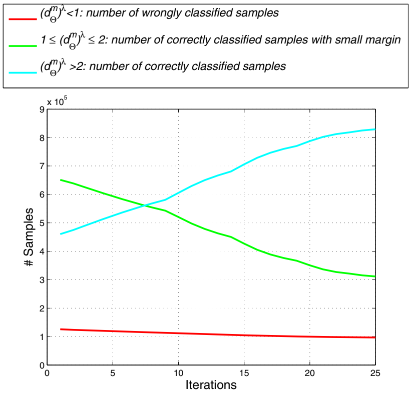
The optimal choice of the parameters (i.e., C^*, σ), number of neurons in the hidden layer, and transfer functions of the above mentioned classifiers was obtained in each case by cross-validation. The parameters for learning CL-GMM and MM-GMM are initialized to the ML estimates.

The experimental results in Figure 3(a) show that CL-GMMs achieve about the same performance compared to MM-GMMs, whereas both DGMMs perform significantly better than generatively learned GMMs. The classification results of the MM-GMM are $\approx 0.75\%$ lower compared to NNs and SVMs. The number of parameters for the DGMM is 16404 compared to 202425 and 400442 support vectors of the SVM for the Ma-Fe-4Class and Ma-Fe-6Class data, respectively. Hence, SVMs have roughly $4 \cdot 10^6$ and $8 \cdot 10^6$ parameters using the dimensionality of $d = 20$ for each support vector. This means that DGMM has almost 500 times fewer parameters than the SVM for the Ma-Fe-6Class data. Although, the classification results are slightly worse DGMMs offer advantages compared to the SVM. DGMMs can be directly applied to problems with more than two classes, whereas SVMs are usually limited to binary problems – the multiclass problem is decomposed into binary problems. However, multiclass SVMs have been proposed [22]. For SVMs we have to select C^* and σ. For MM-GMMs the number of components M and λ have to be determined. A substantial difference is that the SVMs determine the number of support vectors automatically while in the case of DGMMs the number of components M is prescribed. Hence, in DGMMs the complexity is controlled manually. DGMMs are an excellent choice when a probabilistic model is required, e.g. marginalizing over the unknown variables is supported. The training time for each iteration of the DGMM scales with $\mathcal{O}(MN)$, whereas for the SVM we have $\mathcal{O}\left(N^2\right)$. Hence, DGMMs have a lower training complexity.

In Figure 3(b), we provide an in-depth analysis of the multi-class margin $(d_\Theta^n)^\lambda$ for Ma-Fe-4 ($M = 100$). The cyan and green colored lines denote the number of correctly classified samples with a margin of $(d_\Theta^n)^\lambda > 2$ (i.e. $|\mathcal{X}^3|$) and $1 \leq (d_\Theta^n)^\lambda \leq 2$ (i.e. $|\mathcal{X}^2|$) over the iterations, respectively. The samples with margin between one and two are still considered during optimization and the algorithm tries to increase the margin above two, i.e the number of those

		Classifier				
Data set	Class	GMM	GMM CL	GMM MM	NN 100	SVM 1-0.1
Ma+Fe	4	90.17 ± 0.06	92.54 ± 0.06	92.30 ± 0.06	92.58 ± 0.06	92.78 ± 0.06
Ma	4	90.17 ± 0.08	92.50 ± 0.07	92.31 ± 0.07	92.73 ± 0.07	92.69 ± 0.07
Fe	4	90.56 ± 0.11	92.55 ± 0.10	92.63 ± 0.10	92.91 ± 0.10	92.97 ± 0.10
Ma+Fe	6	82.42 ± 0.08	85.81 ± 0.07	85.14 ± 0.07	86.05 ± 0.07	86.26 ± 0.07
Ma	6	82.49 ± 0.10	85.66 ± 0.09	85.19 ± 0.09	86.04 ± 0.09	86.16 ± 0.09
Fe	6	82.84 ± 0.14	85.74 ± 0.13	85.69 ± 0.13	86.37 ± 0.12	86.65 ± 0.12

(a)



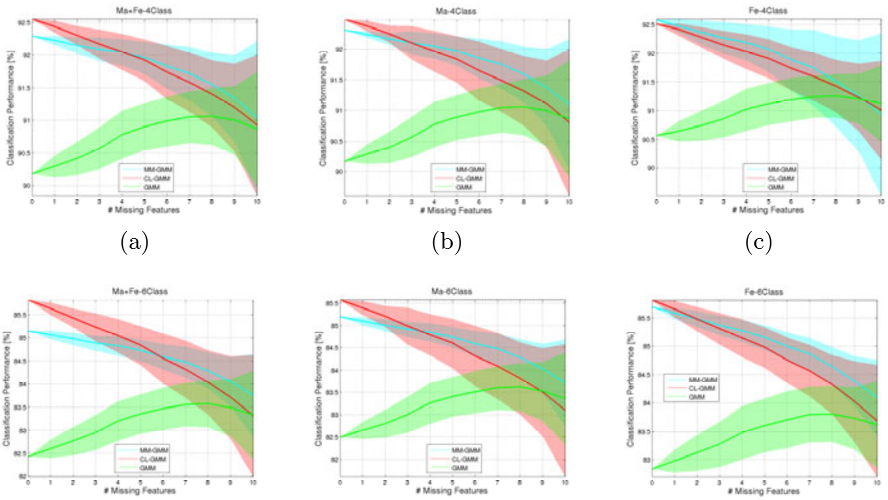
(b)

Fig. 3. Broad phonetic classification: (a) Classification accuracy in [%] for 4 and 6 classes with standard deviation. (b) Number of samples in $\mathcal{X}^1$, $\mathcal{X}^2$, and $\mathcal{X}^3$ over the iterations of MM-GMM.

samples decreases over the iterations while the number of samples with $\left(d_\Theta^n\right)^\lambda > 2$ increases. Additionally, the number of wrongly classified samples (i.e. $\left(d_\Theta^n\right)^\lambda < 1$) decreases (red line).

In the following, we verify that a discriminatively parameterized generative GMM $p\left(\mathbf{x}|\mathbf{\Theta}_c\right)$ still offers its advantages in the missing feature case. In particular, the ability to go from $p\left(\mathbf{x}|\mathbf{\Theta}_c\right)$ to $p\left(\mathbf{x}'|\mathbf{\Theta}_c\right)$ is maintained where $\mathbf{x}'$ is a subset of the features in $\mathbf{x}$ and $\mathbf{x}''$ is the set of missing features, i.e. $\mathbf{x} \setminus \mathbf{x}'$. This amounts to performing the marginalization $p\left(\mathbf{x}'|\mathbf{\Theta}_c\right) = \int p\left(\mathbf{x}|\mathbf{\Theta}_c\right) \mathrm{d}\mathbf{x}''$. A discriminative model, however, is inherently conditional and it is not possible in general to simply marginalize away any missing features. This problem is also true for SVMs, logistic regression, and neural networks.

We are particularly interested in a testing context which has arbitrary sets of missing features for each classification sample, unanticipated at training time. In such a case, it is not possible to re-train the model for each potential set of missing features without also memorizing the training set. In Figure 4, we present the classification performance of GMM, CL-GMM, and MM-GMM assuming missing features using the data of TIMIT-4/6. The x-axis denotes the number of missing features. The curves are the average over 100 classifications of the test data with uniformly at random selected missing features. Standard deviation bars indicate that the resulting differences are significant for a low number of missing features. We use exactly the same missing features for each classifier. We observe that discriminatively parameterized GMM classifiers outperform classical GMMs in the case of a low number of missing features. In case of many missing features classical GMMs seem to be more robust. The rising performance of the generative GMM classifier in case of missing features can be attributed to the phenomenon observed in the feature selection community. There, the reduction of the feature set size may even improve the classification



(a) (b) (c)

(d) (e) (f)

Fig. 4. Classification performance of GMM, CL-GMM, and MM-GMM assuming missing features using data of TIMIT-4/6. The x-axis denotes the number of missing features and the shaded region corresponds to the standard deviation over 100 classifications. (a) Ma+Fe-4, (b) Ma-4, (c) Fe-4, (d) Ma+Fe-6, (e) Ma-6, (f) Fe-6.

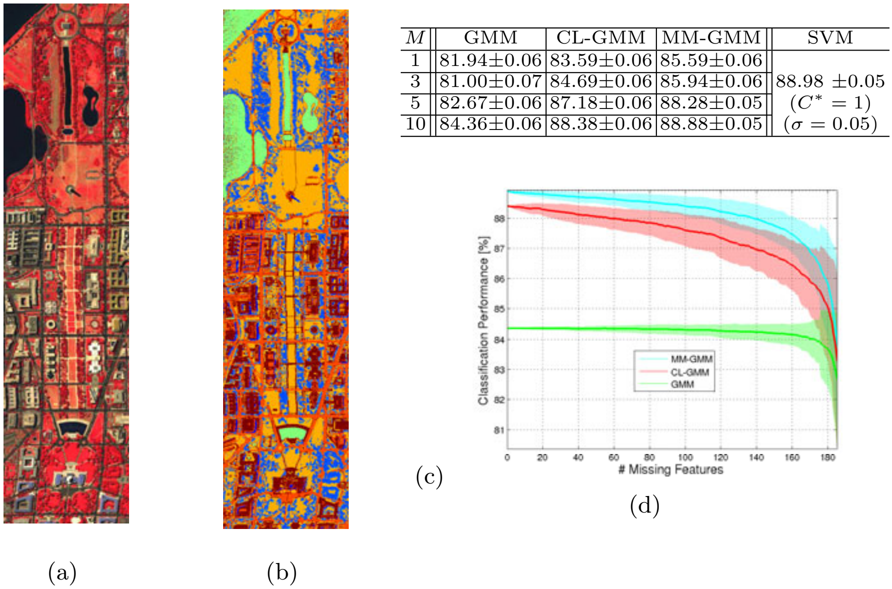
rate by reducing estimation errors associated with finite sample size effects [23]. Generally, this demonstrates, at least empirically, that discriminatively parameterized generative GMMs do not lose their ability to impute missing features.

5.3 Remote Sensing

We use a hyperspectral remote sensing image of the Washington, D.C., Mall area containing 191 spectral bands having a spectral width of 5-10 nm.[2] As ground reference a classification performed at Purdue University was used containing 7 classes, namely, roofs, road, grass, trees, trail, water, and shadow.[3] The aerial image using bands 63, 52, and 36 for red, green, and blue colors, respectively, and the reference image are shown in Figure 5(a) and (b). The image contains 1280 × 307 hyperspectral pixels, i.e. 392960 samples. We arbitrarily choose 5000 samples of each class to learn the classifier. This remote sensing application is in particular interesting for our classifiers since spectral bands might be missing or should be neglected due to atmospheric effects, i.e. radiation within the visible range should be neglected in case of clouds or darkness. We use generative GMM as well as discriminatively optimized GMM classifiers, whereas the parameters for discriminative training are initialized to ML estimates. The classification

[2] http://cobweb.ecn.purdue.edu/~biehl/MultiSpec/hyperspectral.html

[3] http://cobweb.ecn.purdue.edu/~landgreb/Hyperspectral.Ex.html



(a)

(b)

M	GMM	CL-GMM	MM-GMM	SVM
1	81.94±0.06	83.59±0.06	85.59±0.06	
3	81.00±0.07	84.69±0.06	85.94±0.06	88.98 ±0.05
5	82.67±0.06	87.18±0.06	88.28±0.05	($C^* = 1$)
10	84.36±0.06	88.38±0.06	88.88±0.05	($\sigma = 0.05$)

(c)

(d)

Fig. 5. Washington, D.C., Mall: (a) Spectral bands 63, 52, and 36 are used for pseudo color image. (b) Reference image. (c) Classification results in [%]. (d) Classification results of GMM, CL-GMM, and MM-GMM assuming missing features.

performances for $M \in \{1, 3, 5, 10\}$ components are shown in Table 5(c). CL-GMM and MM-GMM significantly outperforms the generative GMM classifier whereas best performances are obtained with MM-GMM classifiers. Remarkably, MM-GMMs and SVMs achieve a highly similar performance. The number of parameters for the GMM if roughly 85 times lower than for SVMs (26817 versus 2279394 (i.e. 11934 support vectors, N=191)).

In Figure 5(d), we report classification results for GMM, CL-GMM, and MM-GMM using $M = 10$ components assuming missing features. The x-axis denotes the number of missing features. We average the performances over 100 classifications of the test data with randomly missing features. Standard deviation bars indicate that the resulting differences are significant for a low number of missing features. Discriminatively parameterized GMM classifiers significantly outperform classical GMMs in the case of few missing features.

6 Conclusions

We derive two discriminative training methods for GMM-based Bayesian classifiers maximizing either the conditional likelihood or the margin. Both algorithms are based on the extended Baum-Welch (EBW) algorithm. In the experiments we depict the differences of the decision boundary for generatively and discriminatively learned GMMs for classification using synthetic data. Furthermore, we show results for broad phonetic classification and compare discriminatively optimized GMM classifiers to SVMs and NNs. DGMMs perform slightly worse compared to SVMs in terms of classification rate, however the GMM model uses almost 500 times fewer parameters than the SVM. Additionally, we show that discriminatively optimized GMM classifiers are superior even in the case of missing features. Finally, we compare our classifiers on a hyperspectral remote sensing application which is in particular interesting concerning the missing feature aspect. Margin-based GMMs outperform CL-based GMMs, whereas both significantly outperform generatively optimized GMMs.

References

1. Gopalakrishnan, O., Kanevsky, D., Nàdas, A., Nahamoo, D.: An inequality for rational functions with applications to some statistical estimation problems. IEEE Transactions on Information Theory 37(1), 107–113 (1991)
2. Vapnik, V.: Statistical learning theory. Wiley & Sons, Chichester (1998)
3. Schölkopf, B., Smola, A.: Learning with kernels: Support Vector Machines, regularization, optimization, and beyond. MIT Press, Cambridge (2001)
4. Taskar, B., Guestrin, C., Koller, D.: Max-margin markov networks. In: Advances in Neural Information Processing Systems, NIPS (2003)
5. Guo, Y., Wilkinson, D., Schuurmans, D.: Maximum margin Bayesian networks. In: International Conference on Uncertainty in Artificial Intelligence, UAI (2005)
6. Roos, T., Wettig, H., Grünwald, P., Myllymäki, P., Tirri, H.: On discriminative Bayesian network classifiers and logistic regression. Machine Learning 59, 267–296 (2005)
7. Sha, F., Saul, L.: Large margin Gaussian mixture modeling for phonetic classification and recognition. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP (2006)
8. Sha, F., Saul, L.: Comparison of large margin training to other discriminative methods for phonetic recognition by hidden Markov models. In: IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 313–316 (2007)
9. Heigold, G., Deselaers, T., Schlüter, R., Ney, H.: Modified MMI/MPE: A direct evaluation of the margin in speech recognition. In: International Conference on Machine Learning (ICML), pp. 384–391 (2008)
10. Collobert, R., Siz, F., Weston, J., Bottou, L.: Trading convexity for scalability. In: International Conference on Machine Learning (ICML), pp. 201–208 (2006)
11. Schlüter, R., Macherey, W., Müller, B., Ney, H.: Comparison of discriminative training criteria and optimization methods for speech recognition. Speech Communication 34, 287–310 (2001)
12. Bahl, L., Brown, P., de Souza, P., Mercer, R.: Maximum Mutual Information estimation of HMM parameters for speech recognition. In: IEEE Conf. on Acoustics, Speech, and Signal Proc., pp. 49–52 (1986)
13. Woodland, P., Povey, D.: Large scale discriminative training of hidden Markov models for speech recognition. Computer Speech and Language 16, 25–47 (2002)
14. Klautau, A., Jevtić, N., Orlitsky, A.: Discriminative Gaussian mixture models: A comparison with kernel classifiers. In: Inter. Conf. on Machine Learning (ICML), pp. 353–360 (2003)
15. Pernkopf, F., Van Pham, T., Bilmes, J.: Broad phonetic classification using discriminative Bayesian networks. Speech Communication 143(1), 123–138 (2008)

16. Bishop, C.M.: Pattern recognition and machine learning. Springer, Heidelberg (2006)
17. Pernkopf, F., Bouchaffra, D.: Genetic-based EM algorithm for learning Gaussian mixture models. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(8), 1344–1348 (2005)
18. Merialdo, B.: Phonetic recognition using hidden Markov models and maximum mutual information training. In: IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 111–114 (1988)
19. Normandin, Y., Morgera, S.: An improved MMIE training algorithm for speaker-independent small vocabulary, continuous speech recognition. In: IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 537–540 (1991)
20. Normandin, Y., Cardin, R., De Mori, R.: High-performance connected digit recognition using maximum mutual information estimation. IEEE Trans. on Speech and Audio Proc. 2(2), 299–311 (1994)
21. Lamel, L., Kassel, R., Seneff, S.: Speech database development: Design and analysis of the acoustic-phonetic corpus. In: DARPA Speech Recognition Workshop, Report No. SAIC-86/1546 (1986)
22. Crammer, K., Singer, Y.: On the algorithmic interpretation of multiclass kernel-based vector machines. Journal of Machine Learning Research 2, 265–292 (2001)
23. Jain, A., Chandrasekaran, B.: Dimensionality and sample size considerations in pattern recognition in practice. Handbook of Statistics, vol. 2. North-Holland, Amsterdam (1982)
24. Baum, L., Eagon, J.: An inequality with applications to statistical prediction for functions of Markov processes and to a model of ecology. Bull. Amer. Math. Soc. 73, 360–363 (1967)

Appendix A: EBW Algorithm

In its original form [24], the Baum-Eagon inequality has been formulated for domains of discrete probabilities. Consider a domain E of discrete probability values $\mathbf{\Phi} = \{\varphi_i^j\}$, with $\varphi_i^j \geq 0$, $\sum_i \varphi_i^j = 1$, and $j = 1, ..., J$. Given a homogeneous polynomial $Q(\mathbf{\Phi})$ with nonnegative coefficients over the domain E, the Baum-Eagon inequality offers an iterative method to find local maxima in Q. It provides a transformation, $T : E \to E$, such that $Q\left(T\left(\mathbf{\Phi}\right)\right) > Q\left(\mathbf{\Phi}\right))$, unless $T\left(\mathbf{\Phi}\right) = \mathbf{\Phi}$. This transformation, called *growth transform*, maps from $\hat{\mathbf{\Phi}} \in E$ to $T(\hat{\mathbf{\Phi}}) = \bar{\mathbf{\Phi}} \in E$, where

$$\bar{\varphi}_i^j = \frac{\hat{\varphi}_i^j \frac{\partial Q(\hat{\mathbf{\Phi}})}{\partial \varphi_i^j}}{\sum_{i'} \hat{\varphi}_{i'}^j \frac{\partial Q(\hat{\mathbf{\Phi}})}{\partial \varphi_{i'}^j}}. \tag{11}$$

For brevity, $\frac{\partial Q(\hat{\mathbf{\Phi}})}{\partial \varphi_i^j}$ denotes the partial derivative $\frac{\partial Q}{\partial \varphi_i^j}$ evaluated at point $\hat{\mathbf{\Phi}}$.

In [1], the growth transform is extended[4] to rational functions $R(\mathbf{\Phi})$ over E:

$$R\left(\mathbf{\Phi}\right) = \frac{\mathrm{Num}(\mathbf{\Phi})}{\mathrm{Den}(\mathbf{\Phi})}.$$

[4] Additionally, they show that the growth transform in Eq. (11) can be applied to nonhomogeneous polynomials.

This is done by converting $R(\mathbf{\Phi})$ into a polynomial $Q_{\hat{\mathbf{\Phi}}}(\mathbf{\Phi})$ for a given $\hat{\mathbf{\Phi}}$ such that if $Q_{\hat{\mathbf{\Phi}}}\left(T\left(\hat{\mathbf{\Phi}}\right)\right) > Q_{\hat{\mathbf{\Phi}}}(\hat{\mathbf{\Phi}})$, then $R\left(T\left(\hat{\mathbf{\Phi}}\right)\right) > R\left(\hat{\mathbf{\Phi}}\right)$, except $T\left(\hat{\mathbf{\Phi}}\right) = \hat{\mathbf{\Phi}}$. The polynomial $Q_{\hat{\mathbf{\Phi}}}(\mathbf{\Phi})$ that fulfills this condition is given in [1] as

$$Q_{\hat{\mathbf{\Phi}}}(\mathbf{\Phi}) = \mathrm{Num}(\mathbf{\Phi}) - R(\hat{\mathbf{\Phi}})\mathrm{Den}(\mathbf{\Phi}).$$

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

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

$$S_{\hat{\mathbf{\Phi}}}(\mathbf{\Phi}) = Q_{\hat{\mathbf{\Phi}}}(\mathbf{\Phi}) + C(\mathbf{\Phi}),$$

where

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

has constant value over E, since $\sum_i \varphi_i^j = 1$, and r denotes the maximal order of $Q_{\hat{\mathbf{\Phi}}}(\mathbf{\Phi})$. Hence, $C(\mathbf{\Phi})$ adds a constant κ to every monomial in $Q_{\hat{\mathbf{\Phi}}}(\mathbf{\Phi})$. This constant κ must be chosen such that $S_{\hat{\mathbf{\Phi}}}(\mathbf{\Phi})$ has nonnegative coefficients for every $\hat{\mathbf{\Phi}}$. Thus, $S_{\hat{\mathbf{\Phi}}}(\mathbf{\Phi})$ has positive coefficients and still has the same important property as $Q_{\hat{\mathbf{\Phi}}}(\mathbf{\Phi})$. This polynomial with positive coefficients can now be considered for the growth transform in Eq. (11).

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

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

where $D = \kappa r(J+1)^{r-1}$ is the derivative of $C(\mathbf{\Phi})$. Plugging this result into Eq. (11), we finally obtain the extended Baum-Welch re-estimation equation for discrete probability distributions of the form

$$\bar{\varphi}_i^j = \frac{\hat{\varphi}_i^j\left(\frac{\partial \log R\left(\hat{\mathbf{\Phi}}\right)}{\partial \varphi_i^j} + D\right)}{\sum_{i'}\hat{\varphi}_{i'}^j\left(\frac{\partial \log R\left(\hat{\mathbf{\Phi}}\right)}{\partial \varphi_{i'}^j} + D\right)}, \tag{12}$$

where the $\bar{\varphi}_i^j$ denotes the updated parameters, and constant D must be chosen to be sufficiently large.