On Discriminative Parameter Learning of Bayesian Network Classifiers

Franz Pernkopf and Michael Wohlmayr

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

Abstract. We introduce three discriminative parameter learning algorithms for Bayesian network classifiers based on optimizing either the conditional likelihood (CL) or a lower-bound surrogate of the CL. One training procedure is based on the extended Baum-Welch (EBW) algorithm. Similarly, the remaining two approaches iteratively optimize the parameters (initialized to ML) with a 2-step algorithm. In the first step, either the class posterior probabilities or class assignments are determined based on current parameter estimates. Based on these posteriors (class assignment, respectively), the parameters are updated in the second step. We show that one of these algorithms is strongly related to EBW. Additionally, we compare all algorithms to conjugate gradient conditional likelihood (CGCL) parameter optimization [1].
We present classification results for frame- and segment-based phonetic classification and handwritten digit recognition. Discriminative parameter learning shows a significant improvement over generative ML estimation for naive Bayes (NB) and tree augmented naive Bayes (TAN) structures on all data sets. In general, the performance improvement of discriminative parameter learning is large for simple Bayesian network structures which are not optimized for classification.

1 Introduction

There are two avenues for learning statistical classifiers: Generative and discriminative approaches [2–4]. In generative classifiers we learn a model of the joint probability of the features and the corresponding class label and perform predictions by using Bayes rule to determine the class posterior probability. ML estimation is usually used to learn a generative classifier. Discriminative approaches model the class posterior probability directly. Hence, the class conditional probability is optimized when we learn the classifier which is most important for classification accuracy. There are several reasons for using discriminative rather than generative classifiers, one of which is that the classification problem should be solved most simply and directly, and never via a more general problem such as the intermediate step of estimating the joint distribution [5]. However, there are also a number of reasons why in certain contexts a generative model is preferred including: parameter tying and domain knowledge-based hierarchical decomposition is facilitated; it is easy to work with structured data; and it is easy to work with missing features by marginalizing over the unknown variables.

The expectation-maximization (EM) algorithm is commonly used for generatively learning hidden variable models, e.g. Gaussian mixtures. It optimizes a globally valid lower bound of the likelihood function [6] which therefore guarantees an increase of the likelihood itself. The straightforward application of the EM algorithm to optimize the CL is not possible since we have to optimize a rational function and the constructed lower bounds are only locally valid, i.e. for current parameter estimates [7]. Due to this fact, convergence is not stringent. In [4] and [7] a global lower bound for *EM-like* CL optimization algorithms has been proposed.

A sufficient (but not necessary) condition for optimal classification is for the conditional likelihood (CL) to be optimized. Unfortunately, the CL function for Bayesian networks does not decompose and there is no closed-form solution for determining its parameters. In current approaches, the structure and/or the parameters are learned in a discriminative manner by maximizing CL[1]. Greiner et al. [1] express general Bayesian networks as standard logistic regression – they optimize parameters with respect to the conditional likelihood using a conjugate gradient method. Similarly, Roos et al. [8] provide conditions for general Bayesian networks under which the correspondence to logistic regression holds. An empirical and theoretical comparison of discriminative and generative classifiers (logistic regression and the NB classifier) is given in [9]. It is shown that for small sample sizes the generative NB classifier can outperform the discriminatively trained model. An experimental comparison of discriminative and generative parameter training on both discriminatively and generatively structured Bayesian network classifiers has been performed in [10].

The CL function is closely related to the maximum mutual information (MMI) criterion which is popular in the speech community [2, 11]. It was proposed for hidden Markov models (HMM) [2] and attempts to maximize the posterior probability of the transcriptions given the utterances. In this context, the extended Baum-Welch (EBW) algorithm [12, 13] has been introduced to discriminatively optimize HMMs. In [14], it has been applied to optimize Gaussian mixture models.

In this paper, we present three discriminative *EM-like* parameter learning methods for Bayesian network classifiers and compare them to conjugate gradient CL optimization [1]. As first method, we introduce EBW for Bayesian networks. The two remaining approaches are based on optimizing either the CL or a lower-bound surrogate of it. The algorithms are abbreviated as ECL (exact CL decomposition) and ACL (approximate CL decomposition), respectively. Our ACL algorithm has been formulated for HMMs in [11]. In fact, we can show that ECL with Laplace smoothing is closely related to EBW. Both algorithms (i.e. ECL and ACL) iteratively optimize the parameters (initialized to ML) with a 2-step algorithm similar as in [11]. In the first step, the class posterior probabilities for ECL (class assignments for ACL, respectively) are calculated based

[1] By "discriminative structure learning", we mean that the aim of optimization is to maximize a cost function that is suitable for reducing classification errors, such as conditional likelihood or classification rate.

on current parameter estimates. Based on these posteriors (class assignments, respectively), the parameters are updated in the second step. Additionally, we have to introduce mechanisms to avoid negative values for probabilities. However, both algorithms do not show a monotone improvement of the objective function as we have for the conjugate gradient approach [1] and the EBW method. Nevertheless, we obtain excellent results at low computational costs for two phonetic classification tasks using the TIMIT speech corpus [15] and for handwritten digit recognition using the MNIST and USPS data sets. Discriminative parameter learning significantly outperforms ML parameter estimation for NB and TAN structures. In general, the performance improvement is larger for simple structures which are not optimized for classification, i.e. for structures which are not optimized with respect to the CL or the classification rate (CR) [1].

The paper is organized as follows: In Section 2, we introduce our notation and briefly review Bayesian networks, ML parameter learning, NB, TAN, and 2-tree structures. In Section 3, we introduce all discriminative parameter learning algorithms, i.e. ECL, ACL, EBW, and CGCL. In Section 4, we present experimental results. Section 5 concludes the paper.

2 Bayesian Network Classifiers

A Bayesian network [16] $\mathcal{B} = \langle \mathcal{G}, \Theta \rangle$ is a directed acyclic graph $\mathcal{G} = (\mathbf{Z}, \mathbf{E})$ consisting of a set of nodes $\mathbf{Z}$ and a set of directed edges connecting the nodes. This graph represents factorization properties of the distribution of a set of random variables $\mathbf{Z} = \{Z_1, \ldots, Z_{N+1}\}$. Each variable in $\mathbf{Z}$ has values denoted by lower case letters $\{z_1, \ldots, z_{N+1}\}$. We use boldface capital letters, e.g. $\mathbf{Z}$, to denote a set of random variables and correspondingly boldface lower case letters denote a set of instantiations (values). Without loss of generality, in Bayesian network classifiers the random variable Z_1 represents the class variable $C \in \{1, \ldots, |C|\}$, $|C|$ is the cardinality of C or equivalently the number of classes, $\mathbf{X}_{1:N} = \{X_1, \ldots, X_N\} = \{Z_2, \ldots, Z_{N+1}\}$ denote the set of random variables of the N attributes of the classifier. In a Bayesian network each node is independent of its non-descendants given its parents [16]. Conditional independence reduces computation for exact inference on such a graph. The set of parameters which quantify the network are represented by Θ. Each node Z_j is represented as a local conditional probability distribution given its parents Z_{Π_j}. We use $\theta^j_{i|h}$ to denote a specific conditional probability table entry (assuming discrete variables), the probability that variable Z_j takes on its i^{th} value assignment given that its parents Z_{Π_j} take their h^{th} (lexicographically ordered) assignment. That is, $\theta^j_{i|h} = P_\Theta\left(z_j = i | z_{\Pi_j} = h\right) = \prod_{i=1}^{|Z_j|} \prod_h \left(\theta^j_{i|h}\right)^{u^j_{i|h}}$, where

$$u^j_{i|h} = \begin{cases} 1, \text{ if } z_j = i \text{ and } z_{\Pi_j} = h \\ 0, \text{ otherwise} \end{cases}. \tag{1}$$

Hence, h contains the parent configuration assuming that the first element of h, i.e. h_1, relates to the conditioning class and the remaining elements $h \backslash h_1$ denote

the conditioning on parent attribute values. The training data consists of M independent and identically distributed samples $\mathcal{S} = \{\mathbf{z}^m\}_{m=1}^M = \{(c^m, \mathbf{x}_{1:N}^m)\}_{m=1}^M$. The joint probability distribution of the network is determined by the local conditional probability distributions as

$$P_\Theta\left(\mathbf{Z} = \mathbf{z}^m\right) = \prod_{j=1}^{N+1} P_\Theta\left(Z_j = z_j^m | Z_{\Pi_j} = z_{\Pi_j}^m\right) = \prod_{j=1}^{N+1} \prod_{i=1}^{|Z_j|} \prod_h \left(\theta_{i|h}^j\right)^{u_{i|h}^{j,m}}, \quad (2)$$

where $u_{i|h}^{j,m}$ is the obvious extension of $u_{i|h}^j$ to the m^{th} sample.

2.1 Generative ML Parameter Learning

The parameters of the generative model are learned by maximizing the log likelihood of the data which leads to the ML estimation of $\theta_{i|h}^j$. The log likelihood function of a fixed structure of $\mathcal{B}$ is

$$LL\left(\mathcal{B}|\mathcal{S}\right) = \sum_{m=1}^M \log P_\Theta\left(\mathbf{Z} = \mathbf{z}^m\right) = \sum_{m=1}^M \sum_{j=1}^{N+1} \sum_{i=1}^{|Z_j|} \sum_h u_{i|h}^{j,m} \log\left(\theta_{i|h}^j\right).$$

It is easy to show that the ML estimate of the parameters is

$$\theta_{i|h}^j = \frac{\sum_{m=1}^M u_{i|h}^{j,m}}{\sum_{m=1}^M \sum_{l=1}^{|Z_j|} u_{l|h}^{j,m}}, \quad (3)$$

using Lagrange multipliers to constrain the parameters to a valid normalized probability distribution, i.e. $\sum_{i=1}^{|Z_j|} \theta_{i|h}^j = 1$.

2.2 Structures

In this paper, we restrict our experiments to NB, TAN, and 2-tree classifier structures. The NB network assumes that all the attributes are conditionally independent given the class label. This means that, given C, any subset of $\mathbf{X}$ is independent of any other disjoint subset of $\mathbf{X}$. As reported in the literature [17], the performance of the NB classifier is surprisingly good even if the conditional independence assumption between attributes is unrealistic or even wrong for most of the data.

In order to correct some of the limitations of the NB classifier, Friedman et al. [17] introduced the TAN classifier. A TAN is based on structural augmentations of the NB network, where additional edges are added between attributes in order to relax some of the most flagrant conditional independence properties of NB. Each attribute may have at most one other attribute as an additional parent which means that the tree-width of the attribute induced sub-graph is unity, i.e. we have to learn a 1-tree over the attributes. The maximum number of edges added to relax the independence assumption between the attributes is $N - 1$.

A TAN network is typically initialized as a NB network and additional edges between attributes are determined through structure learning. An extension of the TAN network is the k-tree, where each attribute can have a maximum of k attribute nodes as parents. In this work, TAN and k-tree structures are restricted such that the class node remains parent-less, i.e. $C_{\Pi} = \emptyset$.

In the experiments, we apply one generative structure learning algorithm introduced in Friedman et al. [17] for constructing a TAN structure using the conditional mutual information (CMI) [18]. Additionally, we use two discriminative structure learning algorithms: a simple greedy heuristic [19] and an order-based greedy algorithm [20]. Both methods consider CR as scoring function.

3 Discriminative CL Parameter Learning

Optimizing CL is tightly connected to good classification performance. Hence, we want to learn parameters so that CL is maximized. Unfortunately, CL does not decompose as ML does. Consequently, there is no closed-form solution. The objective function of the conditional log likelihood (CLL) is

$$\begin{aligned} CLL\left(\mathcal{B}|\mathcal{S}\right) &= \log \prod_{m=1}^{M} P_{\Theta}\left(C = c^m | \mathbf{X}_{1:N} = \mathbf{x}_{1:N}^m\right) = \\ &\sum_{m=1}^{M} \left[\log P_{\Theta}\left(C = c^m, \mathbf{X}_{1:N} = \mathbf{x}_{1:N}^m\right) - \log \sum_{c=1}^{|C|} P_{\Theta}\left(C = c, \mathbf{X}_{1:N} = \mathbf{x}_{1:N}^m\right) \right]. \end{aligned} \tag{4}$$

Greiner et al. [1] use a conjugate gradient descent algorithm with line-search to optimize $CLL\left(\mathcal{B}|\mathcal{S}\right)$. In contrast, we aim to optimize the CL directly using an iterative *EM-like* procedure. In the following subsections, we introduce EBW, ECL, and ACL parameter learning for Bayesian networks. Additionally, we shortly review CGCL [1]. For the sake of brevity, we only notate instantiations of the random variables in the probabilities.

3.1 ECL Algorithm

We want to optimize $CLL\left(\mathcal{B}|\mathcal{S}\right)$ (see Eq. (4)) under the constraints

$$\sum_{i=1}^{|Z_j|} \theta_{i|h}^{j} = 1 \qquad \forall h, j \tag{5}$$

using Lagrange multipliers ω_h^j. The Lagrangian function is given according to

$$\begin{aligned} L\left(\Theta, \omega\right) = &\sum_{m=1}^{M} \left[\log P_{\Theta}\left(c^m, \mathbf{x}_{1:N}^m\right) - \log \sum_{c=1}^{|C|} P_{\Theta}\left(c, \mathbf{x}_{1:N}^m\right) \right] - \\ &\sum_{j=1}^{N+1} \sum_{h} \omega_h^j \left(1 - \sum_{i=1}^{|Z_j|} \theta_{i|h}^{j} \right). \end{aligned}$$

The derivative of the Lagrangian function is

$$\frac{\partial L\left(\Theta,\omega\right)}{\partial\theta_{i|h}^{j}}=\sum_{m=1}^{M}\left[\frac{\partial}{\partial\theta_{i|h}^{j}}\log P_{\Theta}\left(c^{m},\mathbf{x}_{1:N}^{m}\right)-\frac{\frac{\partial}{\partial\theta_{i|h}^{j}}\sum\limits_{c=1}^{|C|}P_{\Theta}\left(c,\mathbf{x}_{1:N}^{m}\right)}{\sum\limits_{c=1}^{|C|}P_{\Theta}\left(c,\mathbf{x}_{1:N}^{m}\right)}\right]-\omega_{h}^{j}. \quad (6)$$

For TAN, NB, or 2-tree structures each parameter $\theta_{i|h}^{j}$ involves the class node value, either $C=i$ for $j=1$ or $C=h_1$ for $j>1$ where h_1 denotes the class instantiation $h_1\in h$. Due to this fact, only one summand remains nonzero in $\frac{\partial}{\partial\theta_{i|h}^{j}}\sum\limits_{c=1}^{|C|}P_{\Theta}\left(c,\mathbf{x}_{1:N}^{m}\right)$ of Eq. (6). We distinguish two cases for deriving the Lagrangian: class variable ($j=1$) and attribute variables ($j>1$).

Case 1 For the class variable, i.e. $j=1$ and $h=\emptyset$, we get

$$\frac{\partial L\left(\Theta,\omega\right)}{\partial\theta_{i}^{1}}=\sum_{m=1}^{M}\left[\frac{\partial}{\partial\theta_{i}^{1}}\sum_{i=1}^{|C|}u_{i}^{1,m}\log\left(\theta_{i}^{1}\right)\right.$$
$$\left.-\frac{P_{\Theta}\left(i,\mathbf{x}_{1:N}^{m}\right)}{\sum\limits_{c=1}^{|C|}P_{\Theta}\left(c,\mathbf{x}_{1:N}^{m}\right)}\frac{\partial}{\partial\theta_{i}^{1}}\log P_{\Theta}\left(i,\mathbf{x}_{1:N}^{m}\right)\right]-\omega_{h}^{j}=\sum_{m=1}^{M}\left[\frac{u_{i}^{1,m}}{\theta_{i}^{1}}-\frac{W_{i}^{m}}{\theta_{i}^{1}}\right]-\omega_{h}^{j}=0, \quad (7)$$

where we use Eq. (2) for deriving the first term (omitting the sum over j and h) and we introduced the class posterior $W_{i}^{m}=P_{\Theta}\left(i|\mathbf{x}_{1:N}^{m}\right)$ as

$$W_{i}^{m}=\frac{P_{\Theta}\left(i,\mathbf{x}_{1:N}^{m}\right)}{\sum\limits_{c=1}^{|C|}P_{\Theta}\left(c,\mathbf{x}_{1:N}^{m}\right)}.$$

Multiplying Eq. (7) by θ_{i}^{1} and summing over i, we can determine ω_{h}^{j} as

$$\omega_{h}^{j}=\sum_{m=1}^{M}\sum_{i=1}^{|C|}\left[u_{i}^{1,m}-W_{i}^{m}\right],$$

using the constraint of Eq. (5). Finally, we get for the parameters θ_{i}^{1}

$$\theta_{i}^{1}=\frac{\sum\limits_{m=1}^{M}\left[u_{i}^{1,m}-\lambda W_{i}^{m}\right]}{\sum\limits_{m=1}^{M}\sum\limits_{i=1}^{|C|}\left[u_{i}^{1,m}-\lambda W_{i}^{m}\right]}, \quad (8)$$

where we introduced λ to weight the posterior W_{i}^{m} (to be described in the sequel).

Case 2 For the attribute variables, i.e. $j > 1$, we derive correspondingly and have

$$\frac{\partial L\left(\Theta,\omega\right)}{\partial\theta_{i|h}^{j}}=\sum_{m=1}^{M}\left[\frac{u_{i|h}^{j,m}}{\theta_{i|h}^{j}}-W_{h_1}^{m}\frac{v_{i|h\backslash h_1}^{j,m}}{\theta_{i|h}^{j}}\right]-\omega_{h}^{j}=0,$$

where $W_{h_1}^{m}=P_{\Theta}\left(h_1|\mathbf{x}_{1:N}^{m}\right)$ is the posterior for class h_1 and sample m, and

$$v_{i|h\backslash h_1}^{j,m}=\begin{cases}1, \text{ if } z_{j}^{m}=i \text{ and } z_{\Pi_j}^{m}=h\backslash h_1\\ 0, \text{ otherwise}\end{cases}.$$

Employing the constraint from Eq. (5) we obtain the parameters $\theta_{i|h}^{j}$ as

$$\theta_{i|h}^{j}=\frac{\sum_{m=1}^{M}\left[u_{i|h}^{j,m}-\lambda W_{h_1}^{m}v_{i|h\backslash h_1}^{j,m}\right]}{\sum_{m=1}^{M}\sum_{i=1}^{|Z_j|}\left[u_{i|h}^{j,m}-\lambda W_{h_1}^{m}v_{i|h\backslash h_1}^{j,m}\right]}. \tag{9}$$

Again, we introduced λ. Its value is in the range of $0 \leq \lambda \leq 1$. If we set λ to zero the second part of both equations vanishes and we obtain ML parameter learning (see Section 2.1). In the *discriminative* parameter learning case (i.e. $\lambda > 0$), parameters $\theta_{i|h}^{j}$ are affected by the samples m which have a large absolute value for the quantity $u_{i|h}^{j,m}-\lambda W_{h_1}^{m}v_{i|h\backslash h_1}^{j,m}$ (resp. $u_{i}^{1,m}-\lambda W_{i}^{m}$). These are the training samples belonging to class h_1 (resp. i for Eq. (8)) which have a low probability of being classified as h_1 (resp. i) under the current Θ, or the samples which are not in class h_1 (resp. i for Eq. (8)) but which have a large probability of being classified as h_1 (resp. i) [21, 3]. Discriminative learning is concerned with establishing the optimal classification boundary with those samples which might be easily misclassified. Data samples which are simple to classify do not contribute much for discriminative parameter learning. In contrast, generative ML parameter learning optimizes the distribution of the samples belonging to a certain class irrespective of the samples from other classes.

A solution for Eq. (8) and (9) can be determined by the following iterative two step algorithm:

1. Estimate the posterior W_{i}^{m} ($W_{h_1}^{m}$ respectively) using the old parameters Θ.
2. Given W_{i}^{m} and $W_{h_1}^{m}$ from step 1, update the parameters according to Eq. (8) and (9).

The parameters Θ can be initialized randomly, however, empirical results showed that initialization of Θ to the ML estimates leads to better performance. Both steps are repeated iteratively until a specified number of iterations is reached. ECL parameter learning is performed once the structure of the Bayesian network is determined. For large values of λ the numerator/denominator in Eq. (8) and (9) might become negative. To tackle this, we introduce the following two strategies:

– Laplace-like smoothing: We introduce a discounting value D_{h}^{j} for variable j and conditioning parent values h for Eq. (9) (similarly for Eq. (8)) and

obtain the parameters according to

$$\theta_{i|h}^{j} = \frac{-D_h^j + \sum_{m=1}^{M} \left[u_{i|h}^{j,m} - \lambda W_{h_1}^m v_{i|h\setminus h_1}^{j,m} \right]}{-|Z_j| D_h^j + \sum_{m=1}^{M} \sum_{i=1}^{|Z_j|} \left[u_{i|h}^{j,m} - \lambda W_{h_1}^m v_{i|h\setminus h_1}^{j,m} \right]}, \tag{10}$$

where

$$D_h^j = \min_{i \in |Z_j|} \left\{ 0, \sum_{m=1}^{M} \left[u_{i|h}^{j,m} - \lambda W_{h_1}^m v_{i|h\setminus h_1}^{j,m} \right] \right\}$$

– ML fallback: We set $\theta_{i|h}^{j}$ to the ML estimates of Eq. (3) for all i in case $\min_{i \in |Z_j|} \left\{ \sum_{m=1}^{M} \left[u_{i|h}^{j,m} - \lambda W_{h_1}^m v_{i|h\setminus h_1}^{j,m} \right] \right\} < 0$.

3.2 ACL Algorithm

In the approximated CLL optimization method we can find for the second term of the CL in Eq. (4), i.e. for the marginalization over C, a lower and an upper bound according to

$$\log \left(|C| \max_i F_i \right) \geq \log \sum_i F_i \geq \log \left(\max_i F_i \right).$$

Using the first inequality, we obtain a lower bound of the objective function

$$CLL\left(\mathcal{B}|\mathcal{S}\right) \geq \sum_{m=1}^{M} \left[\log P_\Theta \left(c^m, \mathbf{x}_{1:N}^m \right) - \log \max_i P_\Theta \left(i, \mathbf{x}_{1:N}^m \right) \right] - \log |C|,$$

where the last term $\log |C|$ is constant and can be neglected. Further,

$$CLL\left(\mathcal{B}|\mathcal{S}\right) \geq \sum_{m=1}^{M} \log P_\Theta \left(c^m, \mathbf{x}_{1:N}^m \right) - \sum_{c=1}^{|C|} \sum_{v \in B_c} \log P_\Theta \left(c, \mathbf{x}_{1:N}^v \right), \tag{11}$$

where the set B_c contains the indices of samples recognized as class c ($B_c \equiv \{m | c^m = \arg\max_i P_\Theta \left(i, \mathbf{x}_{1:N}^m \right)\}$). Equivalently, we introduce a modified training data set $\mathcal{S}_W$ by changing the class label of each sample m to the most probable class under the current Θ, i.e. $\mathcal{S}_W = \{(\arg\max_i P_\Theta \left(C = i, \mathbf{x}_{1:N}^m \right), \mathbf{x}_{1:N}^m)\}_{m=1}^{M}$. Thus, we can rewrite the lower bound of the objective function in Eq. (11) as

$$J\left(\Theta\right) = \sum_{m=1}^{M} \left[\log P_\Theta \left(c^{m,\mathcal{S}}, \mathbf{x}_{1:N}^{m,\mathcal{S}} \right) - \lambda \log P_\Theta \left(c^{m,\mathcal{S}_W}, \mathbf{x}_{1:N}^{m,\mathcal{S}_W} \right) \right],$$

where $0 \leq \lambda \leq 1$ determines the influence of the *discriminative* part of the objective function and the superscripts $\mathcal{S}$ and $\mathcal{S}_W$ refer to the corresponding data set. If $\lambda = 0$ the ML objective function is obtained (see Section 2.1). Similar as in

Section 3.1, we can use Lagrange multipliers to include the parameter constraint (see Eq. (5)). The derivative of the Lagrangian leads to the expression for the parameters $\theta^j_{i|h}$ according to

$$\theta^j_{i|h} = \frac{\sum_{m=1}^{M} \left[u^{j,m}_{i|h} - \lambda w^{j,m}_{i|h} \right]}{\sum_{l=1}^{|Z_j|} \sum_{m=1}^{M} \left[u^{j,m}_{l|h} - \lambda w^{j,m}_{l|h} \right]}, \tag{12}$$

where $w^{j,m}_{i|h}$ for the class variable ($j = 1$) is

$$w^{1,m}_i = \begin{cases} 1, \text{ if } z^m_1 = i = \arg\max_{k \in C} P_\Theta\left(k, \mathbf{x}^m_{1:N}\right) \\ 0, \text{ otherwise} \end{cases}$$

and for the remaining variables ($j > 1$) we get

$$w^{j,m}_{i|h} = \begin{cases} 1, \text{ if } z^m_j = i \text{ and } z^m_{\Pi_j} = \{h^*_1 \cup h \backslash h_1\} \\ 0, \text{ otherwise} \end{cases},$$

where we replaced the original class label (i.e. h_1) by the most likely class (i.e. $h^*_1 = \arg\max_{k \in C} P_\Theta\left(k, \mathbf{x}^m_{1:N}\right)$) using the current parameter estimates. The parameters $\theta^j_{i|h}$ can be optimized in a discriminative manner using an algorithm analog to the iterative method introduced above (see Section 3.1), i.e. we have the following two steps per iteration:

1. Classify the data set $\mathcal{S}$ to establish $\mathcal{S}_W$ (which directly relates to $w^{1,m}_i$ and $w^{j,m}_{i|h}$) according to $\mathcal{S}_W = \{(\arg\max_i P_\Theta\left(C = i, \mathbf{X}_{1:N} = \mathbf{x}^m_{1:N}\right), \mathbf{x}^m_{1:N})\}_{m=1}^{M}$.
2. Compute new estimates of the parameters $\theta^j_{i|h}$ for all i, h, and j by optimizing $J\left(\Theta\right)$ according to Eq. (12).

3.3 CGCL Algorithm

The objective function of the conditional log likelihood is given in Eq. (4). As in [1] we use a conjugate gradient algorithm with line-search which requires both the objective function and its derivative. In particular, the *Polak-Ribiere* method is used [22]. Similar as in Section 3.1, we distinguish two cases for deriving $\frac{\partial CLL(\mathcal{B}|\mathcal{S})}{\partial \theta^j_{i|h}}$. For TAN, NB, or 2-tree structures each parameter $\theta^j_{i|h}$ involves the class node value, either $C = i$ for $j = 1$ (Case A) or $C = h_1$ for $j > 1$ (Case B) where h_1 denotes the class instantiation $h_1 \in h$. For case A and B we have the derivatives

$$\frac{\partial CLL\left(\mathcal{B}|\mathcal{S}\right)}{\partial \theta^1_i} = \sum_{m=1}^{M} \left[\frac{u^{1,m}_i}{\theta^1_i} - \frac{W^m_i}{\theta^1_i} \right], \text{and} \tag{13}$$

$$\frac{\partial CLL\left(\mathcal{B}|\mathcal{S}\right)}{\partial \theta^j_{i|h}} = \sum_{m=1}^{M} \left[\frac{u^{j,m}_{i|h}}{\theta^j_{i|h}} - W^m_{h_1} \frac{v^{j,m}_{i|h\backslash h_1}}{\theta^j_{i|h}} \right], \tag{14}$$

respectively. The probability $\theta^j_{i|h}$ is constrained to $\theta^j_{i|h} \geq 0$ and $\sum_{i=1}^{|Z_j|} \theta^j_{i|h} = 1$. We re-parameterize the problem to incorporate the constraints of $\theta^j_{i|h}$ in the conjugate gradient algorithm. Thus, we use different parameters $\beta^j_{i|h} \in \mathbb{R}$ according to

$$\theta^j_{i|h} = \frac{\exp\left(\beta^j_{i|h}\right)}{\sum_{l=1}^{|Z_j|} \exp\left(\beta^j_{l|h}\right)}.$$

This requires the gradient $\frac{\partial CLL(\mathcal{B}|\mathcal{S})}{\partial \beta^j_{i|h}}$ which is computed using the chain rule as

$$\frac{\partial CLL\left(\mathcal{B}|\mathcal{S}\right)}{\partial \beta^j_{i|h}} = \sum_{k=1}^{|Z_j|} \frac{\partial CLL\left(\mathcal{B}|\mathcal{S}\right)}{\partial \theta^j_{k|h}} \frac{\partial \theta^j_{k|h}}{\partial \beta^j_{i|h}} = \sum_{m=1}^{M} \left[u_i^{1,m} - W_i^m\right] - \theta_i^1 \sum_{m=1}^{M} \sum_{c=1}^{|C|} \left[u_c^{1,m} - W_c^m\right]$$

for Case A and similarly for Case B we get the gradient

$$\frac{\partial CLL\left(\mathcal{B}|\mathcal{S}\right)}{\partial \beta^j_{i|h}} = \sum_{m=1}^{M} \left[u^{j,m}_{i|h} - W^m_{h_1} v^{j,m}_{i|h\setminus h_1}\right] - \theta^j_{i|h} \sum_{m=1}^{M} \sum_{l=1}^{|Z_j|} \left[u^{j,m}_{l|h} - W^m_{h_1} v^{j,m}_{l|h\setminus h_1}\right].$$

3.4 EBW Algorithm

The EBW algorithm uses the re-estimation equation [12, 13] of the form

$$\theta^j_{i|h} \leftarrow \frac{\theta^j_{i|h}\left(\frac{\partial CLL(\mathcal{B}|\mathcal{S})}{\partial \theta^j_{i|h}} + D\right)}{\sum\limits_{l}^{|Z_j|} \theta^j_{l|h}\left(\frac{\partial CLL(\mathcal{B}|\mathcal{S})}{\partial \theta^j_{l|h}} + D\right)}. \tag{15}$$

Considering the derivative in Eq. (14) (similar for Eq. (13)) we obtain

$$\theta^j_{i|h} \leftarrow \frac{\sum\limits_{m=1}^{M}\left[u^{j,m}_{i|h} - W^m_{h_1} v^{j,m}_{i|h\setminus h_1}\right] + \theta^j_{i|h} D}{\sum\limits_{m=1}^{M}\sum\limits_{i=1}^{|Z_j|}\left[u^{j,m}_{i|h} - \lambda W^m_{h_1} v^{j,m}_{i|h\setminus h_1}\right] + D}.$$

In fact, this equation is related to the parameter estimation equation of ECL using Laplace smoothing (see Eq. (10)), i.e. no λ is needed and the value of D is set globally.

The EBW algorithm converges to a local optimum of $CLL\left(\mathcal{B}|\mathcal{S}\right)$ providing a sufficiently large value for D. Indeed, 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]. The derivatives (Eq. (13) and (14)) are sensitive

to small parameter values. Therefore, we use a more robust approximation for the derivative of Case B (similarly for Case A) as suggested in [23, 13]

$$\frac{\partial CLL\left(\mathcal{B}|\mathcal{S}\right)}{\partial \theta^{j}_{i|h}} \approx \frac{\sum\limits_{m=1}^{M} u^{j,m}_{i|h}}{\sum\limits_{l=1}^{|Z_j|}\sum\limits_{m=1}^{M} u^{j,m}_{l|h}} - \frac{\sum\limits_{m=1}^{M} W^{m}_{h_1} v^{j,m}_{i|h\setminus h_1}}{\sum\limits_{l=1}^{|Z_j|}\sum\limits_{m=1}^{M} W^{m}_{h_1} v^{j,m}_{l|h\setminus h_1}}.$$

Setting D according to $D = 1 + \left|\min_{i,h,j} \frac{\partial CLL(\mathcal{B}|\mathcal{S})}{\partial \theta^{j}_{i|h}}\right|$ shows good performance in our experiments.

4 Experiments

We present results for frame- and segment-based phonetic classification using the TIMIT speech corpus [15] and for handwritten digit recognition using the MNIST [24] and the USPS data. In the following, we list the used structure learning algorithms for TAN and 2-trees [25]:

- TAN-CMI: Generative TAN structure learning using CMI [17].
- TAN-CR: Discriminative TAN structure learning using naive greedy heuristic [19].
- TAN-OMI-CR: Discriminative TAN structure learning using the efficient order-based heuristic [20].
- 2-tree-OMI-CR: Discriminative 2-tree structure learning using the order-based heuristic.

Once the structure has been determined discriminative parameter optimization is performed. In particular, we use the ECL and ACL parameter optimization (see Section 3.1 and 3.2). Additionally, EBW (see Section 3.4) and CGCL (see Section 3.3) parameter learning have been applied. The parameters are initialized to the ML estimates for all discriminative parameter learning methods (empirical results showed it performed better than random initialization). Similar as in [1] we use *cross tuning* to estimate the optimal number of iterations for CGCL. For ECL and ACL we perform 5 iterations — however, the best values are mostly found after the first iteration in case of ML parameter initialization. The λ for the best classification rate is empirically obtained in the range of 0.1 to 0.9. EBW uses 200 iterations. As mentioned earlier, this number depends on the choice of D. Continuous features were discretized using recursive minimal entropy partitioning [26] where the quantization intervals were determined using only the training data. Zero probabilities in the conditional probability tables are replaced with small values ($\varepsilon = 0.00001$). Further, we used the same data set partitioning for various learning algorithms.

4.1 Data characteristics

TIMIT-4/6 Data: This data set is extracted from the TIMIT speech corpus using the dialect speaking region 4 which consists of 320 utterances from 16 male

and 16 female speakers. Speech frames are classified into either four or six classes using 110134 and 121629 samples, respectively. Each sample is represented by 20 MFCC and wavelet-based features. We perform classification experiments on data of male speakers (Ma), female speakers (Fe), and both genders (Ma+Fe), all in all resulting in 6 distinct data sets (i.e., Ma, Fe, Ma+Fe $\times$ 4 and 6 classes). We use 70% of the data for training and the remaining 30% for testing. More details are given in [27].

TIMIT-39 Data: The difference to TIMIT-4/6 is as follows: The phonetic transcription boundaries specify a set of frames belonging to a particular phoneme. From this set of frames – the phonetic segment – a single feature vector is derived. In accordance with [28] 61 phonetic labels are combined into 39 classes, ignoring glottal stops. For training, 462 speakers from the standard NIST training set have been used. For testing the remaining 168 speakers from the overall 630 speakers were employed. We derive from each phonetic segment 66 features, i.e. MFCC's, Derivatives, and log duration. All together we have 140173 training samples and 50735 testing samples. Further information is given in [25].

MNIST Data: We present results for the handwritten digit MNIST data [24] which contains 60000 samples for training and 10000 digits for testing. We downsample the gray-level images by a factor of two which results in a resolution of 14×14 pixels, i.e. 196 features.

USPS Data: This data set contains 11000 uniformly distributed handwritten digit images from zip codes of mail envelopes. The data set is split into 8000 images for training and 3000 for testing. Each digit is represented as a 16×16 grayscale image, where each pixel is considered as feature.

4.2 Results

Tables 1, 2, 3, and 4 show the classification rates for TIMIT-39, MNIST, USPS, and the 6 TIMIT-4/6 data sets for various learning methods[2]. Additionally, we provide classification performances for TIMIT-4/6 employing support vector machines (SVMs) using a radial basis function (RBF) kernel[3].

Discriminative parameter learning using ECL, ACL, EBW and CGCL produces mostly a significantly better classification performance than ML parameter learning on the same classifier structure. Especially, for cases where the structure of the underlying model is not optimized for classification [1] — the average improvement of discriminative parameter learning over ML estimation on NB and generative TAN-CMI structures is large.

[2] The average CR over the 6 TIMIT-4/6 data sets is determined by weighting the CR of each data set with the number of samples in the test set. These values are accumulated and normalized by the total amount of samples in all test sets.

[3] The SVM 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 variance parameter for the RBF kernel. We set the values for these parameters to $C^* = 1$ and $\sigma = 0.05$. The optimal choice of the parameters and kernel function has been established during extensive experiments.

Table 1. Classification results in [%] for TIMIT-39 data with standard deviation. Best parameter learning results for each structure are emphasized using bold font.

	Parameter Learning				
Classifier	ML	ECL	ACL	EBW	CGCL
NB	61.70 ± 0.89	65.70 ± 0.87	65.33 ± 0.87	**70.35** ± 0.83	70.33 ± 0.83
TAN-CMI	65.40 ± 0.87	**66.33** ± 0.86	66.08 ± 0.86	65.38 ± 0.87	66.31 ± 0.86
TAN-OMI-CR	66.61 ± 0.86	66.94 ± 0.86	**67.41** ± 0.86	66.35 ± 0.86	66.87 ± 0.86
TAN-CR	66.78 ± 0.86	67.23 ± 0.86	**67.66** ± 0.85	66.73 ± 0.86	67.23 ± 0.86
2-tree-OMI-CR	66.94 ± 0.86	67.54 ± 0.85	**67.94** ± 0.85	66.86 ± 0.86	67.06 ± 0.86

Table 2. Classification results in [%] for MNIST data with standard deviation. Best parameter learning results for each structure are emphasized using bold font.

	Parameter Learning				
Classifier	ML	ECL	ACL	EBW	CGCL
NB	83.73 ± 0.37	87.75 ± 0.33	87.73 ± 0.33	91.65 ± 0.28	**91.70** ± 0.28
TAN-CMI	91.28 ± 0.28	92.74 ± 0.26	92.77 ± 0.26	93.21 ± 0.25	**93.80** ± 0.24
TAN-OMI-CR	92.01 ± 0.27	**93.62** ± 0.24	93.46 ± 0.25	**93.62** ± 0.24	93.39 ± 0.25
TAN-CR	92.58 ± 0.26	93.86 ± 0.24	93.69 ± 0.24	93.86 ± 0.24	**93.94** ± 0.24
2-tree-OMI-CR	92.69 ± 0.26	**93.11** ± 0.25	92.98 ± 0.26	92.93 ± 0.26	93.09 ± 0.25

Table 3. Classification results in [%] for USPS data with standard deviation. Best parameter learning results for each structure are emphasized using bold font.

	Parameter Learning				
Classifier	ML	ECL	ACL	EBW	CGCL
NB	87.10 ± 0.61	91.77 ± 0.50	91.83 ± 0.50	**94.03** ± 0.43	93.67 ± 0.44
TAN-CMI	91.90 ± 0.50	93.50 ± 0.45	93.60 ± 0.45	92.83 ± 0.47	**94.87** ± 0.40
TAN-OMI-CR	92.40 ± 0.48	94.27 ± 0.42	94.07 ± 0.43	93.73 ± 0.44	**94.90** ± 0.40
TAN-CR	92.57 ± 0.48	93.93 ± 0.44	94.13 ± 0.43	94.23 ± 0.43	**95.83** ± 0.36
2-tree-OMI-CR	94.03 ± 0.43	94.50 ± 0.42	94.60 ± 0.41	94.10 ± 0.43	**94.77** ± 0.41

In particular, for NB structures the convergent EBW and CGCL methods are superior compared to ECL and ACL. Analyzing the results of EBW and CGCL on TAN and 2-tree structures reveal that EBW may overfit the data and *cross tuning* in CGCL is too restrictive concerning the number of iterations. Hence, an alternative regularization method is required. This also explains the good performance of ECL and ACL on those structures even though both algorithms do not converge to a local optimum. *Cross tuning* in CGCL and the selection of D in EBW is time-consuming. Especially, the choice of D strongly influences the convergence rate of EBW. For the ECL and ACL methods we have to select $\lambda \in [0, \ldots, 1]$. The best classification rates are mostly obtained after only 1 iteration. This renders ECL and ACL to be computationally less demanding than EBW and CGCL. For the discriminative 2-tree, discriminatively learned parameters do not help to outperform ML estimation using TIMIT-4/6 (contrary on the remaining data). Again, we suspect overfitting effects since the performance of discriminative parameter learning still improves on the training data. However, the best classification performances for TIMIT-4/6 are achieved with SVMs. One reason might be that SVMs are applied to the continuous feature domain. In contrast to SVMs, a Bayesian network is a generative model. It might be preferred since it is easy to work with missing features, parameter tying and knowledge-based hierarchical decomposition is facilitated, and it is easy to work with structured data.

Table 4. Classification results in [%] for TIMIT-4/6 data with standard deviation. Best parameter learning results for each structure are emphasized using bold font.

Data set Number of Classes	Ma+Fe 4	Ma 4	Fe 4	Ma+Fe 6	Ma 6	Fe 6	
Classifier							Average
NB-ML	87.90 ± 0.18	88.69 ± 0.25	87.67 ± 0.25	81.82 ± 0.20	82.26 ± 0.28	81.93 ± 0.28	84.85
NB-ECL	91.36 ± 0.15	92.13 ± 0.21	90.84 ± 0.22	84.07 ± 0.19	84.75 ± 0.27	83.60 ± 0.27	87.59
NB-ACL	91.19 ± 0.16	91.81 ± 0.21	90.60 ± 0.23	83.67 ± 0.19	85.05 ± 0.26	83.48 ± 0.27	87.40
NB-EBW	91.61 ± 0.15	92.50 ± 0.21	91.15 ± 0.22	85.03 ± 0.19	86.01 ± 0.26	84.54 ± 0.27	88.27
NB-CGCL	**92.12** ± 0.15	**92.81** ± 0.20	**91.57** ± 0.22	**85.41** ± 0.18	**86.28** ± 0.26	**85.12** ± 0.26	**88.69**
TAN-CMI-ML	89.83 ± 0.17	90.20 ± 0.23	90.36 ± 0.23	82.23 ± 0.20	83.20 ± 0.28	82.99 ± 0.28	86.18
TAN-CMI-ECL	90.98 ± 0.16	91.32 ± 0.22	**91.00** ± 0.22	**83.98** ± 0.19	84.79 ± 0.27	83.71 ± 0.27	**87.42**
TAN-CMI-ACL	91.00 ± 0.16	91.36 ± 0.22	90.93 ± 0.22	83.70 ± 0.19	84.46 ± 0.27	83.52 ± 0.27	87.28
TAN-CMI-EBW	**91.35** ± 0.15	**92.01** ± 0.21	90.98 ± 0.22	83.39 ± 0.19	84.45 ± 0.27	83.38 ± 0.27	87.34
TAN-CMI-CGCL	90.96 ± 0.16	91.39 ± 0.22	90.92 ± 0.22	83.06 ± 0.20	**84.85** ± 0.27	**84.05** ± 0.27	87.22
TAN-OMI-CR-ML	91.19 ± 0.16	92.15 ± 0.21	90.51 ± 0.23	84.07 ± 0.19	84.68 ± 0.27	83.71 ± 0.27	87.52
TAN-OMI-CR-ECL	**91.72** ± 0.15	92.45 ± 0.21	**90.88** ± 0.22	**84.54** ± 0.19	85.01 ± 0.27	**84.26** ± 0.27	**87.96**
TAN-OMI-CR-ACL	91.67 ± 0.15	**92.49** ± 0.21	90.77 ± 0.22	84.37 ± 0.19	**85.09** ± 0.26	84.08 ± 0.27	87.88
TAN-OMI-CR-EBW	91.17 ± 0.16	92.48 ± 0.21	90.79 ± 0.22	83.07 ± 0.20	84.15 ± 0.27	83.33 ± 0.28	87.20
TAN-OMI-CR-CGCL	91.37 ± 0.15	92.28 ± 0.21	90.51 ± 0.23	84.00 ± 0.19	84.49 ± 0.27	83.75 ± 0.27	87.54
TAN-CR-ML	91.29 ± 0.16	91.81 ± 0.21	90.52 ± 0.23	84.35 ± 0.19	84.80 ± 0.27	83.93 ± 0.27	87.62
TAN-CR-ECL	**91.69** ± 0.15	92.17 ± 0.21	90.98 ± 0.22	**84.63** ± 0.19	**85.26** ± 0.26	**84.14** ± 0.27	**87.97**
TAN-CR-ACL	91.52 ± 0.15	92.07 ± 0.21	90.14 ± 0.23	84.59 ± 0.19	85.01 ± 0.27	83.84 ± 0.27	87.74
TAN-CR-EBW	91.03 ± 0.16	**92.32** ± 0.21	**91.07** ± 0.22	83.33 ± 0.20	84.73 ± 0.27	82.90 ± 0.28	87.27
TAN-CR-CGCL	91.29 ± 0.16	92.04 ± 0.21	90.52 ± 0.23	83.69 ± 0.19	84.83 ± 0.27	83.91 ± 0.27	87.48
2-Tree-OMI-CR-ML	**91.68** ± 0.15	**92.28** ± 0.21	91.03 ± 0.22	**84.52** ± 0.19	**85.43** ± 0.26	**84.31** ± 0.27	**88.01**
2-Tree-OMI-CR-ECL	91.53 ± 0.15	92.18 ± 0.21	90.98 ± 0.22	84.13 ± 0.19	85.00 ± 0.27	83.81 ± 0.27	87.73
2-Tree-OMI-CR-ACL	91.57 ± 0.15	92.06 ± 0.21	90.95 ± 0.22	84.34 ± 0.19	85.16 ± 0.26	84.07 ± 0.27	87.83
2-Tree-OMI-CR-EBW	91.63 ± 0.15	**92.28** ± 0.21	**91.06** ± 0.22	84.26 ± 0.19	85.22 ± 0.26	84.13 ± 0.27	87.88
2-Tree-OMI-CR-CGCL	91.28 ± 0.16	91.79 ± 0.21	90.53 ± 0.23	83.46 ± 0.19	84.48 ± 0.27	83.42 ± 0.27	87.27
SVM-1-5	**92.49** ± 0.14	**93.30** ± 0.20	**92.14** ± 0.21	**86.24** ± 0.18	**87.19** ± 0.25	**86.19** ± 0.25	**89.38**

Figure 1 shows the classification rate of ECL and ACL parameter learning depending on λ for TIMIT-39 on both the training and test data. A NB structure is used. The best result on the training data was obtained for $\lambda = 0.4$, whereby the corresponding CR on the test set is 65.70% and 65.33% for ECL and ACL, respectively. For large values of λ the CR drops. For ECL we use Laplace-like

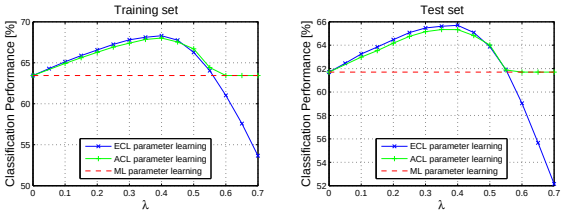

Fig. 1. Classification rate of TIMIT-39 versus λ using a NB structure.

smoothing for cases where the numerator/denominator in Eq. (8) and (9) is negative, whereas for ACL we use the ML fallback. In general, there is no clear winner between ECL and ACL parameter optimization. To prevent negative parameter values during ECL and ACL optimization Laplace-like smoothing and a ML fallback scheme have been proposed. Since both strategies work best on different cases we present the best classification rate from either strategy.

5 Conclusion

We present three *EM-like* discriminative parameter learning algorithms for Bayesian network classifiers. As first method, we introduce the extended Baum-Welch algorithm. The two remaining approaches are based on iteratively optimizing either the CL or a lower-bound surrogate of the conditional likelihood. Both algorithms do not show a monotonously increasing objective function unlike the extended Baum-Welch approach. Experiments on various phonetic and handwritten digit classification tasks show that for NB and generatively and discriminatively learned TAN structures discriminative parameter optimization algorithms lead to significant improvements compared to the generative ML parameter estimation. In general, the benefit of discriminative parameter training is large for simple network structures which are not optimized for classification.

Acknowledgments

The authors thank the anonymous reviewers for many useful comments. This work was supported by the Austrian Science Fund (Grant number P19737-N15 and S10604-N13). Thanks to Jeff Bilmes for discussions and support in writing this paper.

References

1. Greiner, R., Su, X., Shen, S., Zhou, W.: Structural extension to logistic regression: Discriminative parameter learning of belief net classifiers. Machine Learning **59** (2005) 297–322
2. 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. (1986) 49–52
3. Lasserre, J.: Hybrid of generative and discriminative methods for machine learning. PhD thesis, University of Cambridge (2008)
4. Jebara, T.: Discriminative, generative and imitative learning. PhD thesis, Media Laboratory, MIT (2001)
5. Vapnik, V.: Statistical learning theory. Wiley & Sons (1998)
6. Bishop, C.: Pattern recognition and machine learning. Springer (2006)
7. Salojärvi, J., Puolamäki, K., Kaski, S.: Expectation maximization algorithms for conditional likelihoods. In: Inter. Conf. on Machine Learning (ICML). (2005) 753 – 760

8. Roos, T., Wettig, H., Grünwald, P., Myllymäki, P., Tirri, H.: On discriminative Bayesian network classifiers and logistic regression. Machine Learning **59** (2005) 267–296
9. Ng, A., Jordan, M.: On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In: NIPS 14. (2002)
10. Pernkopf, F., Bilmes, J.: Discriminative versus generative parameter and structure learning of Bayesian network classifiers. In: Inter. Conf. on Machine Learning (ICML). (2005) 657 – 664
11. Ben-Yishai, A., Burshtein, D.: A discriminative training algorithm for hidden Markov models. IEEE Trans. on Speech and Audio Proc. **12**(3) (2004) 204–217
12. 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) (1991) 107–113
13. Woodland, P., Povey, D.: Large scale discriminative training of hidden Markov models for speech recognition. Computer Speech and Language **16** (2002) 25–47
14. Klautau, A., Jevtić, N., Orlitsky, A.: Discriminative Gaussian mixture models: A comparison with kernel classifiers. In: Inter. Conf. on Machine Learning (ICML). (2003) 353 – 360
15. 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)
16. Pearl, J.: Probabilistic reasoning in intelligent systems: Networks of plausible inference. Morgan Kaufmann (1988)
17. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning **29** (1997) 131–163
18. Cover, T., Thomas, J.: Elements of information theory. John Wiley & Sons (1991)
19. Keogh, E., Pazzani, M.: Learning augmented Bayesian classifiers: A comparison of distribution-based and classification-based approaches. In: Workshop on Artificial Intelligence and Statistics. (1999) 225–230
20. Pernkopf, F., Bilmes, J.: Order-based discriminative structure learning for Bayesian network classifiers. In: International Symposium on Artificial Intelligence and Mathematics. (2008)
21. Bouchard, G., Triggs, B.: The trade-off between generative and discriminative classifiers. In: Intern. Conf. on Computational Statistics. (2004) 721–728
22. Bishop, C.: Neural networks for pattern recognition. Oxford University Press (1995)
23. Merialdo, B.: Phonetic recognition using hidden Markov models and maximum mutual information training. In: IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP). (1988) 111–114
24. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings fo the IEEE **86**(11) (1998) 2278–2324
25. Pernkopf, F., Bilmes, J.: Efficient heuristics for discriminative structure learning of Bayesian network classifiers. Technical report, Laboratory of Signal Processing and Speech Communication, Graz University of Technology (2009)
26. Fayyad, U., Irani, K.: Multi-interval discretizaton of continuous-valued attributes for classification learning. In: Joint Conf. on Artificial Intelligence. (1993) 1022–1027
27. Pernkopf, F., Van Pham, T., Bilmes, J.: Broad phonetic classification using discriminative Bayesian networks. Speech Communication **143**(1) (2008) 123–138
28. Halberstadt, A., Glass, J.: Heterogeneous measurements for phonetic classification. In: Proceedings of EUROSPEECH. (1997) 401–404