Stochastic Margin-based Structure Learning of Bayesian Network Classifiers

Franz Pernkopf and Michael Wohlmayr[1]

Laboratory of Signal Processing and Speech Communication,
Graz University of Technology, Austria
E-mail: pernkopf@tugraz.at, michael.wohlmayr@tugraz.at

Abstract

The margin criterion for parameter learning in graphical models gained significant impact over the last years. We introduce the maximum margin score for discriminatively optimizing the structure of Bayesian network classifiers. Furthermore, greedy hill-climbing and simulated annealing search heuristics are applied to determine the classifier structures. In the experiments, we demonstrate the advantages of maximum margin optimized Bayesian network structures in terms of classification performance compared to traditionally used discriminative structure learning methods. Stochastic simulated annealing requires less score evaluations than greedy heuristics. Additionally, we compare generative and discriminative parameter learning on both generatively and discriminatively structured Bayesian network classifiers. Margin-optimized Bayesian network classifiers achieve similar classification performance as support vector machines. Moreover, missing feature values during classification can be handled by discriminatively optimized Bayesian network classifiers, a case where purely discriminative classifiers usually require mechanisms to complete unknown feature values in the data first.

Keywords: Bayesian network classifier, discriminative learning, maximum margin learning, structure learning

1. Introduction

Generative probabilistic classifiers optimize the joint probability distribution of the features $\mathbf{X}$ and the corresponding class labels C using maximum likelihood (ML) estimation. The class label is usually predicted using the

maximum a-posteriori estimate of the class posteriors $P(C|\mathbf{X})$ obtained by applying Bayes rule. Discriminative probabilistic classifiers such as logistic regression model $P(C|\mathbf{X})$ directly. Discriminative classifiers may lead to better classification performance, particularly when the class conditional distributions poorly approximate the true distribution [1]. Furthermore, Ng and Jordan [2] observed that discriminative models have mostly better asymptotic classification performance while generative classifiers approach its asymptotic performance faster requiring less training samples.

Basically, in Bayesian network classifiers both parameters and structure can be learned either generatively or discriminatively [3]. Discriminative learning requires objective functions such as classification rate (CR), conditional likelihood (CL), or margin (as we propose in this paper), that optimize the model for a particular inference scenario, e.g. for a classification task. We are particularly interested in learning the discriminative structure[2] of a generative Bayesian network classifier that factorize as $P(C, \mathbf{X}) = P(\mathbf{X}|C)P(C)$.

Learning the graph structure of a Bayesian network classifier is hard. Optimally learning various forms of constrained Bayesian network structures is NP-complete [4] even in the "generative" sense. It has been shown that learning paths [5], polytrees [6], k-trees [7] or bounded tree-width graphs [8, 9], and general Bayesian networks [10] are all instances of NP-complete optimization problems. Recently, approaches for finding the *optimal* generative Bayesian network structure have been proposed. These methods are based on dynamic programming [11], branch-and-bound techniques [12], or search over various variable orderings [13]. More methods and a comprehensive overview can be found in [14] and references therein. Discriminative structure learning is not less difficult because of the non-decomposability[3] of the scores. Discriminative structure learning methods – relevant for learning Bayesian network classifiers – are usually approximate methods based on local search heuristics. In [15], a greedy hill-climbing heuristic is used to learn a classifier structure using the CR score. Particularly, at each iteration one edge is added to the structure which complies with the restrictions of the network topology and the acyclicity constraints of a Bayesian network. In a similar algorithm, the

[2]Discriminative scoring functions (e.g. classification rate, conditional likelihood, or margin) are used for structure learning.

[3]Unfortunately, discriminative scores are usually not decomposable, while generative scores, e.g. log likelihood, are decomposeable, i.e. they can be expressed as a sum of terms where each term depends on a variable and its conditioning variables (parents).

CL has been applied for discriminative structure learning [16]. Recently, we introduced a computationally efficient order-based greedy search heuristic for finding discriminative structures [3]. Our order-based structure learning is based on the observations in [17] and shows similarities to the K2 heuristic [18]. However, we proposed to use a discriminative scoring metric (i.e. CR) and suggest approaches for establishing the variable ordering based on conditional mutual information [19].

One of the most successful discriminative classifiers, namely the support vector machine (SVM), finds a decision boundary which maximizes the margin between samples of distinct classes resulting in good generalization properties [20] of the classifier. Recently, the margin criterion has been applied to learn the parameters of probabilistic models. Taskar et al. [21] observed that undirected graphical models can be efficiently trained to maximize the margin. More recently, Guo et al. [22] introduced the maximization of the margin for parameter learning based on convex relaxation to Bayesian networks. We proposed to use a conjugate gradient algorithm for maximum margin optimization of the parameters and show its advantages with respect to computational requirements [23]. Further generative and discriminative *parameter* learning methods for Bayesian network classifiers are summarized in [3, 23] and references therein.

In this paper, we introduce the maximum margin (MM) criterion for discriminative *structure learning*. We use greedy hill-climbing (HC) and stochastic search heuristics such as simulated annealing (SA) [24, 25, 26] for learning discriminative classifier structures. SA is less prone to get stuck in local optima. We empirically evaluate our margin-based discriminative structure learning heuristics on two handwritten digit recognition tasks, one spam e-mail, and one remote sensing data set. We use naive Bayes (NB) as well as generatively and discriminatively optimized tree augmented naive Bayes (TAN) [27] structures. Furthermore, we experimentally compare both discriminative and generative parameter learning on both discriminative and generatively structured Bayesian network classifiers. Maximum margin *structure* learning outperforms recent generative and discriminative structure learning results. SA heuristics mostly lead to better performing structures at a lower number of score evaluations (CR or MM) compared to HC methods. Discriminative parameter learning produces a significantly better classification performance than ML parameter learning on the same classifier structure. This is especially valid for cases where the structure of the underlying model is not optimized for classification [28].

The paper is organized as follows: In Section 2, we introduce our notation, ML parameter learning as well as NB and TAN structures. In Section 3, we introduce the non-decomposeable discriminative scores CL, CR, and MM. Additionally, we discuss techniques for making the determination of these discriminative scores computationally competitive. Section 4 introduces different structure learning heuristics. Particular focus is on SA which is rarely used for discriminative learning of Bayesian network structures. In Section 5, we present experimental results. Section 6 concludes the paper.

2. Bayesian network classifiers

A Bayesian network [29] $\mathcal{B} = \langle \mathcal{G}, \Theta \rangle$ 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 $\mathbf{E} = \left\{E_{Z_i,Z_j}, E_{Z_i,Z_k}, \ldots\right\}$ connecting the nodes where E_{Z_i,Z_j} is an edge directed from Z_i to Z_j. This graph represents factorization properties of the distribution of a set of random variables $\mathbf{Z} = \{Z_1, \ldots, Z_{N+1}\}$. The variables in $\mathbf{Z}$ have values denoted by lower case letters $\mathbf{z} = \{z_1, z_2, \ldots, z_{N+1}\}$. 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 $C \in \{1, \ldots, |C|\}$, where $|C|$ represents the number of classes and $\mathbf{X}_{1:N} = \{X_1, \ldots, X_N\} = \{Z_2, \ldots, Z_{N+1}\}$ denotes the set of random variables representing the N attributes of the classifier. In a Bayesian network each node is independent of its non-descendants given its parents. The set of parameters which quantify the network are represented by Θ. Each random variable 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} assignment, i.e. $\theta^j_{i|h} = P_\Theta\left(Z_j = i | Z_{\Pi_j} = h\right)$. The training data consists of M independent and identically distributed samples $\mathcal{S} = \{\mathbf{z}^m\}_{m=1}^M = \{(c^m, \mathbf{x}^m_{1:N})\}_{m=1}^M$ where $M = |\mathcal{S}|$. The joint probability distribution of a sample $\mathbf{z}^m$ is determined as

$$P_\Theta\left(\mathbf{Z} = \mathbf{z}^m\right) = \prod_{j=1}^{N+1} P_\Theta\left(Z_j = z_j^m | Z_{\Pi_j} = z^m_{\Pi_j}\right). \tag{1}$$

The class labels are predicted using the maximum a-posteriori (MAP) estimate obtained by Bayes rule, i.e.

$$P_\Theta\left(C=c|\mathbf{X}_{1:N}=\mathbf{x}_{1:N}^m\right)=\frac{P_\Theta\left(C=c,\mathbf{X}_{1:N}=\mathbf{x}_{1:N}^m\right)}{\sum_{c'=1}^{|C|}P_\Theta\left(C=c',\mathbf{X}_{1:N}=\mathbf{x}_{1:N}^m\right)},$$

where the most likely class c^* is determined as

$$c^*=\underset{c\in\{1,\ldots,|C|\}}{\operatorname{argmax}}\,P_\Theta\left(C=c|\mathbf{X}_{1:N}=\mathbf{x}_{1:N}^m\right)=\underset{c\in\{1,\ldots,|C|\}}{\operatorname{argmax}}\,P_\Theta\left(C=c,\mathbf{X}_{1:N}=\mathbf{x}_{1:N}^m\right),$$

where the last term follows from neglecting the normalization factor.

The NB network assumes that all the attributes are conditionally independent given the class label. As reported in [27], 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. Friedman et al. [27] introduced the TAN classifier which is based on structural augmentations of the NB network. In order to relax the conditional independence properties of NB, each attribute may have at most one other attribute as an additional parent. This 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. A TAN classifier example is shown in Figure 1. In [3], we noticed that k-trees over the features – k indicates the tree-width[4] – often do not improve classification performance significantly without regularization. Therefore, we limit the experiments to NB and TAN structures.

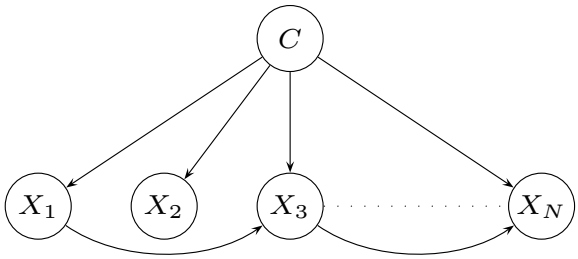

Figure 1: An example of a TAN classifier structure.

[4]The tree-width of a graph is defined as the size of the largest clique (i.e. number of variables within the largest clique) of the moralized and triangulated directed graph minus one. Since there can be multiple triangulated graphs, the tree-width is defined by the triangulation where the largest clique contains the fewest number of variables. More details are given in [30] and references therein.

3. Discriminative Scores for Structure Learning

In this section, we first summarize traditionally used discriminative scores such as CR and CL. Then, we introduce our maximum margin (MM) score. Finally, we provide some techniques to make the computation of discriminative scores computationally competitive. For the sake of brevity, we only notate instantiations of the random variables in the following.

3.1. Traditional Scores: CR and CL

There are two score functions we consider: the CR [15, 31, 3]

$$CR\left(\mathcal{B}|\mathcal{S}\right) = \frac{1}{M}\sum_{m=1}^{M} \mathbf{1}_{\left\{c^m = \arg\max_{c'} P_\Theta\left(c', \mathbf{x}^m_{1:N}\right)\right\}}$$

and the CL [16]

$$CL\left(\mathcal{B}|\mathcal{S}\right) = \sum_{m=1}^{M} \log P_\Theta\left(c^m|\mathbf{x}^m_{1:N}\right).$$

Symbol $\mathbf{1}_{\{i=j\}}$ denotes the indicator function (i.e. equals 1 if the Boolean expression $i = j$ is true and 0 otherwise). CR is tightly connected to CL under the 0/1-loss function (other smooth convex upper-bound surrogates might be employed [32]). While either the CL or the CR can be used as score for structure learning, we in this work restrict our experiments to CR (empirical results show it performs better).

3.2. Maximum Margin Score

The multi-class margin [22] of sample m can be expressed as

$$d^m_\Theta = \min_{c \neq c^m} \frac{P_\Theta\left(c^m|\mathbf{x}^m_{1:N}\right)}{P_\Theta\left(c|\mathbf{x}^m_{1:N}\right)} = \frac{P_\Theta\left(c^m, \mathbf{x}^m_{1:N}\right)}{\max_{c \neq c^m} P_\Theta\left(c, \mathbf{x}^m_{1:N}\right)}.$$

If $d^m_\Theta > 1$, then sample m is correctly classified and vice versa. The magnitude of d^m_Θ is related to the confidence of the classifier about its decision. Taking the logarithm, we obtain

$$\log d^m_\Theta = \log P_\Theta\left(c^m, \mathbf{x}^m_{1:N}\right) - \max_{c \neq c^m}\left(\log P_\Theta\left(c, \mathbf{x}^m_{1:N}\right)\right).$$

Usually, the maximum margin approach maximizes the margin of the sample with the smallest margin for a separable classification problem [33], i.e. the

objective function is written as $M(\mathcal{B}|\mathcal{S}) = \min_{m=1,\ldots,M} \log d_\Theta^m$. For non-separable problems, we aim to relax this by introducing a soft margin, i.e. we focus on samples with $\log d_\Theta^m$ close to zero. For this purpose, we consider the *hinge* loss function

$$M(\mathcal{B}|\mathcal{S}) = \sum_{m=1}^{M} \min\left[1, \lambda \log d_\Theta^m\right],$$

where the scaling parameter $\lambda > 0$ controls the margin with respect to the loss function and is set by cross-validation. Maximizing this function with respect to the parameters Θ implicitly increases the log-margin, whereas the emphasis is on samples with $\lambda \log d_\Theta^m < 1$, i.e. samples with a large positive margin are considered as constant factor during on the optimization. We use $M(\mathcal{B}|\mathcal{S})$ as score for discriminative structure learning.

Note that the CR and MM scores are determined from a classifier trained and tested on different data $\mathcal{S}$. For the sake of simplicity we do not denote this explicitly.

3.3. Computational Considerations

The CR computation can be accelerated by the following techniques:

1. The data samples are reordered during structure learning so that misclassified samples from previous evaluations are classified first. The classification is terminated as soon as the performance drops below the currently best network score [34].
2. During structure learning the parameters are set to the ML values. When learning the structure we only have to update the parameters of those nodes where the set of parents Z_{Π_j} changes. This observation can be also used for computing the joint probability during classification. We can cache the joint probability and exchange only the probabilities of those nodes where the set of parents changed to get the new joint probability [15].
3. Furthermore, since we are restricted to 1-trees, each attribute can have only one of the other attributes as parent. This means that we can compute the $\mathcal{O}(N^2)$ ML parameter estimates beforehand. This prevents redundant parameter learning at later stages of the search. In fact, after the selection of the first edge during HC search, we have already determined all $\mathcal{O}(N^2)$ ML estimates (see also Section 4.2).

4. In case of large memory, a multi-way table $\mathcal{T}$ of order four $M \times N \times N \times |C|$ can be assembled at the beginning which enables to determine the m^{th} joint probability $P_\Theta\left(C = c, \mathbf{X}_{1:N} = \mathbf{x}_{1:N}^m\right)$ for class c of *any* structure. One of the $\mathcal{O}\left(N^2\right)$ ML parameter estimates $\log \theta_{i|h}^j$ for each variable X_j $(j \in 1, \ldots, N)$ and each conditioning possible attribute X_{Π_j} $(\Pi_j \in 1, \ldots, N)$ given the values of sample m and class c constitutes the element at index $\langle m, j, \Pi_j, c\rangle$ of $\mathcal{T}$. Hence, the log joint probability for class c and sample m can be obtained by summing over all entries X_j in $\mathcal{T}$ using the appropriate conditioning parent X_{Π_j} given by the structure. This multi-way table enables to quickly compute the CR for all possible TAN structures.

Observations 2, 3, and 4 equivalently apply for computing the margin score since it is also based on efficiently computing the joint probability $P_\Theta\left(c, \mathbf{x}_{1:N}^m\right)$. Considering all these techniques leads to a tremendous computational speedup.

4. Structure Learning Heuristics

This section provides three structure learning heuristics. Note that the parameters during structure learning are optimized generatively using maximum likelihood estimation [29].

4.1. Generative Structure Learning

The conditional mutual information (CMI) [19] between the attributes given the class variable is determined as:

$$I\left(X_i; X_j | C\right) = E_{P(X_i, X_j, C)}\left[\log \frac{P\left(X_i, X_j | C\right)}{P\left(X_i | C\right) P\left(X_j | C\right)}\right],$$

where $E\left[\cdot\right]$ denotes the expectation. This measures the information between X_i and X_j in the context of C. Friedman et al. [27] gives an algorithm for constructing a TAN network using this measure.

First, the pairwise CMI $I\left(X_i; X_j | C\right) \quad \forall \quad 1 \leq i \leq N$ and $i < j \leq N$ is computed. Then, an undirected 1-tree is built using the maximal weighted spanning tree algorithm [29] where each edge connecting X_i and X_j is weighted by $I\left(X_i; X_j | C\right)$. The undirected 1-tree is transformed to a directed tree. Therefore, a root variable is selected and all edges are directed away from this root. Finally, the class node C and the edges from C to all attributes $X_1, \ldots, X_N$ are added.

4.2. Discriminative Structure Learning: Greedy Hill-Climbing (HC)

A Bayesian network is initialized to NB and at each iteration we add the edge that, while maintaining a partial 1-tree, gives the largest improvement of the scoring function. Basically, all discriminative scoring functions can be considered, i.e. CR, CL, and MM. Structure learning is performed until we obtain a 1-tree over the attributes, i.e. we add $N-1$ edges. This approach is computationally expensive since each time an edge is added, the scores for $\mathcal{O}(N^2)$ edges of all attributes without conditioning attribute complying with the acyclicity requirements need to be re-evaluated due to the discriminative non-decomposable scoring functions we employ. Overall, for learning a 1-tree structure, $\mathcal{O}(N^3)$ score evaluations are necessary. In our experiments, we consider either the CR or the margin as objective.

4.3. Discriminative Structure Learning: Simulated Annealing (SA)

The main advantage of SA compared to HC is that this heuristic is capable to escape from local optima, although, SA is not guaranteed to find global optimal solutions. In the context of discriminative structure learning, we empirically show in Section 5 that SA is beneficial in terms of finding well performing structures at low computational costs. SA for learning discriminative TAN structures is summarized in Algorithm 1. The basic principle of SA is that there is an additional temperature parameter T which enables to accept worse solutions than the currently best one with a certain probability. Parameter T is decreased periodically after either the maximum number of trials with one temperature ($MaxTry$) or the maximum number of successful network changes ($MaxSuccess$) is reached. Therefore, a *cooling schedule* is introduced which makes the acceptance of lower scoring solutions less likely at later stages of the search, i.e. towards the end of the optimization, the values of T are small and SA accepts almost only score improvements similar as greedy hill-climbing. For learning MM structures, the CR score is replaced by $M(\mathcal{B}|\mathcal{S})$.

```
Input: X_{1:N}, C, S;
Output: Set of edges E for TAN network;
Initialization: T ← 1, i ← 0, RejectCounter ← 0, SuccessCounter ← 0;
CR_old ← CR(B|S) for NB classifier;
E_old ← ∅;
while RejectCounter < MaxConsecutiveRejection do
    while (SuccessCounter < MaxSuccess)&(i < MaxTry) do
        i ← i + 1;
        E ← GenerateNeighboringTree(E_old);
        CR ← CR(B|S) Evaluate current network;
        if CR − CR_old > 0 then
            RejectCounter ← 0;
            SuccessCounter ← SuccessCounter + 1;
            CR_old ← CR;
            E_old ← E;
        else
            if random[0, 1) < exp((CR−CR_old)/(rT)) then
                SuccessCounter ← SuccessCounter + 1;
                CR_old ← CR;
                E_old ← E;
            else
                RejectCounter ← RejectCounter + 1;
            end
        end
    end
    T ← DT;
    i ← 0;
    SuccessCounter ← 0;
end
```

Algorithm 1: Simulated Annealing for discriminative structure learning.

We initialize the network to NB. The constant D implements the annealing schedule and r denotes a constant[5]. Various values of D have been tested in the experiments. The values for $MaxConsecutiveRejection$, $MaxTry$, and $MaxSuccess$ depend on the number of features and are set to $100N$, $50N$, and $5N$, respectively. The function $GenerateNeighboringTree$ generates a new neighboring tree by randomly changing one edge in the 1-tree. First, one variable X_j is selected randomly. For X_j we randomly select a new

[5]For the CR and the MM score r is set to 1 and 100, respectively.

parent variable X_{Π_j}. The edge $E_{X_{\Pi_j},X_j}$ is added/replaced if the acyclicity constraints are not violated.

5. Experiments

We present results for two handwritten digit recognition tasks, spam e-mail classification, and for a remote sensing application. In the following, we provide details about the data sets:
MNIST Data: The MNIST data [35] contains 60000 samples for training and 10000 handwritten digits for testing. We down-sample 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.
Spambase Data: This data [36] considers the classification of e-mails into either spam or not-spam. Most of the 57 attributes indicate whether a particular word or character is frequently occurring in the e-mail. The data set contains 2301 and 2300 samples for training and testing, respectively.
DCMall Data: 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[6]. As ground reference a classification performed at Purdue University was used containing 7 classes, namely roof, road, grass, trees, trail, water, and shadow[7]. The image contains 1280×307 hyperspectral pixels, i.e. 392960 samples. We arbitrarily choose 5000 samples of each class to learn the classifier.

For structure learning we use the algorithms introduced in Section 4. We empirically compare deterministic greedy HC versus stochastic SA using either CR or the MM score for discriminative structure learning. In particular, we apply the following approaches for learning TAN structures:

- TAN-ST-CMI: Generative TAN structure learning using the spanning tree (ST) algorithm and the CMI score [27].

[6]http://cobweb.ecn.purdue.edu/~biehl/MultiSpec/hyperspectral.html
[7]http://cobweb.ecn.purdue.edu/~landgreb/Hyperspectral.Ex.html

- TAN-HC-CR: Discriminative greedy hill-climbing learning of TAN structures maximizing the CR score [15, 3].
- TAN-HC-MM: Discriminative greedy hill-climbing learning of TAN structures maximizing the MM score (introduced in this paper).
- TAN-SA-CR: Discriminative stochastic TAN structure learning based on simulated annealing maximizing the CR (introduced in this paper).
- TAN-SA-MM: Discriminative stochastic TAN structure learning based on simulated annealing maximizing the MM (introduced in this paper).

The CR and MM scores are determined by 5-fold cross-validation on the training data. Zero probabilities in the conditional probability tables are replaced with small values ε. Various learning algorithms use the same data set partitioning. We use ML parameter learning during discriminative structure learning. We recently developed a MM parameter learning method for Bayesian network classifiers [23]. We are interested in the case if MM parameter learning is further improving the classification performance once the structure has been discriminatively optimized, i.e. we empirically compare both discriminative and generative parameter learning on both discriminative and generatively structured Bayesian network classifiers. Both parameter learning methods are abbreviated as ML and MM.

Since there is an abundance of combinations of algorithms a simple naming scheme is introduced: We use a 2-, or 4-tag scheme A-B-C-D where "A" (if given) refers to either NB or TAN (1-tree), "B" and "C" (if given) refer to the structure learning heuristic and score, respectively, and "D" (if given) refers to the parameter training method of the *final* resultant model structure, i.e. either ML or MM.

5.1. Results – Discriminative Structure Learning

Table 1 shows the classification performance of all combinations of structure learning methods using ML parameter learning.

The best discriminatively optimized structures significantly outperform generatively learned, i.e. TAN-CMI, and NB structures. For Spambase TAN-HC-CR-ML is performing worse compared to generative structure learning. One reason might lie in the small data set size since generative classifiers approach its asymptotic performance faster requiring less training samples while discriminative models have mostly better asymptotic classification performance [2]. Future work includes investigation of sample size effects.

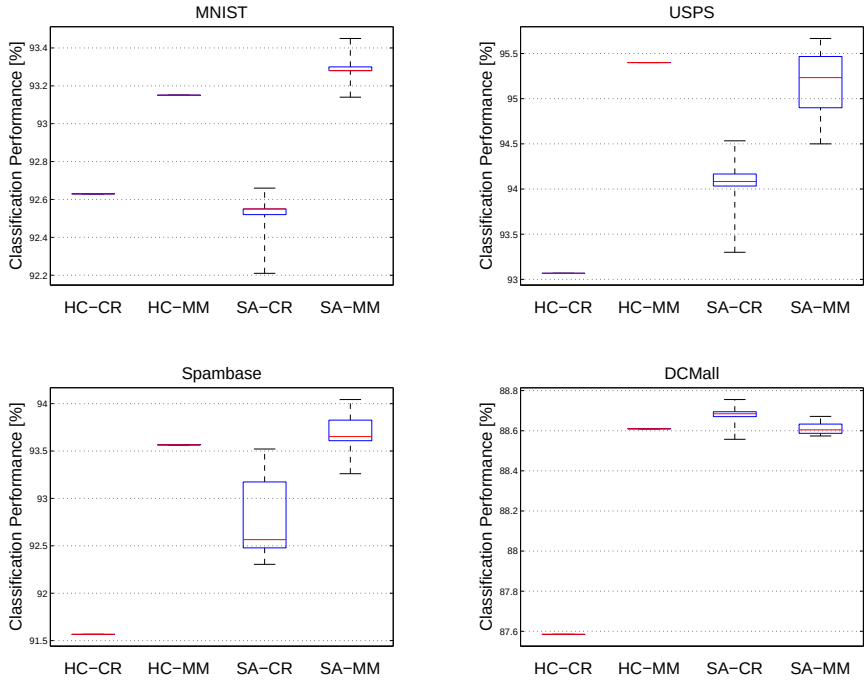

Figure 2: Classification performance of discriminatively structured Bayesian network classifier using either the CR or MM score in the HC and SA heuristics for structure learning. Horizontal red lines of the box plot corresponds to the median over 10 runs, the boxes represent the 25% and 75% quantiles, and the whiskers visualize the total range.

Comparing the classification rates achieved with discriminative scores in Figure 2 (also Table 1) we observe that the MM score is leading to better or equal performing networks for all data sets. Since SA is a stochastic optimization method we perform 10 independent structure learning runs for each data set. While Table 1 reports the result of the best run, Figure 2 uses a box plot (median, 25% and 75% quantiles, and total range (whiskers)) to summarize all runs. SA achieves mostly a better classification performance as greedy HC search using the same scoring measure. SA is able to escape from local optimal solutions due to the cooling schedule (see Section 4.3) while HC has no mechanism for accepting lower-performing structures, i.e. SA has a more flexible treatment of local optima. However, also SA does not guarantee to find global optima. Furthermore, in Figure 3 we compare the number of score evaluations used to learn the discriminative structures for each of the

Table 1: Classification results in [%] with standard deviation using ML parameter learning. Best structure learning results are emphasized using bold font.

Classifier Structure	MNIST	USPS	Spambase	DCMall
NB-ML	83.73±0.37	87.10±0.61	89.87±0.63	81.07±0.06
TAN-ST-CMI-ML	91.28±0.28	91.90 ±0.50	92.86±0.54	85.63±0.06
TAN-HC-CR-ML	92.63±0.26	93.07±0.46	92.08±0.56	87.58±0.06
TAN-HC-MM-ML	**93.15**±0.25	**95.40**±0.38	**93.43**±0.52	**88.61**±0.05
TAN-SA-CR-ML	92.66±0.26	94.07±0.43	**93.52**±0.51	**88.69**±0.05
TAN-SA-MM-ML	**93.30**±0.25	**95.46**±0.38	**93.26**±0.52	**88.67**±0.05

data sets. Fortunately, SA requires roughly one order of magnitude less score evaluations for data sets with a large number of features N, i.e. for MNIST, USPS, and DCMall. While all the SA results have been determined for $D = 0.8$, Figure 4 shows the influence of the annealing schedule $T \leftarrow DT$ of Algorithm 1. In particular, we compare the classification performance over the number of evaluations for $D \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$ using the USPS data. Ten SA structure optimizations have been performed for each value of D. These curves reveal that a faster schedule (i.e. lower value for D) might even reduce the number of score evaluations at similar classification performance.

5.2. Results – Discriminative Parameter Learning

Table 2 shows the classification performance of all combinations of structure learning methods using MM parameter learning[8].

Table 2: Classification results in [%] with standard deviation using MM parameter learning. Best MM parameter learning results are emphasized using bold font.

Classifier Structure	MNIST	USPS	Spambase	DCMall
NB-MM	91.73±0.28	94.80±0.41	**94.39**±0.48	89.23±0.05
TAN-ST-CMI-MM	94.13±0.24	95.50 ±0.38	93.91±0.50	89.22±0.05
TAN-HC-CR-MM	94.83±0.22	**96.27**±0.35	93.56±0.51	**90.29**±0.05
TAN-HC-MM-MM	**95.20**±0.21	**96.23**±0.35	**94.21**±0.49	**90.29**±0.05
TAN-SA-CR-MM	94.82±0.22	**96.10**±0.35	**94.26**±0.49	89.96±0.05
TAN-SA-MM-MM	**95.25**±0.21	**96.40**±0.35	**94.47**±0.48	**90.28**±0.05
SVM	96.40±0.19 $C^* = 1, \sigma = 0.005$	97.86±0.26 $C^* = 1, \sigma = 0.05$	94.57±0.47 $C^* = 1, \sigma = 0.01$	88.98±0.05 $C^* = 1, \sigma = 0.05$

Discriminative parameter learning produces a better classification per-

[8]Parameter h for the Huber loss of the MM parameter parameter learning [23] is set to 0.5. The value of λ for the margin criterion is empirically obtained during cross-validation.

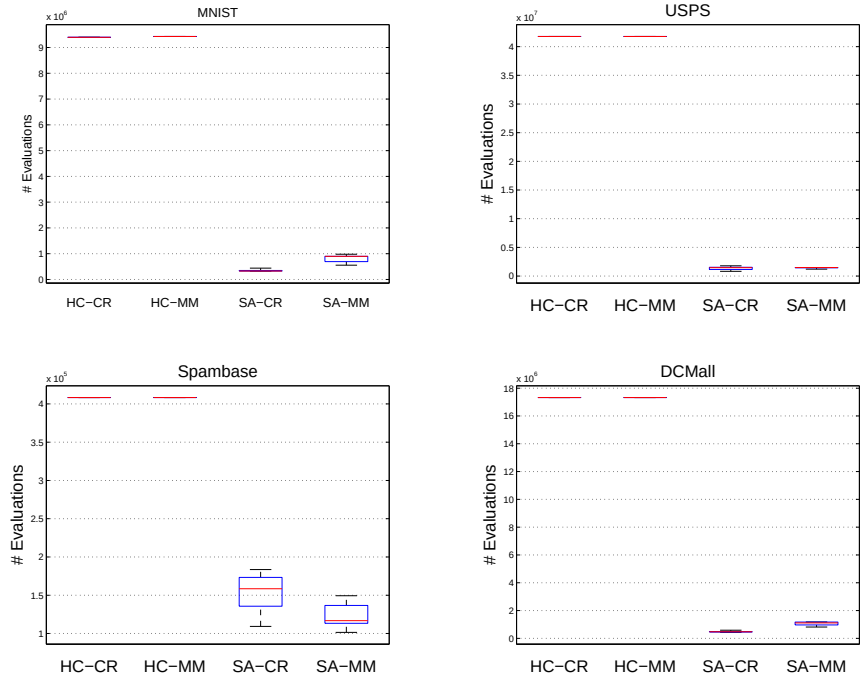

Figure 3: Number of score evaluations used to learn discriminatively structured Bayesian network classifiers. Horizontal red lines of the box plot corresponds to the median over 10 runs, the boxes represent the 25% and 75% quantiles, and the whiskers visualize the total range.

formance than ML parameter learning (see Table 1) on the same classifier structures. This is especially valid for cases where the structure of the underlying model is not optimized for classification [28], i.e. NB and TAN-CMI. SVMs[9] are slightly better than our discriminative Bayesian network classifier on three data sets, i.e. MNIST, USPS, and Spambase. In Table 3 we compare the model complexity, i.e. the number of parameters, between SVMs, the NB, and the best performing TAN Bayesian network classifier. This table reveals that for the cases where the Bayesian network achieve slightly

[9]SVMs with a radial basis function kernel are used. They have two parameters 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 kernel. The optimal choice of σ has been determined by 5-fold cross-validation on the training set.

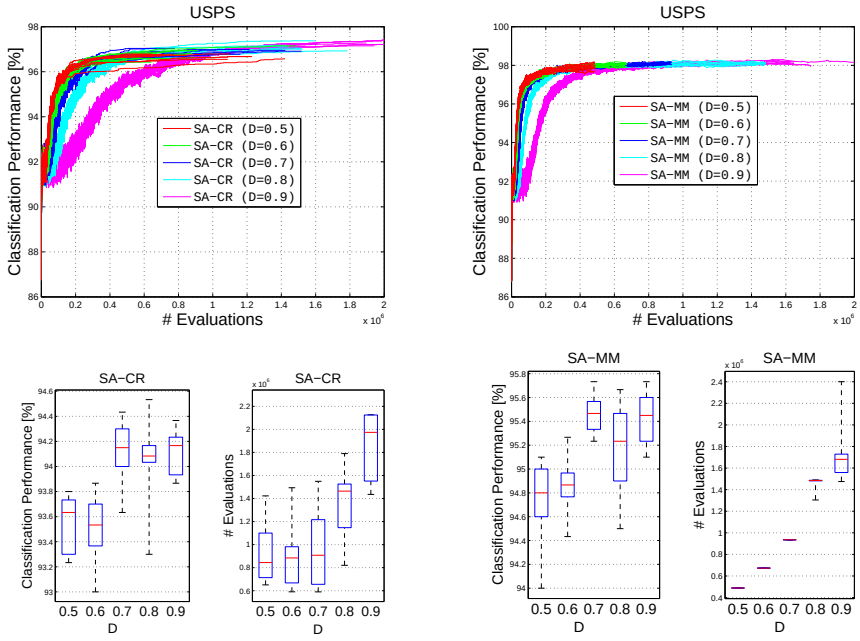

Figure 4: Influence of the annealing schedule for $D \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$ using the USPS data: SA-CR (left column) and SA-MM (right column). For each value of D, 10 stochastic structure learning runs have been performed.

inferior performance the model is dramatically smaller. The best Bayesian network uses ~ 107, ~ 44, and ~ 57 times fewer parameters than the SVM for MNIST, USPS, and Spambase, respectively. For DCMall the Bayesian network outperforms SVMs by $\sim 1\%$ but the model has twice as many parameters than the SVM. It is a well-known fact that the number of support vectors in classical SVMs increases linearly with the number of training samples [37]. In contrast, the structure of Bayesian network classifiers naturally limits the number of parameters. Furthermore, the used Bayesian network structures are probabilistic generative models. They might be preferred since it is easy to work with missing features, domain knowledge can be directly incorporated into the graph structure, and it is easy to work with structured data.

In the following, we verify that a discriminatively optimized generative model still offers its advantages in the missing feature case. Our MM parameter learning keeps the sum-to-one constraint of the probability distribu-

Table 3: Model complexity for NB, best TAN Bayesian network (BN), and SVM.

Data	N	# of SVs	# SVM parameters	# of NB BN parameters	# of TAN BN parameters
MNIST	196	17201	3371396	5159	31399
USPS	256	3837	982272	6089	22139
Spambase	57	517	29469	161	517
DCMall	191	11934	2279394	61228	4804505

tions [3]. Therefore, we suggest that we can similarly to the generatively optimized model simply sum over the missing feature values. The interpretation of *marginalizing* over missing features is delicate since the discriminatively optimized parameters might not have anything in common with *consistently* estimated probabilities (such as e.g. maximum likelihood estimation) [38]. However, at least empirically there is a strong support for using the marginal density $P(C, \mathbf{X}') = \sum_{\mathbf{X}_{1:N} \setminus \mathbf{X}'} P(C, \mathbf{X}_{1:N})$ where $\mathbf{X}'$ is a subset of the features in $\mathbf{X}_{1:N}$. This is tractable if the complexity class of $P(C, \mathbf{X}_{1:N})$ is limited (e.g., 1-tree) and the variable order in the summation is chosen appropriately. A discriminative classifier, however, is inherently conditional and it is not possible to go from $P(C|\mathbf{X}_{1:N})$ to $P(C|\mathbf{X}')$. This problem is also true for SVMs, logistic regression, and multi-layered perceptrons.

In Figure 5, we report classification results for NB-ML and NB-MM assuming missing features using the DCMall data. 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 moderate number of missing features. Discriminatively parameterized NB classifiers outperform NB-ML in the case of up to 150 missing features. Additionally, we present results for SVMs where missing features are replaced with the average of all training samples $\mathcal{S}$. This mean value imputation heavily degrades the classification performance of SVMs in case of missing features. Handling missing features with NB structures is easy since we can simply neglect the conditional probability of the missing feature Z_j in Eq. (1), i.e. the joint probability is the product of the available features only.

6. Conclusions

In this paper, we proposed the maximum margin score for learning discriminative Bayesian network classifier structures. Furthermore, we replaced traditional greedy hill-climbing heuristics for discriminative structure opti-

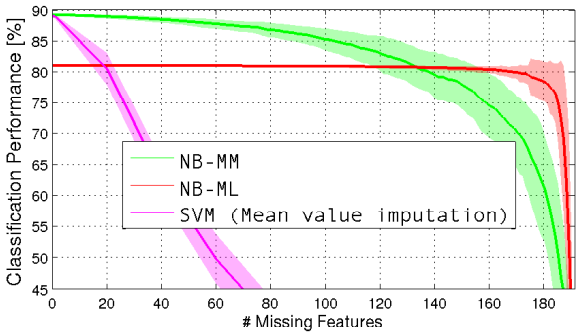

Figure 5: DCMall Data: Classification results for NB-ML, NB-MM, and SVMs (using mean value imputation) assuming missing features.

mization with stochastic simulated annealing methods. Simulated annealing offers mechanisms to escape local optima. The main results of the work are: (i) Maximum margin structure learning mostly outperform other generative and discriminative structure learning methods. (ii) Stochastic optimization leads for most cases to better performing Bayesian network structures and requires a lower number of score evaluations in comparison to greedy hill-climbing. (iii) We also provide results for discriminative parameter learning on top of generatively or discriminatively optimized structures. Margin-optimized Bayesian networks perform on par with SVMs in terms of classification rate, however the Bayesian network classifiers can directly deal with missing features, a case where discriminative classifiers usually require feature imputation techniques.

Acknowledgments

Thanks to Jeff Bilmes for discussions and support in writing this paper.

References

[1] C. Bishop, Pattern recognition and machine learning, Springer, 2006.

[2] A. Ng, M. Jordan, On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes, in: Advances in Neural Information Processing Systems 14, 2002, pp. 841–848.

[3] F. Pernkopf, J. Bilmes, Efficient heuristics for discriminative structure learning of Bayesian network classifiers, Journal of Machine Learning Res. 11 (2010) 2323–2360.

[4] D. M. Chickering, Learning Bayesian networks is NP-Complete, in: Learning from Data: Artificial Intelligence and Statistics V, Springer-Verlag, 1996, pp. 121–130.

[5] C. Meek, Causal inference and causal explanation with background knowledge, in: 11^{th} Conference on Uncertainty in Artificial Intelligence (UAI), 1995, pp. 403–410.

[6] S. Dasgupta, The sample complexity of learning fixed-structure Bayesian networks, Machine Learning 29 (2) (1997) 165–180.

[7] S. Arnborg, D. Corneil, A. Proskurowski, Complexity of finding embeddings in a k-tree, SIAM Journal of Algebraic and Discrete Methods 8 (2) (1987) 277–284.

[8] D. Karger, N. Srebro, Learning Markov networks: Maximum bounded tree-width graphs, in: Symposium on Discrete Algorithms, 2001, pp. 302–401.

[9] N. Srebro, Maximum likelihood bounded tree-width Markov networks, Artificial Intelligence 143 (1) (2003) 123–138.

[10] D. Geiger, D. Heckerman, Knowledge representation and inference in similarity networks and Bayesian multinets, Artificial Intelligence 82 (1996) 45–74.

[11] P. Parviainen, M. Koivisto, Exact structure discovery in Bayesian networks with less space, in: Uncertainty in Artificial Intelligence (UAI), 2009, pp. 436–443.

[12] C. de Campos, Z. Zeng, Q. Ji, Structure learning of bayesian networks using constraints, in: International Conference on Machine Learning (ICML), 2009, pp. 113–120.

[13] M. Teyssier, D. Koller, Ordering-based search: A simple and effective algorithm for learning Bayesian networks, in: 21^{th} Conference on Uncertainty in AI (UAI), 2005, pp. 584 – 590.

[14] T. Jaakkola, D. Sontag, A. Globerson, M. Meila, Learning Bayesian network structure using LP relaxations, in: Intern. Conf. on Artificial Intelligence and Statistics (AISTATS), 2010, pp. 358–365.

[15] E. Keogh, M. Pazzani, Learning augmented Bayesian classifiers: A comparison of distribution-based and classification-based approaches, in: International Workshop on Artificial Intelligence and Statistics, 1999, pp. 225–230.

[16] D. Grossman, P. Domingos, Learning Bayesian network classifiers by maximizing conditional likelihood, in: International Conference of Machine Learning (ICML), 2004, pp. 361–368.

[17] W. Buntine, Theory refinement on Bayesian networks, in: Uncertainty in Artificial Intelligence (UAI), 1991, pp. 52–60.

[18] G. Cooper, E. Herskovits, A Bayesian method for the induction of probabilistic networks from data, Machine Learning 9 (1992) 309–347.

[19] T. Cover, J. Thomas, Elements of information theory, John Wiley & Sons, 1991.

[20] V. Vapnik, Statistical learning theory, Wiley & Sons, 1998.

[21] B. Taskar, C. Guestrin, D. Koller, Max-margin markov networks, in: Advances in Neural Information Processing Systems (NIPS), 2003, pp. 25–32.

[22] Y. Guo, D. Wilkinson, D. Schuurmans, Maximum margin Bayesian networks, in: International Conference on Uncertainty in Artificial Intelligence (UAI), 2005, pp. 233–242.

[23] F. Pernkopf, M. Wohlmayr, S. Tschiatschek, Maximum margin Bayesian network classifiers, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (3) (2012) 521–532.

[24] Z. Michalewicz, D. Fogel, How to solve it: Modern Heuristics, Springer Verlag, 2000.

[25] T. Wang, J. Touchman, G. Xue, Applying two-level simulated annealing on bayesian structure learning to infer genetic networks, in: IEEE Computational Systems Bioinformatics Conference, 2004, pp. 647 – 648.

[26] S. Ye, H. Cai, R. Sun, An algorithm for Bayesian networks structure learning based on simulated annealing with MDL restriction, in: International Conference on Natural Computation (ICNC), 2008, pp. 72–76.

[27] N. Friedman, D. Geiger, M. Goldszmidt, Bayesian network classifiers, Machine Learning 29 (1997) 131–163.

[28] R. Greiner, X. Su, S. Shen, W. Zhou, Structural extension to logistic regression: Discriminative parameter learning of belief net classifiers, Machine Learning 59 (2005) 297–322.

[29] J. Pearl, Probabilistic reasoning in intelligent systems: Networks of plausible inference, Morgan Kaufmann, 1988.

[30] D. Koller, N. Friedman, Probabilistic graphical models, The MIT Press, 2009.

[31] F. Pernkopf, Bayesian network classifiers versus selective k-NN classifier, Pattern Recognition 38 (3) (2005) 1–10.

[32] P. Bartlett, M. Jordan, J. McAuliffe, Convexity, classification, and risk bounds, Journal of the American Statistical Association 101 (473) (2006) 138–156.

[33] B. Schölkopf, A. Smola, Learning with kernels: Support vector machines, regularization, optimization, and beyond, MIT Press, 2001.

[34] M. Pazzani, Searching for dependencies in Bayesian classifiers, in: Learning from data: Artificial intelligence and statistics V, 1996, pp. 239–248.

[35] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.

[36] A. Frank, A. Asuncion, UCI machine learning repository (2010).
URL `http://archive.ics.uci.edu/ml`

[37] R. Collobert, F. Sinz, J. Weston, L. Bottou, Trading convexity for scalability, in: International Conference on Machine Learning (ICML), 2006, pp. 201–208.

[38] T. Minka, Discriminative models, not discriminative training, Tech. Rep. MSR-TR-2005-144, Microsoft Research (2005).