Journal of Machine Learning Research 11 (2010) 2323-2360 Submitted 10/08; Revised 5/09; Published 8/10

Efficient Heuristics for Discriminative Structure Learning of Bayesian Network Classifiers

Franz Pernkopf PERNKOPF@TUGRAZ.AT
Department of Electrical Engineering
Graz University of Technology
A-8010 Graz, Austria

Jeff A. Bilmes BILMES@EE.WASHINGTON.EDU
Department of Electrical Engineering
University of Washington
Seattle, WA 98195, USA

Editor: Russ Greiner

Abstract

We introduce a simple order-based greedy heuristic for learning discriminative structure within generative Bayesian network classifiers. We propose two methods for establishing an order of N features. They are based on the conditional mutual information and classification rate (i.e., risk), respectively. Given an ordering, we can find a discriminative structure with $O\left(N^{k+1}\right)$ score evaluations (where constant k is the tree-width of the sub-graph over the attributes). We present results on 25 data sets from the UCI repository, for phonetic classification using the TIMIT database, for a visual surface inspection task, and for two handwritten digit recognition tasks. We provide classification performance for *both* discriminative *and* generative parameter learning on *both* discriminatively *and* generatively structured networks. The discriminative structure found by our new procedures significantly outperforms generatively produced structures, and achieves a classification accuracy on par with the best discriminative (greedy) Bayesian network learning approach, but does so with a factor of $\sim$10-40 speedup. We also show that the advantages of generative discriminatively structured Bayesian network classifiers still hold in the case of missing features, a case where generative classifiers have an advantage over discriminative classifiers.

Keywords: Bayesian networks, classification, discriminative learning, structure learning, graphical model, missing feature

1. Introduction

Bayesian networks (Pearl, 1988; Cowell et al., 1999) have been widely used as a space within which to search for high performing statistical pattern classifiers. Such networks can be produced in a number of ways, and ideally the structure of such networks will be learned discriminatively. By "discriminative learning" of Bayesian network structure, we mean simply that the process of learning corresponds to optimizing an objective function that is highly representative of classification error, such as maximizing class conditional likelihood, or minimizing classification error under the 0/1-loss function or some smooth convex upper-bound surrogate (Bartlett et al., 2006).

Unfortunately, learning the structure of Bayesian networks is hard. There have been a number of negative results over the past years, showing that optimally learning various forms of constrained Bayesian networks is NP-complete even in the "generative" sense. For example, it has been shown

that learning paths (Meek, 1995), polytrees (Dasgupta, 1997), k-trees (Arnborg et al., 1987) or bounded tree-width graphs (Karger and Srebro, 2001; Srebro, 2003), and general Bayesian networks (Geiger and Heckerman, 1996) are all instances of NP-complete optimization problems. Learning the best "discriminative structure" is no less difficult, largely because the cost functions that are needed to be optimized do not in general decompose (Lauritzen, 1996), but there has as of yet not been any formal hardness results in the discriminative case.

Discriminative optimization of a Bayesian network structure for the purposes of classification does have its advantages, however. For example, the resulting networks are amenable to interpretation compared to a purely discriminative model (the structure specifies conditional independencies between variables that may indicate distinctive aspects of how best to discern between objects Bilmes et al., 2001), it is simple to work with missing features and latent variables (as we show in this paper), and to incorporate prior knowledge (see below for further details). Since discriminative learning of such networks optimizes for only one inference scenario (e.g., classification) the resulting networks might be simpler or more parsimonious than generatively derived networks, may better abide Occam's razor, and may restore some of the benefits mentioned in Vapnik (1998).

Many heuristic methods have been produced in the past to learn the structure of Bayesian network classifiers. For example, Friedman et al. (1997) introduced the tree-augmented naive (TAN) Bayes approach, where a naive Bayes (NB) classifier is augmented with edges according to various conditional mutual information criteria. Bilmes (1999, 2000) introduced the *explaining away residual* (EAR) for discriminative structure learning of dynamic Bayesian networks for speech recognition applications, which also happens to correspond to "synergy" in the neural code (Brenner et al., 2000). The EAR measure is in fact an approximation to the expected class conditional distribution, and so improving EAR is likely to decrease the KL-divergence between the true class posterior and the resultant approximate class posterior. A procedure for providing a local optimum of the EAR measure was outlined in Narasimhan and Bilmes (2005) but it may be computationally expensive. Greiner and Zhou (2002); Greiner et al. (2005) express general Bayesian networks as standard logistic regression—they optimize parameters with respect to the conditional likelihood (CL) using a conjugate gradient method. Similarly, Roos et al. (2005) provide conditions for general Bayesian networks under which the correspondence to logistic regression holds. In Grossman and Domingos (2004) the conditional log likelihood (CLL) function is used to learn a discriminative structure. The parameters are set using maximum likelihood (ML) learning. They use a greedy hill climbing search with the CLL function as a scoring measure, where at each iteration one edge is added to the structure which conforms with the restrictions of the network topology (e.g., TAN) and the acyclicity property of Bayesian networks. In a similar algorithm, the classification rate (CR)[1] has also been used for discriminative structure learning (Keogh and Pazzani, 1999; Pernkopf, 2005). The hill climbing search is terminated when there is no edge which further improves the CR. The CR is the discriminative criterion with the fewest approximations, so it is expected to perform well given sufficient data. The problem, however, is that this approach is extremely computationally expensive, as a complete re-evaluation of the training set is needed for each considered edge. Many generative structure learning algorithms have been proposed and are reviewed in Heckerman (1995), Murphy (2002), Jordan (1999) and Cooper and Herskovits (1992). Independence tests may also be used for generative structure learning using, say, mutual information (de Campos, 2006) while other recent

1. Maximizing CR is equivalent to minimizing classification error which is identical to empirical risk (Vapnik, 1998) under the 0/1-loss function. We use the CR terminology in this paper since it is somewhat more consistent with previous Bayesian network discriminative structure learning literature.

independence test work includes Gretton and Gÿorfi (2008) and Zhang et al. (2009). An experimental comparison of discriminative and generative parameter training on both discriminatively and generatively structured Bayesian network classifiers has been performed in Pernkopf and Bilmes (2005). An empirical and theoretical comparison of certain discriminative and generative classifiers (specifically logistic regression and NB) is given in Ng and Jordan (2002). It is shown that for small sample sizes the generative NB classifier can outperform the discriminative model.

This work contains the following offerings. First, a new case is made for why and when discriminatively structured generative models can be usefully used to solve multi-class classification problems.

Second, we introduce a new order-based greedy search heuristic for finding discriminative structures in generative Bayesian network classifiers that is computationally efficient and that matches the performance of the currently top-performing but computationally expensive greedy "classification rate" approach. Our resulting classifiers are restricted to TAN 1-tree and TAN 2-trees, and so our method is a form of search within a complexity-constrained model space. The approach we employ looks first for an ordering of the N features according to classification based information measures. Given the resulting ordering, the algorithm efficiently discovers high-performing discriminative network structure with no more than $O\left(N^{k+1}\right)$ score evaluations where k indicates the tree-width of the sub-graph over the attributes, and where a score evaluation can either be a mutual-information or a classification error-rate query. Our order-based structure learning is based on the observations in Buntine (1991) and the framework is similar to the K2 algorithm proposed in Cooper and Herskovits (1992). We use, however, a discriminative scoring metric and suggest approaches for establishing the variable ordering based on conditional mutual information (CMI) (Cover and Thomas, 1991) and CR.

Lastly, we provide a wide variety of empirical results on a diverse collection of data sets showing that the order-based heuristic provides comparable classification results to the best procedure - the greedy heuristic using the CR score, but our approach is computationally much cheaper. Furthermore, we empirically show that the chosen approaches for ordering the variables improve the classification performance compared to simple random orderings. We experimentally compare both discriminative and generative parameter training on *both* discriminative *and* generatively structured Bayesian network classifiers. Moreover, one of the key advantages of generative models over discriminative ones is that it is still possible to marginalize away any missing features. If it is not known at training time which features might be missing, a typical discriminative model is rendered unusable. We provide empirical results showing that discriminatively learned generative models are reasonably insensitive to such missing features and retain their advantages over generative models in such case.

The organization of the paper is as follows: In Section 2, Bayesian networks are reviewed and our notation is introduced. We briefly present the NB, TAN, and 2-tree network structures. In Section 3, a practical case is made for why discriminative structure can be desirable. The most commonly used approaches for generative and discriminative structure and parameter learning are summarized in Section 4. Section 5 introduces our order-based greedy heuristic. In Section 6, we report classification results on 25 data sets from the UCI repository (Merz et al., 1997) and from Kohavi and John (1997) using all combinations of generative/discriminative structure/parameter learning. Additionally, we present classification experiments for synthetic data, for frame- and segment-based phonetic classification using the TIMIT speech corpus (Lamel et al., 1986), for a visual surface inspection task (Pernkopf, 2004), and for handwritten digit recognition using the MNIST (LeCun

et al., 1998) and USPS data set. Last, Section 7 concludes. We note that a preliminary version of a subset of our results appeared in Pernkopf and Bilmes (2008b).

2. Bayesian Network Classifiers

A Bayesian network (BN) (Pearl, 1988; Cowell et al., 1999) $\mathcal{B} = \langle \mathcal{G}, \Theta \rangle$ 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 $\mathbf{E} = \{E_{Z_i,Z_j}, E_{Z_i,Z_k}, \ldots\}$ 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 $\mathbf{Z} = \{Z_1, \ldots, Z_{N+1}\}$. Each variable in $\mathbf{Z}$ has values denoted by lower case letters $\{z_1, z_2, \ldots, z_{N+1}\}$. We use boldface capital letters, for example, $\mathbf{Z}$, 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. Each graph node represents a random variable, while the lack of edges in a graph specifies some conditional independence relationships. Specifically, in a Bayesian network each node is independent of its non-descendants given its parents (Lauritzen, 1996). A Bayesian network's conditional independence relationships arise due to missing parents in the graph. Moreover, conditional independence can reduce 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, $\theta^j_{i|h} = P_\Theta\left(Z_j = i | Z_{\Pi_j} = h\right)$. Hence, h contains the parent configuration assuming that the first element of h, that is, 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 $\mathcal{S} = \{\mathbf{z}^m\}_{m=1}^M = \left\{\left(c^m, \mathbf{x}^m_{1:N}\right)\right\}_{m=1}^M$. For most of this work, we assume a complete data set with no missing values (the exception being Section 6.6 where input features are missing at test time). The joint probability distribution of the network is determined by the local conditional probability distributions as

$$P_\Theta(\mathbf{Z}) = \prod_{j=1}^{N+1} P_\Theta\left(Z_j | Z_{\Pi_j}\right)$$

and the probability of a sample $\mathbf{z}^m$ is

$$P_\Theta(\mathbf{Z} = \mathbf{z}^m) = \prod_{j=1}^{N+1} \prod_{i=1}^{|Z_j|} \prod_h \left(\theta^j_{i|h}\right)^{u^{j,m}_{i|h}},$$

where we introduced an indicator function $u^{j,m}_{i|h}$ of the m^{th} sample

$$u^{j,m}_{i|h} = \begin{cases} 1, & \text{if } z^m_j = i \text{ and } z^m_{\Pi_j} = h \\ 0, & \text{otherwise} \end{cases}.$$

In this paper, we restrict our experiments to NB, TAN 1-tree (Friedman et al., 1997), and TAN 2-tree classifier structures (defined in the next several paragraphs). 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 (Friedman et al., 1997; Domingos and Pazzani, 1997), the performance of the NB classifier is surprisingly good even if the conditional independence assumption between attributes is unrealistic or even false in most of the data. Reasons for the utility of the NB classifier range between benefits from the bias/variance tradeoff perspective (Friedman et al., 1997) to structures that are inherently poor from a generative perspective but good from a discriminative perspective (Bilmes, 2000). The structure of the naive Bayes classifier represented as a Bayesian network is illustrated in Figure 1a.

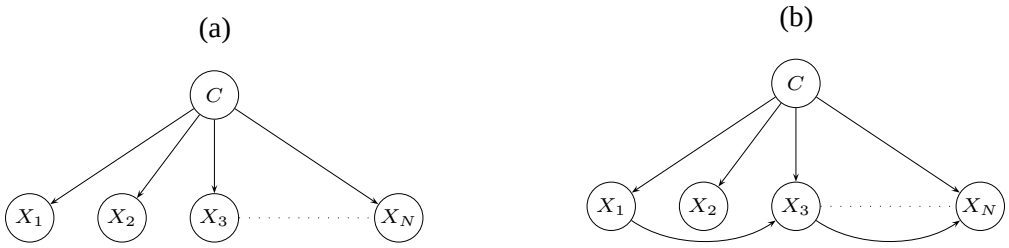

Figure 1: Bayesian Network: (a) NB, (b) TAN.

In order to correct some of the limitations of the NB classifier, Friedman et al. (1997) 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, that is, 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 $N-1$. Thus, two attributes might not be conditionally independent given the class label in a TAN. An example of a TAN 1-tree network is shown in Figure 1b. 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 to use a k-tree, that is, 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, that is, $C_\Pi = \emptyset$. While many other network topologies have been suggested in the past (and a good overview is provided in Acid et al., 2005), in this work we keep the class variable parent-less since it allows us to achieve one of our goals, which is to concentrating on generative models and their structures.

3. Discriminative Learning in Generative Models

A dichotomy exists between the two primary approaches to statistical pattern classifiers, *generative* and *discriminative* (Bishop and Lasserre, 2007; Jebara, 2001; Ng and Jordan, 2002; Bishop, 2006; Raina et al., 2004; Juang et al., 1997; Juang and Katagiri, 1992; Bahl et al., 1986). Under generative models, what is learned is a model of the joint probability of the features $\mathbf{X}$ and the corresponding class random variables C. Complexity penalized likelihood of the data is often the objective used for optimization, leading to standard maximum likelihood (ML) learning. Prediction with such a model is then performed either by using Bayes rule to form the class posterior probability or equivalently by forming class-prior penalized likelihood. Generative models have been widely studied, and are

desirable because they are amenable to interpretation (e.g., the structure of a generative Bayesian network specifies conditional independencies between variables that might have a useful high-level explanation). Additionally, they are amenable to a variety of probabilistic inference scenarios owing to the fact that they often decompose (Lauritzen, 1996)—the decomposition (or factorization) properties of a model are often crucial to their efficient computation.

Discriminative approaches, on the other hand, more directly represent aspects of the distribution that are important for classification accuracy, and there are a number of ways this can be done. For example, some approaches model only the class posterior probability (the conditional probability of the class given the features) or $p(C|\mathbf{X})$. Other approaches, such as support vector machines (SVMs) (Schölkopf and Smola, 2001; Burges, 1998) or neural networks (Bishop, 2006, 1995), directly model information about decision boundary sometimes without needing to concentrate on obtaining an accurate conditional distribution (neural networks, however, are also used to produce conditional distributions above and beyond just getting the class ranks correct Bishop, 1995). In each case, the objective function that is optimized is one whose minima occur not necessarily when the joint distribution $p(C,\mathbf{X})$ is accurate, but rather when the classification error rate on a training set is small. Discriminative models are usually restricted to one particular inference scenario, that is, the mapping from observed input features $\mathbf{X}$ to the unknown class output variable C, and not the other way around.

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 (Vapnik, 1998). The superior performance of discriminative classifiers has been reported in many application domains (Ng and Jordan, 2002; Raina et al., 2004; Juang et al., 1997; Juang and Katagiri, 1992; Bahl et al., 1986).

Why then should we have an interest in generative models for discrimination? We address this question in the next several paragraphs. The distinction between generative and discriminative models becomes somewhat blurred when one considers that there are both generative and discriminative methods to learn a generative model, and within a generative model one may make a distinction between learning model structure and learning its parameters. In fact, in this paper, we make a clear distinction between learning the parameters of a generative model and learning the structure of a generative model. When using Bayesian networks to describe factorization properties of generative models, the structure of the model corresponds to the graph: fixing the graph, the parameters of the model are such that they must respect the factorization properties expressed by that graph. The structure of the model, however, can be independently learned, and different structures correspond to different families of graph (each family is spanned by the parameters respecting a particular structure). A given structure is then evaluated under a particular "best" set of parameter values, one possibility being the maximum likelihood settings. Of course, one could consider optimizing both parameters and structure simultaneously. Indeed, both structure and parameters are "parameters" of the model, and it is possible to learn the structure along with the parameters when a complexity penalty is applied that encourages sparse solutions, such as ℓ_1-regularization (Tibshirani, 1996) in linear regression and other models. We, however, find it useful to maintain this distinction between structure and parameters for the reason that parameter learning is inherently a continuous optimization procedure, while structure learning is inherently a combinatorial optimization problem. In our case, moreover, it is possible to stay within a given fixed-complexity model family—if we wish to stay within the family of say k-trees for fixed k, ℓ_1 regularization is not guaranteed to oblige.

Moreover, both parameters and structure of a generative model can be learned either generatively or discriminatively. Discriminative parameter learning of generative models, such as hidden Markov models (HMMs) has occurred for many years in the speech recognition community (Bahl et al., 1986; Ephraim et al., 1989; Ephraim and Rabiner, 1990; Juang and Katagiri, 1992; Juang et al., 1997; Heigold et al., 2008), and more recently in the machine learning community (Greiner and Zhou, 2002; Greiner et al., 2005; Roos et al., 2005; Ng and Jordan, 2002; Bishop and Lasserre, 2007; Pernkopf and Wohlmayr, 2009). Discriminative structure learning has also more recently received some attention (Bilmes, 1999, 2000; Pernkopf and Bilmes, 2005; Keogh and Pazzani, 1999; Grossman and Domingos, 2004). In fact, there are four possible cases of learning a generative model as depicted in Figure 2. Case A is when both structure and parameter learning is generative. Case B is when the structure is learned generatively, but the parameters are learned discriminatively. Case C is the mirror image of case B. Case D, potentially the most preferable case for classification, is where both the structure and parameters are discriminatively learned.

		Parameter Learning	
		Generative	Discriminative
Structure Learning	Generative	Case A	Case B
	Discriminative	Case C	Case D

Figure 2: Learning generative-model based classifiers: Cases for each possible combination of generative and discriminative learning of either the parameters or the structure of Bayesian network classifiers.

In this paper, we are particularly interested in learning the discriminative structure of a generative model. With a generative model, even discriminatively structured, some aspect of the joint distribution $p(C,\mathbf{X})$ is still being represented. Of course, a discriminatively structured generative model needs only represent that aspect of the joint distribution that is beneficial from a classification error rate perspective, and need not "generate" well (Bilmes et al., 2001). For this reason, it is likely that a discriminatively trained generative model will not need to be as complex as an accurate generatively trained model. In other words, the advantage of parsimony of a discriminative model over a generative model will likely be partially if not mostly recovered when one trains a generative model discriminatively. Moreover, there are a number of reasons why one might, in certain contexts, prefer a generative to a discriminative model including: parameter tying and domain knowledge-based hierarchical decomposition is facilitated; it is easy to work with structured data; there is less sensitivity to training data class skew; generative models can still be trained and structured discriminatively (as mentioned above); and it is easy to work with missing features by marginalizing over the unknown variables. This last point is particularly important: a discriminatively structured generative model still has the ability to go from $p(C,\mathbf{X})$ to $p(C,\mathbf{X}')$ where $\mathbf{X}'$ is a subset of the features in $\mathbf{X}$. This amounts to performing the marginalization $p(C,\mathbf{X}') = \sum_{\mathbf{X}\setminus\mathbf{X}'} p(C,\mathbf{X})$, something that can be tractable if the complexity class of $p(C,\mathbf{X})$ is limited (e.g., k-trees) and the variable order in the summation is chosen appropriately. In this work, we verify that a discriminatively structured model retains its advantages in the missing feature case (see Section 6.6). A discriminative model, however, is inherently conditional and it is not possible in general when some of the features are missing to go from $p(C|\mathbf{X})$ to $p(C|\mathbf{X}')$. This problem is also true for SVMs, logistic regression, and multi-layered perceptrons.

Learning a discriminatively structured generative model is inherently a combinatorial optimization problem on a "discriminative" objective function. This means that there is an algorithm that operates by tending to prefer structures that perform better on some measure that is related to classification error. Assuming sufficient training data, the ideal objective function is empirical risk under the 0/1-loss (what we call CR, or the average error rate over training data), which can be implicitly regularized by constraining the optimization process to consider only a limited complexity model family (e.g., k-trees for fixed k). In the case of discriminative parameter learning, CR can be used, but typically alternative continuous and differentiable cost functions, which may upper-bound CR and might be convex (Bartlett et al., 2006), are used and include conditional (log) likelihood $CLL(\mathcal{B}|\mathcal{S}) = \log \prod_{m=1}^{M} P_\Theta\left(C = c^m | \mathbf{X}_{1:N} = \mathbf{x}_{1:N}^m\right)$—this last objective function in fact corresponds to maximizing the mutual information between the class variable and the features (Bilmes, 2000), and can easily be augmented by a regularization term as well.

One may ask, given discriminative parameter learning, is discriminative structure still necessary? In the following, we present a simple synthetic example (similar to Narasimhan and Bilmes, 2005) and actual training and test results that indicate when a discriminative structure would be necessary for good classification performance in a generative model. The model consists of 3 binary valued attributes X_1, X_2, X_3 and a binary uniformly distributed class variable C. $\bar{X_1}$ denotes the negation of X_1. For both classes, X_1 is uniformly distributed and $X_2 = X_1$ with probability 0.5 and a uniformly distributed random number with probability 0.5. So we have the following probabilities for both classes:

$$X_1 := \begin{cases} 0 & \text{with probability } 0.5 \\ 1 & \text{with probability } 0.5 \end{cases}$$

$$X_2 := \begin{cases} X_1 & \text{with probability } 0.5 \\ 0 & \text{with probability } 0.25 \\ 1 & \text{with probability } 0.25 \end{cases}$$

For class 1, X_3 is determined according to the following:

$$X_3 := \begin{cases} X_1 & \text{with probability } 0.3 \\ X_2 & \text{with probability } 0.5 \\ 0 & \text{with probability } 0.1 \\ 1 & \text{with probability } 0.1 \end{cases}.$$

For class 2, X_3 is given by:

$$X_3 := \begin{cases} \bar{X_1} & \text{with probability } 0.3 \\ X_2 & \text{with probability } 0.5 \\ 0 & \text{with probability } 0.1 \\ 1 & \text{with probability } 0.1 \end{cases}.$$

For both classes, the dependence between $X_1 - X_2$ is strong. The dependence $X_2 - X_3$ is stronger than $X_1 - X_3$, but only from a generative perspective (i.e., $I(X_2;X_3) > I(X_1;X_3)$ and $I(X_2;X_3|C) > I(X_1;X_3|C)$). Hence, if we were to use the strength of mutual information, or conditional mutual information, to choose the edge, we would choose $X_2 - X_3$. However, it is the $X_1 - X_3$ dependency that enables discrimination between the classes. Sampling from this distribution, we first learn structures using generative and discriminative methods, and then we perform parameter training

on these structures using either ML or CL (Greiner et al., 2005). For learning a generative TAN structure, we use the algorithm proposed by Friedman et al. (1997) which is based on optimizing the CMI between attributes given the class variable. For learning a discriminative structure, we apply our order-based algorithm proposed in Section 5 (we note that optimizing the EAR measure (Pernkopf and Bilmes, 2005) leads to similar results in this case).

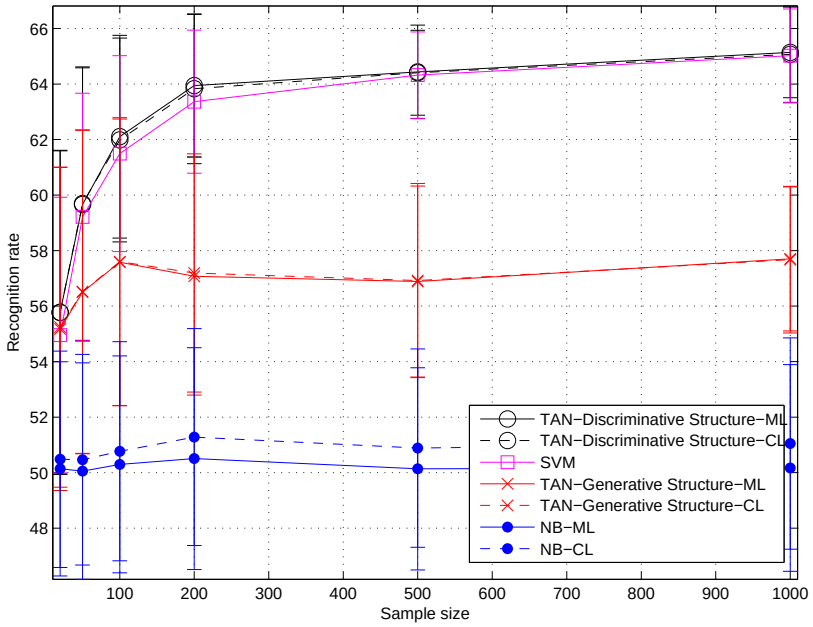

Figure 3: Generative and discriminative learning of Bayesian network classifiers on synthetic data.

Figure 3 compares the classification performance of these various cases, and in addition we show results for a NB classifier, which resorts to random guessing between both classes due to the lack of any feature dependency. Additionally, we provide the classification performance achieved with SVMs using a radial basis function (RBF) kernel.[2] On the *x*-axis, the training set *sample size* varies according to $\{20, 50, 100, 200, 500, 1000\}$ and the test data set contains 1000 samples. Plots are averaged over 100 independent simulations. The solid line is the performance of the classifiers using ML parameter learning, whereas, the dashed line corresponds to CL parameter training.

Figure 4: (a) Generatively learned 1-tree, (b) Discriminatively learned 1-tree.

Figure 4 shows (a) the generative (b) the discriminative 1-tree over the attributes of the resulting TAN network (the class variable which is the parent of each feature is not shown in this figure). A generative model prefers edges between $X_1 - X_2$ and $X_2 - X_3$ which do not help discrimination.

2. The SVM uses 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 RBF kernel. We set the values for these parameters to $C^* = 3$ and $\sigma = 1$.

The dependency between X_1 and X_3 enables discrimination to occur. Note that for this example the difference between ML and CL parameter learning is insignificant and for the generative model, only a discriminative structure enables correct classification. The performance of the non-generative SVM is similar to our discriminatively structured Bayesian network classifier. Therefore, when a generative model is desirable (see the reasons why this might be the case above), there is clearly a need for good discriminative structure learning.

In this paper, we show that the loss of a "generative meaning" of a generative model (when it is structured discriminatively) does not impair the generative model's ability to easily deal with missing features (Figure 11).

4. Learning Bayesian Networks

In the following sections, we briefly summarize state-of-the-art generative and discriminative structure and parameter learning procedures that are used to compare our order-based discriminative structure learning heuristics (which will be described in Section 5 and evaluated in Section 6).

4.1 Generative 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 $\theta^j_{i|h}$. The log likelihood function of a fixed structure of $\mathcal{B}$ is

$$\begin{aligned} LL(\mathcal{B}|\mathcal{S}) = \sum_{m=1}^{M} \log P_\Theta(\mathbf{Z}=\mathbf{z}^m) = \sum_{m=1}^{M}\sum_{j=1}^{N+1} \log P_\Theta\left(Z_j = z_j^m | Z_{\Pi_j} = z_{\Pi_j}^m\right) = \\ \sum_{m=1}^{M}\sum_{j=1}^{N+1}\sum_{i=1}^{|Z_j|}\sum_{h} u_{i|h}^{j,m} \log\left(\theta^j_{i|h}\right). \end{aligned} \tag{1}$$

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

$$\theta^j_{i|h} = \frac{\sum_{m=1}^{M} u_{i|h}^{j,m}}{\sum_{m=1}^{M}\sum_{l=1}^{|Z_j|} u_{l|h}^{j,m}},$$

using Lagrange multipliers to constrain the parameters to a valid normalized probability distribution. Since we are optimizing over constrained BN structures (k-trees), we do not perform any further regularization during training other than simple smoothing to remove zero-probability entries (see Section 6.1).

4.2 Discriminative Parameter Learning

As mentioned above, for classification purposes, having a good approximation to the posterior probability is sufficient. 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 and we have to resort to iterative optimization techniques. The objective function of the conditional log likelihood

is

$$CLL(\mathcal{B}|\mathcal{S}) = \log \prod_{m=1}^{M} P_\Theta(C = c^m | \mathbf{X}_{1:N} = \mathbf{x}^m_{1:N}) = \sum_{m=1}^{M} \log \frac{P_\Theta\left(C = c^m, \mathbf{X}_{1:N} = \mathbf{x}^m_{1:N}\right)}{\sum_{c=1}^{|C|} P_\Theta\left(C = c, \mathbf{X}_{1:N} = \mathbf{x}^m_{1:N}\right)} =$$

$$\sum_{m=1}^{M} \left[\log P_\Theta(C = c^m, \mathbf{X}_{1:N} = \mathbf{x}^m_{1:N}) - \log \sum_{c=1}^{|C|} P_\Theta(C = c, \mathbf{X}_{1:N} = \mathbf{x}^m_{1:N}) \right].$$

Similar to Greiner and Zhou (2002) we use a conjugate gradient algorithm with line-search (Press et al., 1992). In particular, the *Polak-Ribiere* method is used (Bishop, 1995). The derivative of the objective function is

$$\frac{\partial CLL(\mathcal{B}|\mathcal{S})}{\partial \theta^j_{i|h}} = \sum_{m=1}^{M} \left[\frac{\partial}{\partial \theta^j_{i|h}} \log P_\Theta(C = c^m, \mathbf{X}_{1:N} = \mathbf{x}^m_{1:N}) - \right.$$

$$\left. \frac{1}{\sum_{c=1}^{|C|} P_\Theta\left(C = c, \mathbf{X}_{1:N} = \mathbf{x}^m_{1:N}\right)} \frac{\partial}{\partial \theta^j_{i|h}} \sum_{c=1}^{|C|} P_\Theta(C = c, \mathbf{X}_{1:N} = \mathbf{x}^m_{1:N}) \right].$$

Further, 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 $h_1 \in h$.

4.2.1 CASE A

For the class variable, that is, $j = 1$ and $h = \emptyset$, we get

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

where we use Equation 1 for deriving the first term (omitting the sum over j and h) and we introduced the posterior

$$W^m_i = P_\Theta(C = i | \mathbf{X}_{1:N} = \mathbf{x}^m_{1:N}) = \frac{P_\Theta\left(C = i, \mathbf{X}_{1:N} = \mathbf{x}^m_{1:N}\right)}{\sum_{c=1}^{|C|} P_\Theta\left(C = c, \mathbf{X}_{1:N} = \mathbf{x}^m_{1:N}\right)}.$$

4.2.2 CASE B

For the attribute variables, that is, $j > 1$, we derive correspondingly and have

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

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

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

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 $\beta^j_{i|h}$ as follows

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

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

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

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

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

4.3 Generative Structure Learning

The conditional mutual information between the attributes given the class variable is computed as:

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

This measures the information between X_i and X_j in the context of C. Friedman et al. (1997) gives an algorithm for constructing a TAN network using this measure. This algorithm is an extension of the approach in Chow and Liu (1968). We briefly review this algorithm in the following:

1. Compute the pairwise CMI $I\left(X_i; X_j|C\right) \quad \forall \quad 1 \leq i \leq N$ and $i < j \leq N$.

2. Build an undirected 1-tree using the maximal weighted spanning tree algorithm (Kruskal, 1956) where each edge connecting X_i and X_j is weighted by $I\left(X_i; X_j|C\right)$.

3. Transform the undirected 1-tree to a directed tree. That is, select a root variable and direct all edges away from this root. Add to this tree the class node C and the edges from C to all attributes $X_1, \ldots, X_N$.

4.4 Discriminative Structure Learning

As a baseline discriminative structure learning method, we use a greedy edge augmentation method and also the *SuperParent* algorithm (Keogh and Pazzani, 1999).

4.4.1 GREEDY HEURISTICS

While this method is expected to perform well, it is much more computationally costly then the method we propose below. The method proceeds as follows: a 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 (defined below). This process is terminated when there is no edge which further improves the score. This process might thus result in a partial 1-tree (forest) over the attributes. This approach is computationally expensive since each time an edge is added, the scores for all $O\left(N^2\right)$ edges need to be re-evaluated due to the discriminative non-decomposable scoring functions we employ. This method overall has cost $O\left(N^3\right)$ score evaluations to produce a 1-tree, which in the case of an $O(NM)$) score evaluation cost (such as the below), has an overall complexity of $O\left(N^4\right)$. There are two score functions we consider: the CR (Keogh and Pazzani, 1999; Pernkopf, 2005)

$$CR\left(\mathcal{B}_{\mathcal{S}}|\mathcal{S}\right)=\frac{1}{M}\sum_{m=1}^{M}\delta\left(\mathcal{B}_{\mathcal{S}}\left(\mathbf{x}_{1:N}^{m}\right),c^{m}\right)$$

and the CL (Grossman and Domingos, 2004)

$$CL\left(\mathcal{B}|\mathcal{S}\right)=\prod_{m=1}^{M}P_{\Theta}\left(C=c^{m}|\mathbf{X}_{1:N}=\mathbf{x}_{1:N}^{m}\right),$$

where the expression $\delta\left(\mathcal{B}_{\mathcal{S}}\left(\mathbf{x}_{1:N}^{m}\right),c^{m}\right)=1$ if the Bayesian network classifier $B_{\mathcal{S}}\left(\mathbf{x}_{1:N}^{m}\right)$ trained with samples in $\mathcal{S}$ assigns the correct class label c^{m} to the attribute values $\mathbf{x}_{1:N}^{m}$, and is equal to 0 otherwise.[3] In our experiments, we consider the CR score which is directly related to the empirical risk in Vapnik (1998). The CR is the discriminative criterion that, given sufficient training data, most directly judges what we wish to optimize (error rate), while an alternative would be to use a convex upper-bound on the 0/1-loss function (Bartlett et al., 2006). Like in the generative case above, since we are optimizing over a constrained model space (k-trees), and are performing simple parameter smoothing, again regularization is implicit. This approach has in the literature been shown to be the algorithm that produces the best performing discriminative structure (Keogh and Pazzani, 1999; Pernkopf, 2005) but at the cost of a very expensive optimization procedure. To accelerate this algorithm in our implementation of this procedure (which we use as a baseline to compare against our still to-be-defined proposed approach), we apply two 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 (Pazzani, 1996).

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 memorize the joint probability and exchange only the probabilities of those nodes where the set of parents changed to get the new joint probability (Keogh and Pazzani, 1999).

In the experiments this greedy heuristic is labeled as TAN-CR and 2-tree-CR for 1-tree and 2-tree structures, respectively.

4.4.2 SUPERPARENT AND ITS k-TREE GENERALIZATION

Keogh and Pazzani (1999) introduced the *SuperParent* algorithm to efficiently learn a discriminative TAN structure. The algorithm starts with a NB network and the edges pointing from the class

3. Note that the CR scoring measure is determined from a classifier trained and tested on the same data $\mathcal{S}$.

variable to each attribute remain fixed throughout the algorithm. In the first step, each attribute in turn is considered as a parent of all other parentless attributes (except the class variable). If there are no parentless attributes left, the algorithm terminates. The parent which improves the CR the most is selected and designated the current superparent. The second step fixes the most recently chosen superparent and keeps only the single best child attribute of that superparent. The single edge between superparent and best child is then kept and the process of selecting a new superparent is repeated, unless no improvement is found at which point the algorithm terminates. The number of CR evaluations therefore in a complete run of the algorithm is $O\left(N^2\right)$. Moreover, CR determination can be accelerated as mentioned above.

We can extend this heuristic to learn 2-trees by simply modifying the first step accordingly: consider each attribute as an additional parent of all parentless or single-parented attributes (while ensuring acyclicity), and choose as the superparent the one that evaluates best, requiring $O(N)$ CR evaluations. Next, we retain the pair of edges between superparent and (parentless or single-parented) children that evaluates best using CR, requiring $O\left(N^2\right)$ CR evaluations. The process repeats if successful and otherwise terminates. The obvious k-tree generalization modifies the first step to choose an additional parent of all attributes with fewer than k parents, and then selects the best children for edge retention, leading overall to a process with $O\left(N^{k+1}\right)$ score evaluations. In this paper, we compare against this heuristic in the case of $k = 1$ and $k = 2$, abbreviating them, respectively, as *TAN-SuperParent* and *2-tree-SuperParent*.

5. New Heuristics: Order-based Greedy Algorithms

It was first noticed in Buntine (1991); Cooper and Herskovits (1992) that the best network consistent with a given variable ordering can be found with $O(N^{q+c})$ score evaluations where q is the maximum number of parents per node in a Bayesian network, and where c is a small fixed constant. These facts were recently exploited in Teyssier and Koller (2005) where generative structures were learned. Here, we are inspired by these ideas and apply them to the case of learning discriminative structures. Also, unlike Teyssier and Koller (2005), we establish only one ordering, and since our scoring cost is discriminative, it does not decompose and the learned discriminative structure is not guaranteed to be optimal. However, experiments show good results at relatively low computational learning costs.

Our procedure looks first for a total ordering $\prec$ of the variables $\mathbf{X}_{1:N}$ according to certain criteria which are outlined below. The parents of each node are chosen in such a way that the ordering is respected, and that the procedure results in at most a k-tree. We note here, a k-tree is typically defined on an undirected graphical model as one that has a tree-width of k—equivalently, there exists an elimination order in the graph such that at each elimination step, the node being eliminated has no more than k neighbors at the time of elimination. When we speak of a Bayesian network being a k-tree, what we really mean is that the moralized version of the Bayesian network is a k-tree. As a reminder, our approach is to learn a k-tree (i.e., a computationally and parameter constrained Bayesian network) over the features $\mathbf{X}_{1:N}$. We still assume, as is done with a naive Bayes model, that C is a parent of each X_i and this additional is not counted in k—thus, a 1-tree would have two parents for each X_i, both C and one additional feature. As mentioned above, in order to stay strictly within the realm of generative models, we do not consider the case where C has any parents.

5.1 Step 1: Establishing an Order $\prec$

We propose three separate heuristics for establishing an ordering $\prec$ of the attribute nodes prior to parent selection. In each case as we will see later, we use the resulting ordering such that later features may only have earlier features as parents—this limit placed on the set of parents leads to reduced computational complexity. Two of the heuristics are based on the conditional mutual information $I(C;\mathbf{X}_A|\mathbf{X}_B)$ between the class variable C and some subset of the features $\mathbf{X}_A$ given some disjoint subset of variables $\mathbf{X}_B$ (so $A \cap B = \emptyset$). The conditional mutual information (CMI) measures the degree of dependence between the class variable and $\mathbf{X}_A$ given $\mathbf{X}_B$ and it may be expressed entirely in terms of entropy via $I(C;\mathbf{X}_A|\mathbf{X}_B) = -H(C,\mathbf{X}_A,\mathbf{X}_B) + H(\mathbf{X}_A,\mathbf{X}_B) + H(C,\mathbf{X}_B) - H(\mathbf{X}_B)$, where entropy of X is defined as $H(X) = -\sum_x p(x)\log p(x)$. When $B = \emptyset$, we of course obtain (unconditional) mutual information. A structure that maximizes the mutual information between C and $\mathbf{X}$ is one that will lead to the best approximation of the posterior probability. In other words, an ideal form of optimization would do the following:

$$\mathcal{B}^* \in \operatorname*{argmax}_{\mathcal{B}\in\mathcal{F}_\mathcal{B}} I_\mathcal{B}(\mathbf{X}_{1:N};C),$$

where $\mathcal{F}_\mathcal{B}$ is a complexity constrained class of BNs (e.g., k-trees), and $\mathcal{B}^*$ is an optimum network. Of course, this procedure being intractable, we use mutual information to produce efficient heuristics but that we show work well in practice on a wide variety of tasks (Section 6). The third heuristic we describe is similar to the first two except that it is based directly on CR (i.e., empirical error or 0/1-loss) itself. The heuristics detailed in the following are compared against *random orderings* (RO) of the attributes in Section 6 to show that they are doing better than chance.

1: OMI: Our initial approach to finding an order is a greedy algorithm that first chooses the attribute node that is most informative about C. The next attribute in the order is the attribute node that is most informative about C conditioned on the first attribute, and subsequent nodes are chosen to be most informative about C conditioned on previously chosen attribute nodes. More specifically, our algorithm forms an ordered sequence of attribute nodes $\mathbf{X}_\prec^{1:N} = \left\{X_\prec^1, X_\prec^2, \ldots, X_\prec^N\right\}$ according to

$$X_\prec^j \leftarrow \operatorname*{argmax}_{X\in\mathbf{X}_{1:N}\setminus\mathbf{X}_\prec^{1:j-1}} \left[I\left(C;X|\mathbf{X}_\prec^{1:j-1}\right)\right], \tag{2}$$

where $j \in \{1,\ldots,N\}$.

It is not possible to describe the motivation for this approach without considering at least the general way parents of each attribute node are ultimately selected—more details are given below, but for now it is sufficient to say that each node's set of potential parents is restricted to come from nodes earlier in the ordering. Let $\mathbf{X}_{\Pi_j} \subseteq \mathbf{X}_\prec^{1:j-1}$ be the set of chosen parents for X_j in an ordering. There are several reasons why the above ordering should be useful. First, suppose we consider two potential next variables X_{j_1} and X_{j_2} as the j^{th} variable in the ordering, where $I(X_{j_1};C|\mathbf{X}_{1:j-1}) \ll I(X_{j_2};C|\mathbf{X}_{1:j-1})$. Choosing X_{j_1} could potentially lead to the case that no additional variable within the allowable set of parents $\mathbf{X}_{1:j-1}$ could be beneficially added to the model as a parent of X_{j_1}. The reason is that, conditioning on all of the potential parents of X_{j_1}, the variable X_{j_1} is less informative about C. If X_{j_2} is chosen, however, then there is a possibility that some edge augmentation as parents of X_{j_2} will render X_{j_2} residually informative about C—the reason for this is that X_{j_2} chosen to have this property, and one set of parents that renders X_{j_2} residually informative about C is the set $\mathbf{X}_\prec^{1:j-1}$. Stated more concisely, we wish to choose as a next variable in the ordering one that has the

potential to be strongly and residually predictive of C when choosing earlier variables as parents. When choosing X_j such that $I\left(C; X_j | \mathbf{X}_{\prec}^{1:j-1}\right)$ is large, this is possible at least in the case when X may have up to $j-1$ additional parents.

Of course, only a subset of these nodes will ultimately be chosen to ensure that the model is a k-tree and remains tractable and just because $I(X_j; C|\mathbf{X}_{1:j-1})$ is large does not necessarily mean that $I(X_j; C|\mathbf{X}_B)$ is also large for some $B \subset \{1, \ldots, (j-1)\}$. The strict sub-set relationship, where $|B| < (j-1)$, is necessary to restrict the complexity class of our models, but this goal involves an accuracy-computation tradeoff. Our approach, therefore, is only a heuristic. Nevertheless, one justification for our ordering heuristic is based on the aspect of our algorithm that achieves computational tractability, namely the parent-selection strategy where variables are only allowed to have previously ordered variables as their parents (as we describe in more detail below). Moreover, we have empirically found this property to be the case in both real and artificial random data (see below). Loosely speaking, we see our ordering as somewhat analogous to Ada-boost but applied to feature selection, where later decisions on the ordering are chosen to correct for the deficiencies of earlier decisions.

A second reason our ordering may be beneficial stems from the reason that a naive Bayes model is itself useful. In a NB, we have that each X_i is independent of X_j given C. This has beneficial properties both from the bias-variance (Friedman et al., 1997) and from the discriminative structure perspective (Bilmes, 2000). In any given ordering, variables chosen earlier in the order have more of a chance *in the resulting model* to render later variables conditionally independent of each other conditioned on both C and the earlier variable. For example, if two later variables both ask for the same earlier single parent, the two later variables are modeled as independent given C and that earlier parent. This normally would not be useful, but in our ordering, since the earlier variables are in general more correlated with C, this mimics the situation in NB: C and variables similar to C render conditionally independent other variables that are less similar to C (with NB alone, C renders all other variables conditionally independent). For reasons similar to NB (Friedman et al., 1997), such an ordering will tend to work well.

Our approach rests on being able to compute CMI queries over a large number of variables, something that requires both solving a potentially difficult inference problem and also is sensitive to training-data sparsity. In our case, however, a conditional mutual information query can be computed efficiently by making only one pass over the training data, albeit with a potential problem with bias and variance of the mutual information estimate. As mentioned above, each CMI query can be represented as a sum of signed entropy terms. Moreover, since all variables are discrete in our studies, an entropy query can be obtained in one pass over the data by computing an empirical histogram of random variable values only that exist in the data, then summing over only the resulting non-zero values. Let us assume, for simplicity, that integer variable Y represents the Cartesian product of all possible values of the vector random variable for we wish to obtain an entropy value. Normally, $H(Y) = -\sum_y p(y) \log p(y)$ would require an exponential number of terms, but we avoid this by computing $H(Y) = -\sum_{y \in \mathcal{T}_y} p(y) \log p(y) - |\mathcal{D}_y \setminus \mathcal{T}_y| \varepsilon \log \varepsilon$, where $\mathcal{T}_y$ are the set of y values that occur in the training data, and $\mathcal{D}_y$ is the set of all possible y values, and ε is any smoothing value that we might use to fill in zeros in the empirical histogram. Therefore, if our algorithm requires only a polynomial number of CMI queries, then the complexity of the algorithm is still only polynomial in the size of the training data. Of course, as the number of actual variables increases, the quality of this estimate decreases. To mitigate these problems, we can restrict the number of variables in $\mathbf{X}_{\prec}^{1:j-1}$ for a CMI query, leading to the following second heuristic based on CMI.

2: OMISP: For a 1-tree each variable $X_{\prec}^{j}$ has one single parent (SP) X_{Π_j} which is selected from the variables $\mathbf{X}_{\prec}^{1:j-1}$ appearing before $X_{\prec}^{j}$ in the ordering (i.e., $|\Pi_j| = 1, \forall j$). This leads to a simple variant of the above, where CMI conditions only on a single variable within $\mathbf{X}_{\prec}^{1:j-1}$. Under this heuristic, an ordered sequence is determined by

$$X_{\prec}^{j} \leftarrow \operatorname*{argmax}_{X \in \mathbf{X}_{1:N} \setminus \mathbf{X}_{\prec}^{1:j-1}} \left[\max_{X_{\prec} \in \mathbf{X}_{\prec}^{1:j-1}} \left[I\left(C; X | X_{\prec}\right) \right] \right].$$

Note, in this work, we do not present results using OMISP since the results were not significantly different than OMI. We refer the interested reader to Pernkopf and Bilmes (2008a) which gives the results for this heuristic, and more, in their full glory.

3: OCR: Here, CR on the training data is used in a way similar to the aforementioned greedy OMI approach. The ordered sequence of nodes $\mathbf{X}_{\prec}^{1:N}$ is determined according to

$$X_{\prec}^{j} \leftarrow \operatorname*{argmax}_{X \in \mathbf{X}_{1:N} \setminus \mathbf{X}_{\prec}^{1:j-1}} CR\left(\mathcal{B}_{\mathcal{S}} | \mathcal{S}\right),$$

where $j \in \{1, \ldots, N\}$ and the graph of $\mathcal{B}_{\mathcal{S}}$ at each evaluation is a fully connected sub-graph only over the nodes C, X, and $\mathbf{X}_{\prec}^{1:j-1}$, that is, we have a *saturated* sub-graph (note, here $\mathcal{B}_{\mathcal{S}}$ depends on the current X and the previously chosen attribute nodes in the order, but this is not indicated for notational simplicity). This can of course lead to very large local conditional probability tables. We thus perform this computation by using sparse probability tables that have been slightly smoothed as described above. We then compute CR on the basis of $P\left(C | X, \mathbf{X}_{\prec}^{1:j-1}\right) \propto P\left(X, \mathbf{X}_{\prec}^{1:j-1} | C\right) P\left(C\right)$. The justification for this approach is that it produces an ordering based not on mutual information but on a measure more directly related to classification accuracy.

5.2 Step 2: Heuristic for Selecting Parents w.r.t. a Given Order to Form a k-tree

Once we have the ordering $\mathbf{X}_{\prec}^{1:N}$, we select $X_{\Pi_j} \subseteq \mathbf{X}_{\prec}^{1:j-1}$ for each $X_{\prec}^{j}$, with $j \in \{2, \ldots, N\}$. When the size of $\mathbf{X}_{\Pi_j}$ (i.e., N) and of k are small we can use even a computational costly scoring function to find X_{Π_j}. In case of a large N, we can restrict the size of the parent set $\mathbf{X}_{\Pi_j}$ similar to the *sparse candidate algorithm* (Friedman et al., 1999). While either the CL or the CR can be used as a cost function for selecting parents, we in this work restrict our experiments to CR for parent selection (empirical results show it performed better). The parent selection proceeds as follows. For each $j \in \{2, \ldots, N\}$, we choose the best k parents $X_{\Pi_j} \subseteq \mathbf{X}_{\prec}^{1:j-1}$ for $X_{\prec}^{j}$ by scoring each of the $O\left(\binom{N}{k}\right)$ possibilities with CR. We note that for $j \in \{2, \ldots, N-1\}$ there will be a set of variables that have not yet had their parents chosen, namely variables $\mathbf{X}_{\prec}^{j+1:N}$—for these variables, we simply use the NB assumption. That is, those variables have no parents other than C for the selection of parents for $X_{\prec}^{j}$ (we relax this property in Pernkopf and Bilmes, 2008a). Note that the set of parents is judged using CR, but the model parameters for any given candidate set of parents selected are trained using ML (we did not find further advantages, in addition to using CR for parent selection, in also using discriminative parameter training). We also note that the parents for each attribute node are retained in the model only when CR is improved, and otherwise the node $X_{\prec}^{j}$ is left parent-less. This therefore might result in a partial k-tree (forest) over the attributes. We evaluate our algorithm for $k = 1$ and $k = 2$, but is defined above to learn k-trees ($k \geq 1$), and thus uses $O\left(N^{k+1}\right)$ score evaluations where,

due to ML training, each CR evaluation is $O(NM)$. Overall, for learning a 1-tree, the ordering and the parent selection costs $O\left(N^2\right)$ score evaluations. We see that the computation is comparable to that of the *SuperParent* algorithm and its *k*-tree generalization.

Algorithm 1 OMI-CR

Input: $\mathbf{X}_{1:N}, C, \mathcal{S}$
Output: set of edges $\mathbf{E}$ for TAN network
$X^1_{\prec}, X^2_{\prec} \leftarrow \text{argmax}_{X,X' \in \mathbf{X}_{1:N}} \left[I(C;X,X') \right]$
if $I\left(C;X^1_{\prec}\right) < I\left(C;X^2_{\prec}\right)$ **then**
  $X^2_{\prec} \leftrightarrow X^1_{\prec}$
end if
$\mathbf{E} \leftarrow \left\{ \mathbf{E}_{\text{Naive Bayes}} \cup E_{X^1_{\prec}, X^2_{\prec}} \right\}$
$j \leftarrow 2$
$CR_{old} \leftarrow 0$
repeat
  $j \leftarrow j+1$
  $X^j_{\prec} \leftarrow \text{argmax}_{X \in \mathbf{X}_{1:N} \setminus \mathbf{X}^{1:j-1}_{\prec}} \left[I\left(C;X|\mathbf{X}^{1:j-1}_{\prec}\right) \right]$
  $X^*_{\prec} \leftarrow \text{argmax}_{X \in \mathbf{X}^{1:j-1}_{\prec}} CR(\mathcal{B}_{\mathcal{S}}|\mathcal{S})$ where edges of $\mathcal{B}_{\mathcal{S}}$ are $\left\{ \mathbf{E} \cup E_{X, X^j_{\prec}} \right\}$
  $CR_{new} \leftarrow CR(\mathcal{B}_{\mathcal{S}}|\mathcal{S})$ where edges of $\mathcal{B}_{\mathcal{S}}$ are $\left\{ \mathbf{E} \cup E_{X^*_{\prec}, X^j_{\prec}} \right\}$
  if $CR_{new} > CR_{old}$ **then**
    $CR_{old} \leftarrow CR_{new}$
    $\mathbf{E} \leftarrow \left\{ \mathbf{E} \cup E_{X^*_{\prec}, X^j_{\prec}} \right\}$
  end if
until $j = N$

5.3 OMI-CR Algorithm

Recapitulating, we have introduced three order-based greedy heuristics for producing discriminative structures in Bayesian network classifiers: First, there is OMI-CR (Order based on Mutual Information with CR used for parent selection); Second, there is OMISP-CR (Order based on Mutual Information conditioned on a Single Parent, with CR used for parent selection); and third OCR-CR (Order based on Classification Rate, with CR used for parent selection). For evaluation purposes, we also consider random orderings in step 1 and CR for parent selection (RO-CR). The OMI-CR procedure is summarized in Algorithm 1 where both steps (order and parent selection) are merged at each loop iteration (which is of course equivalent to considering both steps separately). The different algorithmic variants are obtained by modifying the ordering criterion.

6. Experiments

We present classification results on 25 data sets from the UCI repository (Merz et al., 1997) and from Kohavi and John (1997), for frame- and segment-based phonetic classification experiments using the TIMIT database (Lamel et al., 1986), for a visual surface inspection task (Pernkopf, 2004), and for handwritten digit recognition using the MNIST (LeCun et al., 1998) and USPS data set.

Additionally, we show performance results on synthetic data. We use NB, TAN, and 2-tree network structures. Different combinations of the following parameter/structure learning approaches are used to learn the classifiers:

- Generative (ML) (Pearl, 1988) and discriminative (CL) (Greiner et al., 2005) parameter learning.
- CMI: Generative structure learning using CMI as proposed in Friedman et al. (1997).
- CR: Discriminative structure learning with greedy heuristic using CR as scoring function (Keogh and Pazzani, 1999; Pernkopf, 2005) (see Section 4.4).
- RO-CR: Discriminative structure learning using random ordering (RO) in step 1 and CR for parent selection in step 2 of the order-based heuristic.
- SuperParent k-tree: Discriminative structure learning using the SuperParent algorithm (Keogh and Pazzani, 1999) with $k = 1, 2$ (see Section 4.4).
- OMI-CR: Discriminative structure learning using CMI for ordering the variables (step 1) and CR for parent selection in step 2 of the order-based heuristic.
- For OMI-CR, we also evaluate discriminative parameter learning by optimizing CL during the selection of the parent in step 2. We call this OMI-CRCL. Discriminative parameter learning while optimizing discriminative structure is computationally feasible only on rather small data sets due to the cost of the conjugate gradient parameter optimization.

We do not include experimental results for OMISP-CR and OCR-CR for space reasons. The results, however, show similar performance to OMI-CR, and can be found in an extended technical-report version of this paper (Pernkopf and Bilmes, 2008a).

While we have attempted to avoid a proliferation of algorithm names, some name abundance has unavoidably occurred in this paper. We therefore have attempted to use a simple 2-, 3-, or even 4-tag naming scheme where A-B-C-D is such that "A" (if given) refers to either TAN (1-tree) or 2-tree, "B" and "C" refer to the structure learning approach, and "D" (if given) refers to the parameter training method of the *final* resultant model structure. For the ordering heuristics "B" refers to the ordering method, "C" refers to the parent selection and internal parameter learning strategy. For the remaining structure learning heuristics only "B" is present. Thus, TAN-OMI-CRML-CL would be the OMI procedure for ordering, parent selection evaluated using CR (with ML training used at that time), and with CL used to train the final model which would be a 1-tree (note moreover that TAN-OMI-CR-CL is equivalent since ML is the default training method).

6.1 Experimental Setup

Any continuous features were discretized using recursive minimal entropy partitioning (Fayyad and Irani, 1993) where the codebook is produced using only the training data. This discretization method uses the class entropy of candidate partitions to determine the bin boundaries. The candidate partition with the minimal entropy is selected. This is applied recursively on the established partitions and the minimum description length approach is used as stopping criteria for the recursive partitioning. In Dougherty et al. (1995), an empirical comparison of different discretization methods has been performed and the best results have been achieved with this entropy-based discretization. Throughout our experiments, we use exactly the same data partitioning for each training procedure. We performed simple smoothing, where zero probabilities in the conditional probability tables are replaced with small values ($\varepsilon = 0.00001$). For discriminative parameter learning, the parameters are

initialized to the values obtained by the ML approach (Greiner et al., 2005). The gradient descent parameter optimization is terminated using *cross tuning* as suggested in Greiner et al. (2005).

6.2 Data Characteristics

In the following, we introduce several data sets used in the experiments.

6.2.1 UCI DATA

We use 25 data sets from the UCI repository (Merz et al., 1997) and from Kohavi and John (1997). The same data sets, 5-fold cross-validation, and train/test learning schemes as in Friedman et al. (1997) are employed. The characteristics of the data sets are summarized in Table 7 in the Appendix A.

6.2.2 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. The speech is sampled at 16 kHz. Speech frames are classified into the following classes, voiced (V), unvoiced (U), silence (S), mixed sounds (M), voiced closure (VC), and release (R) of plosives. We therefore are performing frame-by-frame phone classification (contrasted with phone *recognition* using, say, a hidden Markov model). We perform experiments with only four classes V/U/S/M and all six classes V/U/S/M/VC/R using 110134 and 121629 samples, respectively. The class distribution of the four class experiment V/U/S/M is 23.08%, 60.37%, 13.54%, 3.01% and of the six class case V/U/S/M/VC/R is 20.9%, 54.66%, 12.26%, 2.74%, 6.08%, 3.36%. Additionally, we perform classification experiments on data of male speakers (Ma), female speakers (Fe), and both genders (Ma+Fe). For each gender we have approximately the same number of samples. The data have been split into 2 mutually exclusive subsets of $\mathcal{D} \in \{\mathcal{S}_1, \mathcal{S}_2\}$ where the size of the training data $\mathcal{S}_1$ is 70% and of the test data $\mathcal{S}_2$ is 30% of $\mathcal{D}$. The classification experiments have been performed with 8 wavelet-based features combined with 12 mel-frequency cepstral coefficients (MFCC) features, that is, 20 features. More details about the features can be found in Pernkopf et al. (2008). We have 6 different classification tasks for each classifier, that is, Ma+Fe, Ma, Fe $\times$ 4 or 6 Classes.

6.2.3 TIMIT-39 DATA

This again is a phone classification test but with a larger number of classes. In accordance with Halberstadt and Glass (1997) we cluster the 61 phonetic labels 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. Each speaker speaks 10 sentences including two sentences which are the same among all speakers (labeled as *sa*), five sentences which were read from a list of phonetically balanced sentences (labeled as *sx*), and 3 randomly selected sentences (labeled as *si*). In the experiments, we only use the *sx* and *si* sentences since the *sa* sentences introduce a bias for certain phonemes in a particular context. This means that we have 3696 and 1344 utterances in the training and test set, respectively. We derive from each phonetic segment one feature vector which results in 140173 training samples and 50735 testing samples. The features are derived similarly as proposed in Halberstadt and Glass (1997). First, 12 MFCC + log-energy feature (13 MFCC's) and their derivatives (13 Derivatives) are calculated for

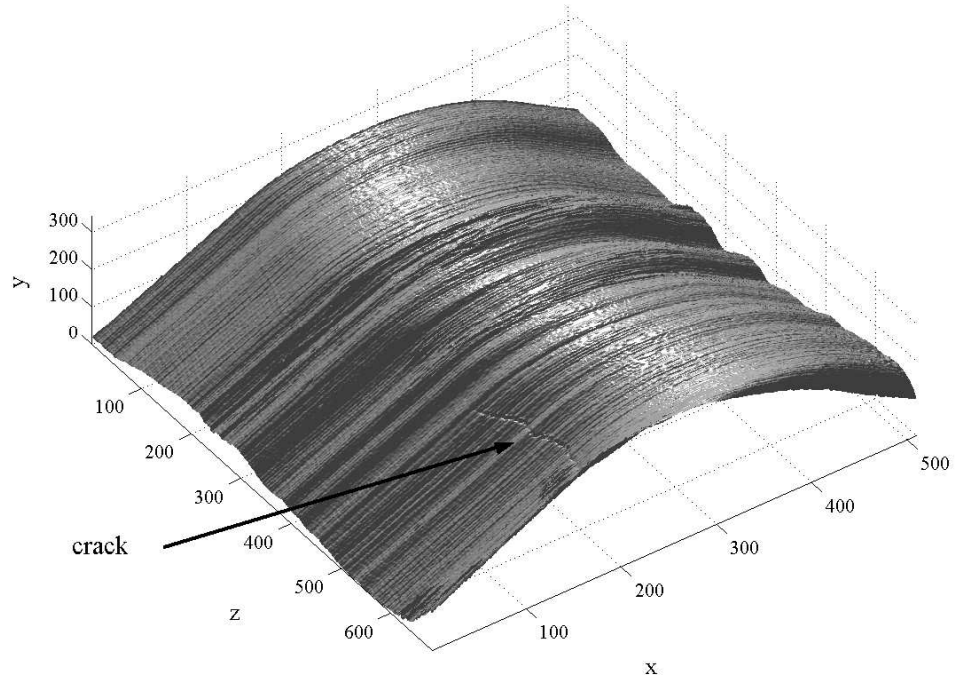

Figure 5: Acquired surface data with an embedded crack.

every 10ms of the utterance with a window size of 25ms. A phonetic segment, which can be variable length, is split at a 3:4:3 ratio into 3 parts. The fixed-length feature vector is composed of: 1) three averages of the 13 MFCC's calculated from the 3 portions (39 features); 2) the 13 Derivatives of the beginning of the first and the end of the third segment part (26 features); and 3) the log duration of the segment (1 feature). Hence, each phonetic segment is represented by 66 features.

6.2.4 SURFACE INSPECTION DATA (SURFINSP)

This data set was acquired from a surface inspection task. Surface defects with three-dimensional characteristics on scale-covered steel blocks have to be classified into 3 classes. The 3-dimensional raw data showing the case of an embedded surface crack is given in Figure 5. The data set consists of 450 surface segments uniformly distributed into three classes. Each sample (surface segment) is represented by 40 features. More details on the inspection task and the features used can be found elsewhere (Pernkopf, 2004).

6.2.5 MNIST DATA

We evaluate our classifiers on the MNIST data set of handwritten digits (LeCun et al., 1998) which contains 60000 samples for training and 10000 digits for testing. The digits are centered in a 28×28 gray-level image. We re-sample these images at a resolution of 14×14 pixels. This gives 196 features where the gray-levels are discretized using the procedure from Fayyad and Irani (1993).

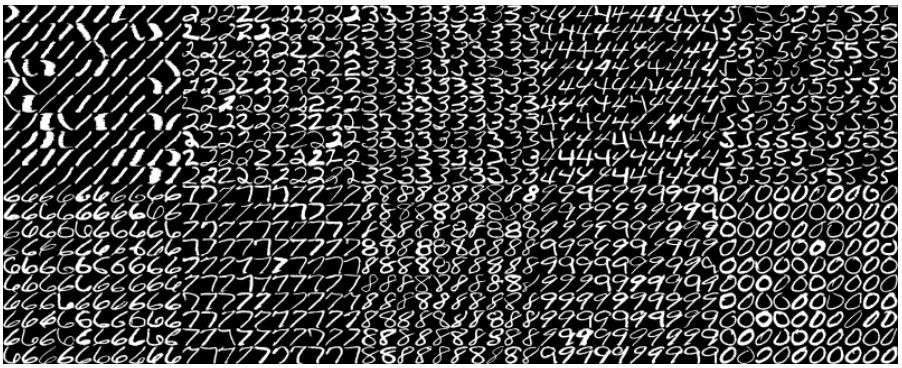

Figure 6: USPS data.

6.2.6 USPS DATA

This data set contains 11000 handwritten digit images (uniformly distributed) collected from zip codes of mail envelopes. Each digit is represented as a 16×16 grayscale image, where each pixel is considered as individual feature. Figure 6 shows a random sample of the data set. We use 8000 digits for training and the remaining images as a test set.

6.3 Conditional Likelihood and Maximum Mutual Information Orderings

In the following, we evaluate the ordering heuristics using 31 different classification scenarios (from the UCI and the TIMIT-4/6 data sets) comprising differing input features and differing numbers of classes. We compare our ordering procedure (i.e., OMI, where we maximize the mutual information as in Equation 2) with several other possible orderings in an attempt to empirically show that our aforementioned intuition regarding order (see Section 5.1) is sound in the majority of cases. In particular, we compare against an ordering produced by minimizing the mutual information (replacing argmax with argmin in Equation 2). Additionally, we also compare against 100 uniformly-at-random orderings. For the selection of the conditioning variables (see Section 5.2) the CL score is used in each case. ML parameter estimation is used for all examples in this section.

Figure 7 and Figure 8 show the resulting conditional log likelihoods (CLL) of the model scoring the training data after the TAN network structures (1-trees in this case) have been determined for the various data sets. As can be seen, our ordering heuristic performs better than both the random and the minimum mutual information orderings on 28 of the 31 cases. The random case shows the mean and $\pm$ one standard deviation out of 100 orderings. For *Corral*, *Glass*, and *Heart* there is no benefit, but the data sets are on the smaller side where it is less unexpected that generative structure learning would perform better (Ng and Jordan, 2002).

To further verify that our ordering tends to produce better conditional likelihood values on training data, we also evaluate on random data. For each number of variables (from 10 up to 14) we generate 1000 random distributions and draw 100000 samples from each one. Using these samples, we learn generative and discriminative TAN structures by the following heuristics and report the resulting conditional log likelihood on the training data: (i) order variables by maximizing mutual information (TAN-OMI-CL), (ii) order variables by minimizing mutual information, (iii) random ordering of variables and CL parent selection (TAN-RO-CL), (iv) optimal generative 1-tree, that

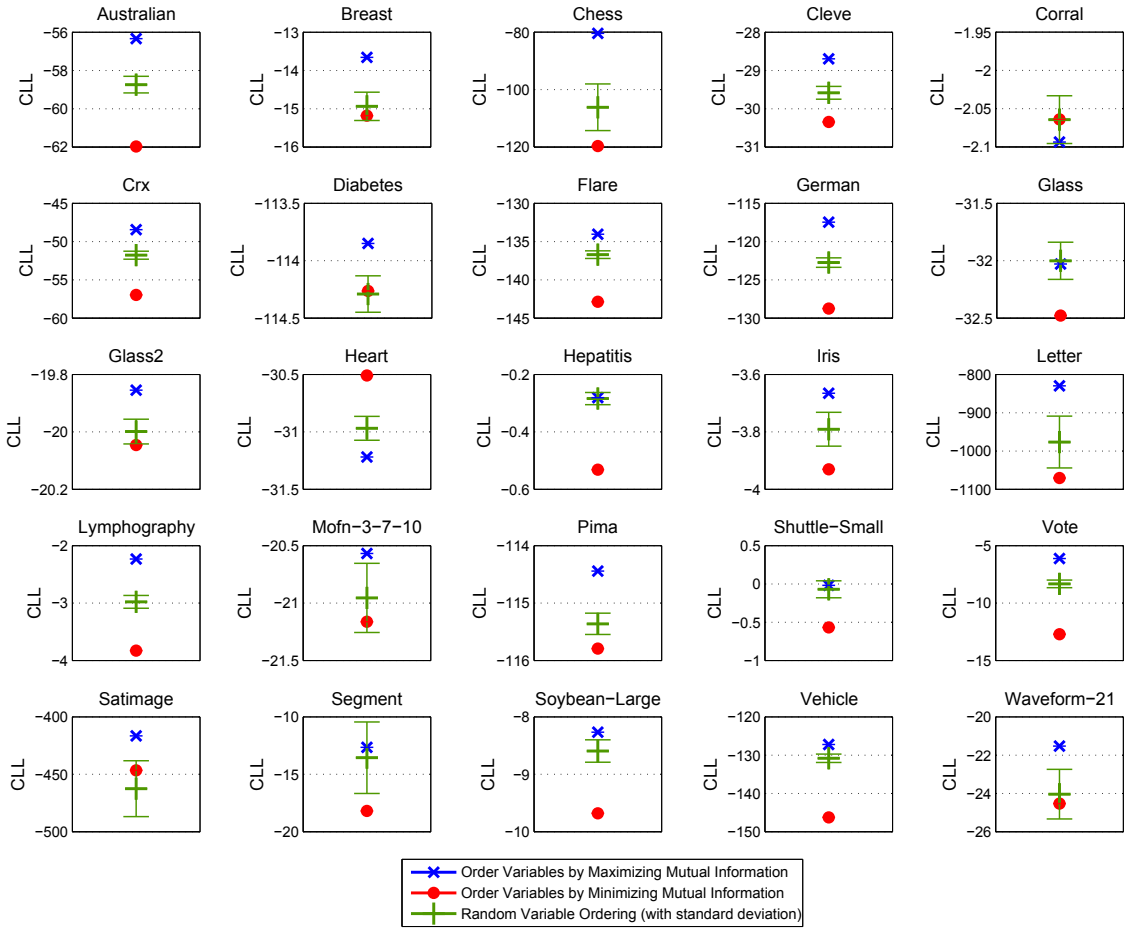

Figure 7: Resulting CLL on the UCI data sets for a maximum mutual information (i.e., OMI), a minimum mutual information, and a random based ordering scheme.

is, TAN-CMI (Friedman et al., 1997), (v) the computationally expensive greedy heuristic using CL (see Section 4.4.1), what we call TAN-CL. In addition we show CLL results for the NB classifier.

Figure 9 shows the CLL values for various algorithms. The CLL is still high even with the much less computationally costly OMI-CL procedure. Additionally, the generative 1-tree method improves likelihood but it does not necessarily produce good conditional likelihood results. We performed a one-sided paired t-test (Mitchell, 1997) for all different structure learning approaches. This test indicates that the CLL differences among the methods are significant at a level of 0.05 for each number of variables. This figure shows that the CLL gets smaller when more attributes are involved. With increasing the number of variables the random distribution becomes more *complex* (i.e., the number of dependencies among variables increases). However, we approximate the true distribution in any case with a 1-tree.

While we have shown empirically that our ordering heuristic tends to produce models that score the training data highly in the conditional likelihood sense, a higher conditional likelihood does not guarantee a higher accuracy, and training data results do not guarantee good generalization. In the

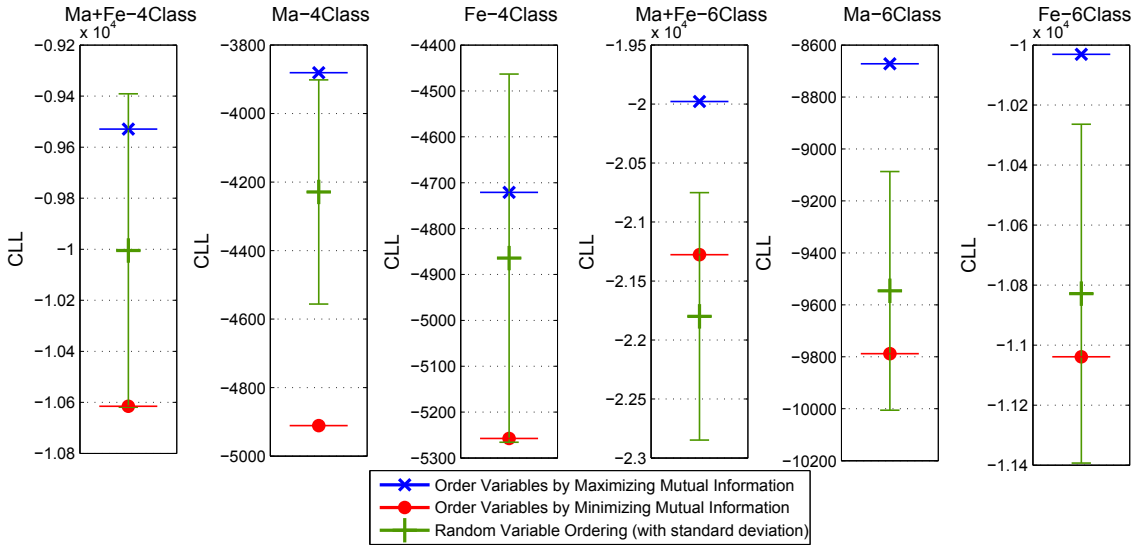

Figure 8: Resulting CLL on the TIMIT-4/6 data sets for a maximum mutual information (i.e., OMI), a minimum mutual information, and a random based ordering scheme.

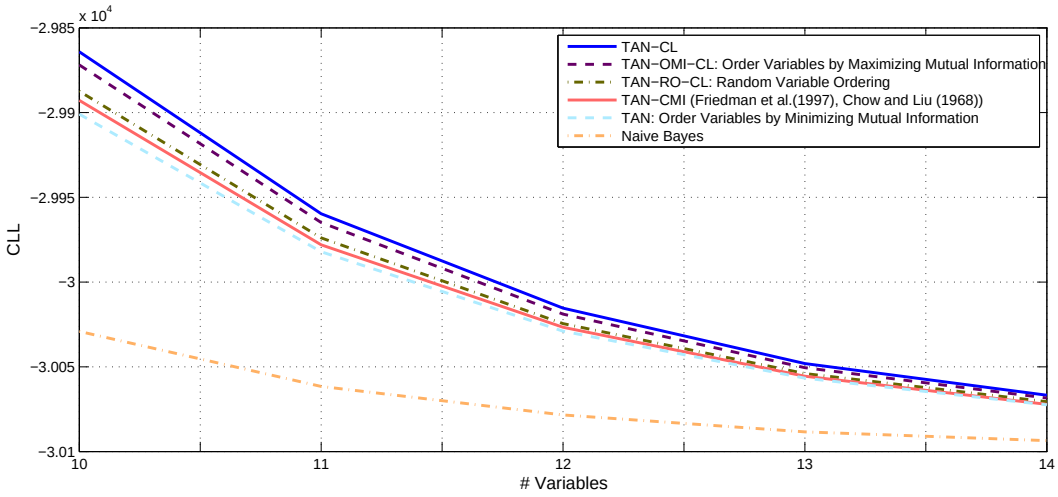

Figure 9: Optimized CLL of TAN structures learned by various algorithms. For each number of variables (x-axis) we generated 1000 random distributions.

next sections, however, we show that on balance, accuracy on test data using our ordering procedure is on par with the expensive greedy procedure, but with significantly less computation.

6.4 Synthetic Data

We show the benefit of the structure learning algorithms for the case where the class-dependent data are sampled from different 1-tree structures. In particular, we randomly determine for each class a 1-tree. The probabilities for each attribute variable are sampled from an uniform distribution, whereas the cardinality is set to 10, that is, $|X_i| = 10$. We use five classes. From the tree for $C = 1$ we draw 25000 samples. Additionally, we sample 6250 samples for each of the remaining four

classes from the same structure for confusion. For the remaining classes we draw 6250 samples from the corresponding random trees. This gives in total 75000 samples for training. The test set also consists of 75000 samples generated likewise. We perform this experiment for varying number of attributes, that is, $N \in \{5, 10, 15, 20, 25, 30\}$. The recognition results are shown in Figure 10, whereas the performance of each algorithm is averaged over 20 independent runs with randomly selected conditional probability distributions and trees. In each run, all algorithms have exactly the same data available.

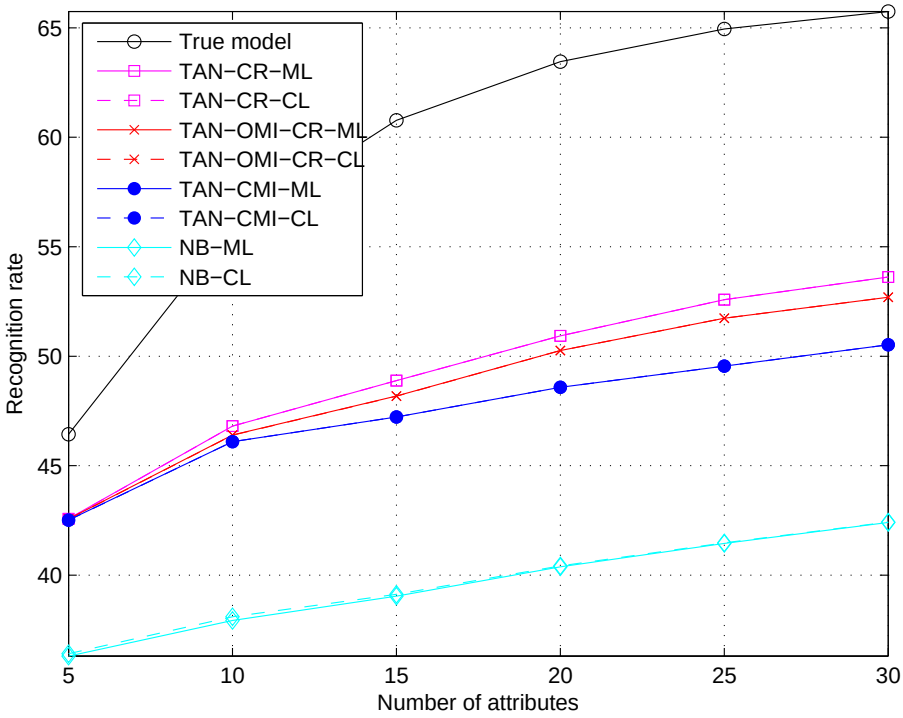

Figure 10: Synthetic data: Recognition performance is averaged over 20 runs.

We compare our OMI-CR heuristic to greedy discriminative structure learning. Additionally, we provide results for NB and generatively optimized TAN structures using CMI. To give a flavor about the data, the classification rates achieved with the true model used to generate the data are reported. This figure indicates that OMI-CR performs slightly worse than the greedy heuristic. However, the one-sided paired t-test (Mitchell, 1997) indicates that TAN-OMI-CR performs significantly better than TAN-CMI for more than 5 attributes at a level of 0.05. Generally, discriminative parameter optimization (i.e., CL) does not help for this data.

6.5 Classification Results and Discussion

Table 1 presents the averaged classification rates over the 25 UCI and 6 TIMIT-4/6 data sets.[4] Additionally, we report the CR on TIMIT-39, SurfInsp, MNIST, and USPS. The individual classification performance of all classification approaches on the 25 UCI data sets are summarized in Pernkopf

4. The average CR is determined by first 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.

DATA SET / CLASSIFIER	UCI	TIMIT-4/6	TIMIT-39	SURFINSP	MNIST	USPS
NB-ML	81.50	84.85	61.70± 0.22	89.11 ± 1.47	83.73 ± 0.37	87.10 ± 0.61
NB-CL	85.18	**88.69**	**70.33** ± 0.20	92.67 ± 0.90	91.70 ± 0.28	93.67 ± 0.44
TAN-CMI-ML	84.82	86.18	65.40 ± 0.21	92.44 ± 0.96	91.28 ± 0.28	91.90 ± 0.50
TAN-CMI-CL	85.47	87.22	66.31 ± 0.21	92.44 ± 0.96	**93.80** ± 0.24	94.87 ± 0.40
TAN-RO-CR-ML **(MEAN)**	85.04	87.43	-	93.13 ± 0.70	-	-
TAN-RO-CR-ML **(MIN)**	85.00	87.57	-	92.67	-	-
TAN-RO-CR-ML **(MAX)**	84.82	87.43	-	92.67	-	-
TAN-SUPERPARENT-ML	84.80	87.54	66.53 ± 0.21	92.22 ±0.78	91.80 ± 0.27	90.67 ± 0.53
TAN-SUPERPARENT-CL	85.70	87.76	66.56 ± 0.21	92.44 ± 0.96	93.50 ± 0.25	94.70 ± 0.41
TAN-OMI-CR-ML	85.11	87.52	66.61± 0.21	**94.00** ±1.14	92.01 ± 0.27	92.40 ± 0.48
TAN-OMI-CR-CL	**85.82**	87.54	66.87 ± 0.21	**94.00** ±1.14	93.39 ± 0.25	94.90 ± 0.40
TAN-OMI-CRCL-ML	85.16	87.46	-	**94.22** ±1.13	-	-
TAN-OMI-CRCL-CL	85.78	87.62	-	**94.22** ±1.13	-	-
TAN-CR-ML	85.38	87.62	66.78 ± 0.21	92.89 ± 0.57	92.58 ± 0.26	92.57 ± 0.48
TAN-CR-CL	**86.00**	87.48	67.23 ± 0.21	92.89 ± 0.57	**93.94** ± 0.24	**95.83** ± 0.36
2-TREE-RO-CR-ML (MEAN)	-	87.86	-	-	-	-
2-TREE-RO-CR-ML (MIN)	-	87.87	-	-	-	-
2-TREE-RO-CR-ML (MAX)	-	87.87	-	-	-	
2-TREE-SUPERPARENT-ML	84.77	87.33	64.78 ± 0.21	92.67 ±1.63	90.56 ± 0.29	90.67 ± 0.53
2-TREE-SUPERPARENT-CL	**85.90**	87.14	67.38 ± 0.21	92.67 ±1.63	92.47 ± 0.26	94.13 ± 0.43
2-TREE-OMI-CR-ML	85.50	88.01	66.94 ± 0.21	**94.22** ± 0.82	92.69 ± 0.26	94.03 ± 0.41
2-TREE-OMI-CR-CL	**85.81**	87.27	67.06 ± 0.21	**94.88** ± 0.90	93.09 ± 0.25	94.76 ± 0.41
2-TREE-CR-ML	85.53	87.94	66.71 ± 0.21	**94.22** ± 1.07	-	-
2-TREE-CR-CL	85.73	86.95	67.36 ± 0.21	**94.22** ± 1.07	-	-

Table 1: Averaged classification results for 25 UCI and 6 TIMIT-4/6 data sets and classification results for TIMIT-39, SurfInsp, MNIST, and USPS with standard deviations. Best results use bold font. ML and CL denote generative and discriminative parameter learning, respectively. The order-based greedy heuristics are OMI-CR (order mutual information-CR) and RO-CR (random order-CR). CRCL refers to using discriminative parameter learning during structure learning. The generative structure learning algorithm is abbreviated as CMI and the greedy discriminative structure learning is TAN-CR and 2-tree-CR.

and Bilmes (2008a), whereas the random order experiment is presented in Table 8 (see Appendix A). For the TIMIT-4/6 data sets the individual classification performances for various classifier learning methods can be found in Table 9 (see Appendix B). The random order experiment for these data sets using a 2-tree is summarized in Appendix B in Table 10.

6.5.1 DISCUSSION: DISCRIMINATIVE VERSUS GENERATIVE PARAMETER LEARNING

Discriminative parameter learning 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 (Greiner et al., 2005)—the average improvement of discriminative parameter learning over ML estimation on NB and generative TAN-CMI structures is large. In particular, for TIMIT-4/6 and TIMIT-39 the discriminatively optimized NB classifier (i.e., NB-CL) achieves the overall best classification performance. One reason is that the final step of MFCC features extraction includes a discrete cosine transform, that is, the features are decorrelated. Hence, the independence assumptions of the NB structure might be a good choice for these data sets. A second reason is that CL parameter learning for the TAN and 2-tree structures overfit the data (even with cross tuning)—the NB structure implicitly keeps the number of parameters low. Analyzing the results of CL parameter learning over various structures (especially for

2-trees) reveal that *cross tuning* is for some cases too restrictive concerning the number of conjugate gradient iterations—an alternative regularization method is required. CL parameter learning of NB classifiers is known to be equivalent to logistic regression. It can be shown that the CLL is concave when using $\log \theta^j_{i|h}$, that is, the global maximum can be found during discriminative parameter learning. Roos et al. (2005) showed that this also holds for more general network structures, for example, TAN.

In Section 6.6, we show that the classification performance of a discriminatively structured model may be superior to discriminatively parameterized models in the case of missing features.

6.5.2 DISCUSSION: DISCRIMINATIVE VERSUS GENERATIVE STRUCTURE LEARNING USING ML PARAMETER LEARNING

The CR objective function produces the best performing network structures. Evaluation of the CR measure is computationally expensive as mentioned above. However, due to the ordering of the variables in the order-based heuristics, we can reduce the number of CR evaluations from $O\left(N^3\right)$ to $O\left(N^2\right)$ for TAN structures. Hence, TAN-CR and 2-tree-CR are restricted to rather small data sets. The order-based heuristic OMI-CR achieve a similar performance at a much lower computational cost. Discriminative parameter learning during discriminative structure learning using our order-based heuristics can slightly improve the performance. This is possible only on small data sets due to the computational burden for the conjugate gradient parameter optimization.

The discriminative SuperParent algorithm performs slightly but not significantly worse compared to the other discriminative structure learning algorithms OMI-CR, OMI-CRCL, and greedy heuristic using CR on the UCI data set—similarly, the performance of SuperParent on TIMIT-4/6 for learning TAN structures, however, the performance using 2-tree structures is low. In summary, SuperParent achieves a lower classification rate compared to other discriminative structure learning algorithms on most of the data sets. The main reason for the degraded performance is an early termination of the algorithm.

For RO-CR we summarize the performance over 1000 random orderings using the mean (Mean), minimum (Min), and maximum (Max) CR (we use only 100 random orders for TIMIT-4/6 though). Min (Max) reports the classification rate on the test set using the structure which achieves the minimum (maximum) performance over 1000 random orderings (resp. 100 orders for TIMIT-4/6) on the training data. In some cases, the average over the data sets show that the worst RO-CR structures scored on the training sets perform better on the test sets than the best structures on the training sets, presumably due to overfitting. These results do show, however, that choosing from a collection of arbitrary orders and judging them based on the training set performance is not likely to perform well on the test set. Our heuristics do improve over these orders.

The TIMIT-39, MNIST, and USPS experiments show that we can perform discriminative structure learning for relatively large classification problems ($\sim$140000 samples, 66 features, 39 classes, $\sim$60000 samples, 196 features, 10 classes, and $\sim$8000 samples, 256 features, 10 classes, resp.). For these data sets, OMI-CR significantly outperform NB and TAN-CMI.

On MNIST we achieve a classification performance of $\sim$92.58% with the discriminative TAN classifier. A number of state-of-the-art algorithms, that is, convolutional net and virtual SVM, achieve an error rate below 1% (LeCun and Cortes). Due to resampling we use only 196 features in contrast to the 784 features of the original data set which might explain some of the loss in classification rate. Another reason why the convolutional neural net and virtual SVM perform better

CLASSIFIER STRUCTURE LEARNING PARAMETER LEARNING	TAN RO-CR ML	TAN SUPERPARENT ML	TAN OMI-CR ML	TAN OMI-CRCL ML	TAN CR ML	2-TREE SUPERPARENT ML	2-TREE OMI-CR ML	2-TREE CR ML
	MAX							
NB-ML	$\Uparrow_{0.0300}$	$\Uparrow_{0.0232}$	$\Uparrow_{0.0242}$	$\Uparrow_{0.0203}$	$\Uparrow_{0.0154}$	$\Uparrow_{0.0103}$	$\Uparrow_{0.0316}$	$\Uparrow_{0.0317}$
TAN-CMI-ML	$\uparrow_{0.120}$	$\Leftarrow_{0.122}$	$\Uparrow_{0.0154}$	$\Uparrow_{0.0094}$	$\Uparrow_{0.0141}$	$\Leftarrow_{0.0705}$	$\Uparrow_{0.0271}$	$\Uparrow_{0.0159}$
TAN-RO-CR-ML		$\Leftarrow_{0.197}$	$\uparrow_{0.144}$	$\uparrow_{0.0945}$	$\Uparrow_{0.0446}$	$\Leftarrow_{0.131}$	$\uparrow_{0.148}$	$\uparrow_{0.140}$
TAN-SUPERPARENT-ML			$\uparrow_{0.136}$	$\uparrow_{0.0848}$	$\uparrow_{0.0917}$	$\Leftarrow_{0.189}$	$\uparrow_{0.149}$	$\uparrow_{0.139}$
TAN-OMI-CR-ML				$\uparrow_{0.182}$	$\uparrow_{0.190}$	$\Leftarrow_{0.194}$	$\uparrow_{0.197}$	$\uparrow_{0.196}$
TAN-OMI-CRCL-ML					$\uparrow_{0.197}$	$\Leftarrow_{0.182}$	$\uparrow_{0.196}$	$\uparrow_{0.197}$
TAN-CR-ML						$\Leftarrow_{0.178}$	$\uparrow_{0.194}$	$\uparrow_{0.197}$
2-TREE-SUPERPARENT-ML							$\uparrow_{0.195}$	$\uparrow_{0.192}$
2-TREE-OMI-CR-ML								$\uparrow_{0.195}$

Table 2: Comparison of different classifiers using the one-sided paired t-test for the 25 UCI data sets: Each entry of the table gives the significance of the difference of the classification rate of two classifiers over the data sets. The arrow points to the superior learning algorithm. We use a double arrow if the difference is significant at the level of 0.05. The order-based greedy heuristics are OMI-CR (order mutual information-CR) and RO-CR (random order-CR). CRCL refers to algorithms using discriminative parameter learning during structure learning. The generative structure learning algorithm is abbreviated as CMI and the naive greedy discriminative structure learning is TAN-CR and 2-tree-CR.

CLASSIFIER STRUCTURE LEARN. PARAMETER LEARN.	2-TREE RO-CR ML	TAN SUPERPARENT ML	TAN OMI-CR ML	TAN OMI-CRCL ML	TAN CR ML	2-TREE SUPERPARENT ML	2-TREE OMI-CR ML	2-TREE CR ML
	MAX							
NB-ML	$\Uparrow_{<0.0001}$	$\Uparrow_{<0.0001}$	$\Uparrow_{<0.0001}$	$\Uparrow_{<0.0001}$	$\Uparrow_{<0.0001}$	$\Uparrow_{<0.0001}$	$\Uparrow_{<0.0001}$	$\Uparrow_{<0.0001}$
TAN-CMI-ML	$\Uparrow_{0.00032}$	$\Uparrow_{0.0012}$	$\Uparrow_{0.0016}$	$\Uparrow_{0.0019}$	$\Uparrow_{0.0012}$	$\Uparrow_{0.0027}$	$\Uparrow_{0.0002}$	$\Uparrow_{0.0002}$
2-TREE-RO-CR-ML		$\Leftarrow_{0.0007}$	$\Leftarrow_{0.0024}$	$\Leftarrow_{0.0011}$	$\Leftarrow_{0.0011}$	$\Leftarrow_{0.0010}$	$\Uparrow_{0.0011}$	$\Uparrow_{0.0092}$
TAN-SUPERPARENT-ML			$\leftarrow_{0.189}$	$\leftarrow_{0.151}$	$\uparrow_{0.147}$	$\Leftarrow_{0.0187}$	$\Uparrow_{0.0002}$	$\Uparrow_{0.0006}$
TAN-OMI-CR-ML				$\leftarrow_{0.078}$	$\uparrow_{0.140}$	$\Leftarrow_{0.0078}$	$\Uparrow_{0.0004}$	$\Uparrow_{0.0013}$
TAN-OMI-CRCL-ML					$\Uparrow_{0.038}$	$\leftarrow_{0.054}$	$\Uparrow_{0.0002}$	$\Uparrow_{0.0007}$
TAN-CR-ML						$\Leftarrow_{0.013}$	$\Uparrow_{0.0002}$	$\Uparrow_{0.0010}$
2-TREE-SUPERPARENT-ML							$\Uparrow_{0.0004}$	$\Uparrow_{0.0005}$
2-TREE-OMI-CR-ML								$\leftarrow_{0.069}$

Table 3: Comparison of different classifiers using the one-sided paired t-test for the 6 TIMIT-4/6 data sets: Each entry of the table gives the significance of the difference of the classification rate of two classifiers over the data sets. The arrow points to the superior learning algorithm. We use a double arrow if the difference is significant at the level of 0.05. The order-based greedy heuristics are OMI-CR (order mutual information-CR) and RO-CR (random order-CR). CRCL refers to algorithms using discriminative parameter learning during structure learning. The generative structure learning algorithm is abbreviated as CMI and the naive greedy discriminative structure learning is TAN-CR and 2-tree-CR.

on digit recognition is probably that images are not treated as unstructured feature vectors, that is, the convolutional neural net has built-in parts that look at particular areas of the image, and the virtual SVM is trained on augmented data that reflects invariance to small translations and rotations.

For the SurfInsp data the standard deviation of the five-fold cross-validation classification accuracy estimate is relatively large. Unfortunately, the size of the data set is limited to 450 samples.

The structure of Bayesian networks is implicitly regularized when we restrict the optimization over a model structure (e.g., 1-trees) assuming sufficient training data. For 2-trees we noticed that the data tended to overfit without further regularization. Therefore, we introduce 5-fold cross validation on the *training* data to find the optimal classifier structure.

Table 2 and Table 6.5.2 present a summary of the classification results over all structure learning experiments using ML parameter learning of the UCI and TIMIT-4/6 data sets.

We compare all pairs of classifiers using the one-sided paired t-test (Mitchell, 1997). The t-test determines whether the classifiers differ significantly under the assumption that the classification differences over the data set are independent and identically normally distributed. In these tables, each entry gives the significance of the difference in classification rate of two classification approaches. The arrow points to the superior learning algorithm and a double arrow indicates whether the difference is significant at a level of 0.05.

These tables show that TAN-OMI-CR, TAN-OMI-CRCL-CR, and TAN-CR significantly outperform the generative structure learning approach TAN-CMI (similar for 2-trees). However, the computationally expensive greedy heuristic TAN-CR and 2-tree-CR do not significantly outperform our discriminative order-based heuristics TAN-OMI-CR and 2-tree-OMI-CR, respectively.

6.5.3 DISCUSSION: COMPUTATIONAL REQUIREMENTS FOR STRUCTURE LEARNING

The running time of the TAN-CMI, TAN-OMI-CR, and TAN-CR structure learning algorithms for the UCI, TIMIT-4/6, TIMIT-39, and the MNIST data sets is summarized in Table 4. The numbers represent the percentage of time that is needed for a particular algorithm compared to TAN-CR. TAN-CMI is roughly 3-10 times faster than TAN-OMI-CR and TAN-CR takes about 10-40 times longer for establishing the discriminative structure than TAN-OMI-CR.

Data set	TAN-CMI	TAN-OMI-CR	TAN-CR
UCI	0.649%	3.155%	100.00%
TIMIT-4/6	3.56%	11.47%	100.00%
TIMIT-39	0.11%	2.08%	100.00%
MNIST	0.21%	2.23%	100.00%

Table 4: Running time of structure learning algorithms relative to TAN-CR.

6.6 Results with Randomly Missing Input Features

As mentioned in Section 3, generative models can easily deal with missing features simply by marginalizing out from the model the missing feature. We are particularly interested in a testing context which has arbitrary sets of missing features for each classification sample. 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. Due to the local-normalization property of Bayesian networks and the structure of any model with a parent-less class node, marginalization is as easy as an $O(r^{k+1})$ operation for a k-tree, where r is the domain size of each feature.

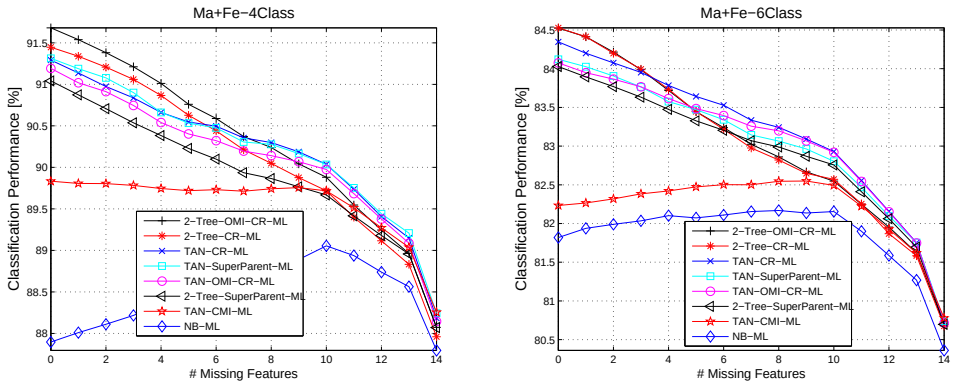

Figure 11: Classification performance of different structure learning methods assuming missing features using Ma+Fe data of TIMIT-4/6.

In Figure 11, we present the classification performance of discriminative and generative structures assuming missing features using the Ma+Fe data of TIMIT-4/6. The x-axis denotes the number of missing features. The curves are averaged over 100 classifications of the test data with uniformly at random selected missing features. We use exactly the same missing features for each classifier. Variance bars are omitted to improve readability, but indicate that the resulting differences are significant between NB-ML, TAN-CMI-ML, and discriminatively structured classifiers for a low number of missing features. Hence, discriminatively structured Bayesian network classifiers outperform TAN-CMI-ML even in the case of missing features. This demonstrates, at least empirically, that discriminatively structured generative models do not lose their ability to impute missing features.

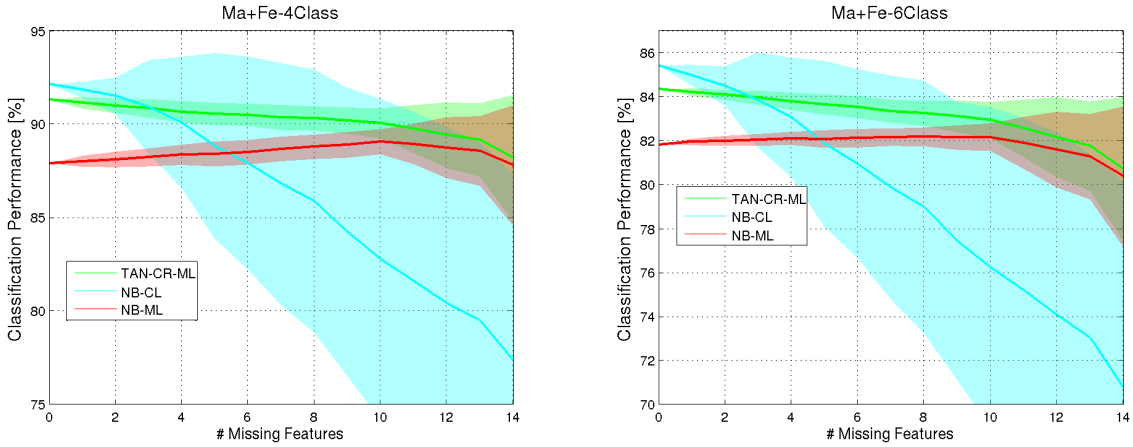

Figure 12: Classification performance of NB-ML, NB-CL, and TAN-CR-ML assuming missing features using Ma+Fe data of TIMIT-4/6. The shaded region corresponds to the standard deviation over 100 classifications.

In Figure 12, we show for the same data sets and experimental setup that the classification performance of a discriminatively structured model may be superior to discriminatively parameterized models in the case of missing features. In particular, for more than three missing features TAN-CR-ML outperforms NB-CL. Similar results can be shown for MNIST in Figure 13.

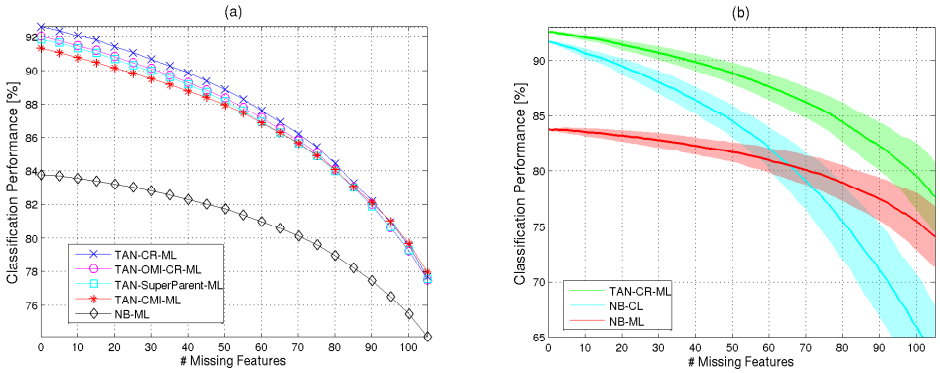

Figure 13: Classification performance using MNIST. The shaded region corresponds to the standard deviation over 100 classifications: (a) Different structure learning methods, (b) NB-ML, NB-CL, and TAN-CR-ML.

6.7 Results with SVMs

In Table 6, we compare classification performances between the best performing Bayesian network classifiers (in Table 1) and SVMs using RBF kernels. SVMs outperform our discriminative Bayesian network classifier. For TIMIT-4/6 one reason might be that SVMs are applied to the continuous feature domain. In Table 5 we compare the model complexity (i.e., number of parameters) between both SVMs and Bayesian network classifiers. This table reveals that the Bayesian network uses $\sim$108, $\sim$66, $\sim$212, and $\sim$259 times fewer parameters for MNIST, USPS, Ma+Fe-4, and Ma+Fe-6 than the SVM. This might also explain the loss in classification performance of discriminative Bayesian networks. Furthermore, Bayesian network classifiers can be directly applied to problems with more than two classes, whereas SVMs in its traditional formulation are limited to binary problems—the multiclass problem is decomposed into binary problems. Additionally, for SVMs we have to select C^* and σ. A substantial difference is that SVMs determine the number of support vectors automatically while in the case of Bayesian networks the number of parameters is specified by the structure. A limited complexity class (e.g., 1-tree) restricts the number of parameters which might be advantageous. In contrast to SVMs, a Bayesian network might be preferred since it is easy to work with missing features (see Section 6.6), parameter tying and knowledge-based hierarchical decomposition is facilitated, and it is easy to work with structured data.

DATA SET	TIMIT-4/6	MNIST	USPS
CLASSIFIER			
NB-CL	88.69		
TAN-CR-CL		93.94 ± 0.24	95.83 ± 0.36
SVM	**89.38**	**96.40** ± 0.19	**97.86** ± 0.26
PARAMETERS	$C^* = 1, \sigma = 0.05$	$C^* = 1, \sigma = 0.01$	$C^* = 1, \sigma = 0.005$

Table 5: Model complexity for best Bayesian network (BN) and SVM.

7. Conclusion

We introduced a simple order-based heuristic for learning discriminative network structure. The metric for establishing the ordering of N features is based on either the conditional mutual infor-

DATA SET	N	NUMBER OF SVS	NUMBER OF SVM PARAMETERS	NUMBER OF BN PARAMETERS
MNIST	196	17201	3371396	31149
USPS	256	3837	982272	14689
TIMIT-4/6 (MA+FE-4)	20	13146	262920	1239
TIMIT-4/6 (MA+FE-6)	20	24350	487000	1877

Table 6: Classification results for TIMIT-4/6 data sets (averaged), MNIST, and USPS using best performing Bayesian network classifier (see Table 1) and SVMs.

mation or the classification rate. Given an ordering, we can find a discriminative classifier structure using $O\left(N^{k+1}\right)$ score evaluations (where constant k is the tree-width of the sub-graph over the attributes). We empirically compare the performance of our algorithms to state-of-the-art discriminative and generative parameter and structure learning algorithms using real data from the TIMIT speech corpus, the UCI repository, a visual surface inspection task, and from handwritten digit recognition tasks. The experiments show that the discriminative structures found by our order-based heuristics achieve on average a significantly better classification performance than the generative approach. Our obtained classification performance is very similar to the greedy search using CR. Our order-based heuristics however, are about 10-40 times faster. Additionally, we show that discriminatively structured Bayesian network classifiers are superior even in the case of missing features.

Acknowledgments

We would like to acknowledge support for this project from the Austrian Science Fund (Project number P22488-N23) and (Project number S10604-N13).

Appendix A. UCI data

	DATA SET	# FEATURES	# CLASSES	# SAMPLES TRAIN	# SAMPLES TEST
1	AUSTRALIAN	14	2	690	CV-5
2	BREAST	10	2	683	CV-5
3	CHESS	36	2	2130	1066
4	CLEVE	13	2	296	CV-5
5	CORRAL	6	2	128	CV-5
6	CRX	15	2	653	CV-5
7	DIABETES	8	2	768	CV-5
8	FLARE	10	2	1066	CV-5
9	GERMAN	20	2	1000	CV-5
10	GLASS	9	7	214	CV-5
11	GLASS2	9	2	163	CV-5
12	HEART	13	2	270	CV-5
13	HEPATITIS	19	2	80	CV-5
14	IRIS	4	3	150	CV-5
15	LETTER	16	26	15000	5000
16	LYMPHOGRAPHY	18	4	148	CV-5
17	MOFN-3-7-10	10	2	300	1024
18	PIMA	8	2	768	CV-5
19	SHUTTLE-SMALL	9	7	3866	1934
20	VOTE	16	2	435	CV-5
21	SATIMAGE	36	6	4435	2000
22	SEGMENT	19	7	1540	770
23	SOYBEAN-LARGE	35	19	562	CV-5
24	VEHICLE	18	4	846	CV-5
25	WAVEFORM-21	21	3	300	4700

Table 7: 25 UCI data sets

CLASSIFIER	OMI-CR	RO-CR			
DATA SET		MEAN ± STD	MEDIAN	MIN	MAX
AUSTRALIAN	82.04	84.58 ± 0.77	84.62	84.63	83.60
BREAST	97.40	97.19 ± 0.26	97.23	97.39	96.95
CHESS	94.93	94.70 ± 0.76	94.75	95.22	93.90
CLEVE	81.76	81.42 ± 0.97	81.42	82.76	82.76
CORRAL	99.20	95.79 ± 2.55	96.17	96.80	95.20
CRX	84.07	84.21 ± 0.82	84.23	84.07	84.22
DIABETES	74.36	75.23 ± 0.52	75.26	75.53	75.01
FLARE	82.74	82.38 ± 0.50	82.44	81.80	81.98
GERMAN	73.20	72.83 ± 0.87	72.80	71.50	71.50
GLASS	70.70	71.74 ± 1.13	71.81	70.20	73.44
GLASS2	82.20	81.94 ± 0.73	82.14	82.14	82.20
HEART	81.85	82.83 ± 0.85	82.96	82.59	83.33
HEPATITIS	90.67	89.25 ± 1.69	89.00	90.33	90.33
IRIS	93.33	93.58 ± 0.55	93.33	92.67	93.33
LETTER	87.00	86.53 ± 0.60	86.54	86.70	86.50
LYMPHOGRAPHY	88.30	85.83 ± 1.72	85.92	87.12	81.27
MOFN-3-7-10	91.41	90.11 ± 1.01	90.14	89.55	89.75
PIMA	75.26	75.70 ± 0.49	75.65	75.78	76.56
SHUTTLE-SMALL	99.17	99.33 ± 0.13	99.33	99.22	99.17
VOTE	94.29	93.90 ± 0.69	93.84	93.36	94.06
SATIMAGE	88.25	87.47 ± 0.44	87.45	87.05	87.20
SEGMENT	94.42	94.63 ± 0.59	94.68	92.99	95.45
SOYBEAN-LARGE	91.11	92.29 ± 0.78	92.34	92.14	91.74
VEHICLE	67.38	68.18 ± 0.90	68.13	67.42	68.20
WAVEFORM-21	78.02	78.19 ± 0.51	78.21	78.77	77.79
AVERAGE	85.11	85.04	85.06	85.00	84.82

Table 8: Classification results in [%] with RO-CR compared to OMI-CR for the UCI data. Min (Max) reports the CR on the test set using the structure which achieves the minimum (maximum) performance over 1000 random orderings on the training data.

Appendix B. TIMIT-4/6 Data

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-CL	**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-CL	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-SUPERPARENT-ML	91.31 ± 0.15	91.84 ± 0.21	90.71 ± 0.23	84.12 ± 0.19	84.84 ± 0.27	83.51 ± 0.27	87.54
TAN-SUPERPARENT-CL	91.56 ± 0.15	92.29 ± 0.21	90.74 ± 0.22	84.36 ± 0.19	84.84 ± 0.27	83.83 ± 0.27	87.76
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-CL	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-OMI-CRCL-ML	91.09 ± 0.16	91.99 ± 0.21	90.35 ± 0.23	84.05 ± 0.19	84.59 ± 0.27	83.87 ± 0.27	87.46
TAN-OMI-CRCL-CL	91.41 ± 0.15	92.55 ± 0.21	90.54 ± 0.23	83.88 ± 0.19	84.71 ± 0.27	84.08 ± 0.27	87.62
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-CL	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-SUPERPARENT-ML	91.02 ± 0.16	91.84 ± 0.21	90.52 ± 0.23	84.01 ± 0.19	84.22 ± 0.27	83.42 ± 0.27	87.33
2-TREE-SUPERPARENT-CL	90.39 ± 0.16	91.51 ± 0.22	90.62 ± 0.23	83.17 ± 0.20	85.30 ± 0.26	83.96 ± 0.27	87.14
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-CL	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
2-TREE-CR-ML	91.45 ± 0.15	92.22 ± 0.21	91.11 ± 0.22	84.53 ± 0.19	85.36 ± 0.26	84.22 ± 0.27	87.94
2-TREE-CR-CL	91.00 ± 0.16	91.41 ± 0.22	90.52 ± 0.23	82.79 ± 0.20	84.46 ± 0.27	83.19 ± 0.28	86.95

Table 9: Classification results in [%] for 4 and 6 classes with standard deviation. Best results use bold font. ML and CL denote generative and discriminative parameter learning, respectively. OMI-CR (order mutual information-CR) refers to the order-based greedy heuristic. OMI-CRCL refers to OMI-CR using discriminative parameter learning during structure learning. The generative structure learning algorithm is abbreviated as CMI and the greedy discriminative structure learning is TAN-CR and 2-tree-CR.

CLASSIFIER		2-TREE-OMI-CR	2-TREE-RO-CR			
DATA SET	NUMBER OF CLASSES		MEAN $\pm$ STD	MEDIAN	MIN	MAX
MA+FE	4	91.68	91.50 ± 0.10	91.49	91.44	91.46
MA	4	92.28	92.23 ± 0.10	92.23	92.35	92.14
FE	4	91.03	90.95 ± 0.13	90.94	90.77	90.94
MA+FE	6	84.52	84.44 ± 0.13	84.45	84.64	84.49
MA	6	85.43	85.17 ± 0.17	85.18	85.06	85.30
FE	6	84.31	84.04 ± 0.16	84.05	83.96	84.11
AVERAGE		**88.01**	87.86	87.86	87.87	87.87

Table 10: Classification results in [%] with 2-tree-RO-CR compared to 2-tree-OMI-CR for 4 and 6 classes. Min (Max) reports the CR on the test set using the structure which achieves the minimum (maximum) performance over 100 random orderings on the training data.

References

S. Acid, L.M. de Campos, and J.G. Castellano. Learning Bayesian network classifiers: Searching in a space of partially directed acyclic graphs. *Machine Learning*, 59:213–235, 2005.

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

L.R. Bahl, P.F. Brown, P.V. de Souza, and R.L. Mercer. Maximum Mutual Information estimation of HMM parameters for speech recognition. In *IEEE Intern. Conf. on Acoustics, Speech, and Signal Processing*, pages 49–52, 1986.

P.L. Bartlett, M.I. Jordan, and J.D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

J. Bilmes. *Natural Statistical Models for Automatic Speech Recognition*. PhD thesis, U.C. Berkeley, 1999.

J. Bilmes. Dynamic Bayesian multinets. In *16th Inter. Conf. of Uncertainty in Artificial Intelligence (UAI)*, pages 38–45, 2000.

J. Bilmes, G. Zweig, T. Richardson, K. Filali, K. Livescu, P. Xu, K. Jackson, Y. Brandman, E. Sandness, E. Holtz, J. Torres, and B. Byrne. Discriminatively structured graphical models for speech recognition: JHU-WS-2001. Technical report, Johns Hopkins University, 2001.

C.M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

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

C.M. Bishop and J. Lasserre. Generative or discriminative? Getting the best of both worlds. *Bayesian Statistics*, 8:3–23, 2007.

N. Brenner, S. Strong, R. Koberle, and W. Bialek. Synergy in a neural code. *Neural Computation*, 12:1531–1552, 2000.

W.L. Buntine. Theory refinement on Bayesian networks. In *7th Conference on Uncertainty in AI (UAI)*, pages 52–60, 1991.

C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

C.K. Chow and C.N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transaction on Information Theory*, 14:462–467, 1968.

G. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

T. Cover and J. Thomas. *Elements of Information Theory*. John Wiley & Sons, 1991.

R.G Cowell, A.P. Dawid, S.L. Lauritzen, and D.J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Springer Verlag, 1999.

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

L.M. de Campos. A scoring function for learning Bayesian networks based on mutual information and conditional independence tests. *Journal of Machine Learning Research*, 7:2149–2187, 2006.

P. Domingos and M.J. Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130, 1997.

J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *12th International Conference on Machine Learning (ICML)*, pages 194–202, 1995.

Y. Ephraim and L.R. Rabiner. On the releations between modeling approaches for speech recognition. *IEEE Transactions on Information Theory*, 36(2):372–380, 1990.

Y. Ephraim, A. Dembo, and L.R. Rabiner. A minimum discrimination information approach for Hidden Markov Models. *IEEE Transactions on Information Theory*, 35(5):1001–1013, 1989.

U.M. Fayyad and K.B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *13th International Joint Conference on Artificial Intelligence*, pages 1022–1027, 1993.

N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29: 131–163, 1997.

N. Friedman, I. Nachman, and D. Peer. Learning Bayesian network structure form massive datasets: The sparse candidate algorithm. In *15th Conference on Uncertainty in AI (UAI)*, pages 196–205, 1999.

D. Geiger and D. Heckerman. Knowledge representation and inference in similarity networks and Bayesian multinets. *Artificial Intelligence*, 82:45–74, 1996.

R. Greiner and W. Zhou. Structural extension to logistic regression: Discriminative parameter learning of belief net classifiers. In *18th Conf. of the AAAI*, pages 167–173, 2002.

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

A. Gretton and L. Gÿorfi. Nonparametric independence tests: Space partitioning and kernel approaches. In *Algorithmic Learning Theory: 19th International Conference (ALT08)*, pages 183–198, 2008.

D. Grossman and P. Domingos. Learning Bayesian network classifiers by maximizing conditional likelihood. In *21st Inter. Conf. of Machine Lerning (ICML)*, pages 361–368, 2004.

A.K. Halberstadt and J. Glass. Heterogeneous measurements for phonetic classification. In *Proceedings of EUROSPEECH*, pages 401–404, 1997.

D. Heckerman. A tutorial on learning Bayesian networks. Technical Report MSR-TR-95-06, Microsoft Research, 1995.

G. Heigold, T. Deselaers, R. Schlüter, and H. Ney. Modified MMI/MPE: A direct evaluation of the margin in speech recognition. In *Intern. Conf. on Machine learning (ICML)*, pages 384–391, 2008.

T. Jebara. *Discriminative, Generative and Imitative Learning*. PhD thesis, Media Laboratory, MIT, 2001.

M.I. Jordan. *Learning in Graphical Models*. MIT Press, 1999.

B.-H. Juang and S. Katagiri. Discriminative learning for minimum error classification. *IEEE Transactions on Signal Processing*, 40(12):3043–3054, 1992.

B.-H. Juang, W. Chou, and C.-H. Lee. Minimum classification error rate methods for speech recognition. *IEEE Transactions on Speech and Audio Processing*, 5(3):257–265, 1997.

D.R. Karger and N. Srebro. Learning Markov networks: Maximum bounded tree-width graphs. In *Symposium on Discrete Algorithms*, pages 302–401, 2001.

E.J. Keogh and M.J. Pazzani. Learning augmented Bayesian classifiers: A comparison of distribution-based and classification-based approaches. In *7th International Workshop on Artificial Intelligence and Statistics*, pages 225–230, 1999.

R. Kohavi and G.H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97:273–324, 1997.

J.B. Kruskal. On the shortest spanning subtree and the traveling salesman problem. In *Proceedings of the American Mathematical Society*, volume 7, pages 48–50, 1956.

L. Lamel, R. Kassel, and S. Seneff. Speech database development: Design and analysis of the acoustic-phonetic corpus. In *Proceedings of the DARPA Speech Recognition Workshop, Report No. SAIC-86/1546*, 1986.

S.L. Lauritzen. *Graphical Models*. Oxford Science Publications, 1996.

Y. LeCun and C. Cortes. The MNIST database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`.

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

C. Meek. Causal inference and causal explanation with background knowledge. In *11th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 403–410, 1995.

C. Merz, P. Murphy, and D. Aha. UCI repository of machine learning databases. Department of Information and Computer Science, University of California, Irvine, 1997. `www.ics.uci.edu/\~{}mlearn/MLRepository.html`.

T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

K.P. Murphy. *Dynamic Bayesian networks: Representation, Iference and Learning*. PhD Thesis, University of California, Berkeley, 2002.

N. Narasimhan and J. Bilmes. A supermodular-submodular procedure with applications to discriminative structure learning. In *21st Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2005.

A.Y. Ng and M. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *Advances in Neural Information Processing Systems 14*, 2002.

M. Pazzani. Searching for dependencies in Bayesian classifiers. In *Learning from Data: Artificial Intelligence and Statistics V*, pages 239–248, 1996.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

F. Pernkopf. Detection of surface defects on raw steel blocks using Bayesian network classifiers. *Pattern Analysis and Applications*, 7(3):333–342, 2004.

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

F. Pernkopf and J. Bilmes. Discriminative versus generative parameter and structure learning of Bayesian network classifiers. In *Intern. Conf. on Machine Learning (ICML)*, pages 657 – 664, 2005.

F. Pernkopf and J. Bilmes. Efficient heuristics for discriminative structure learning of Bayesian network classifiers. Technical report, Laboratory of Signal Processing and Speech Communication, Graz University of Technology, 2008a.

F. Pernkopf and J.A. Bilmes. Order-based discriminative structure learning for Bayesian network classifiers. In *International Symposium on Artificial Intelligence and Mathematics*, 2008b.

F. Pernkopf and M. Wohlmayr. On discriminative parameter learning of bayesian network classifiers. In *European Conference on Machine Learning (ECML)*, pages 221–237, 2009.

F. Pernkopf, T. Van Pham, and J.A. Bilmes. Broad phonetic classification using discriminative Bayesian networks. *Speech Communication*, 143(1):123–138, 2008.

W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical recipes in C*. Cambridge Univ. Press, 1992.

R. Raina, Y. Shen, A.Y Ng, and A. McCallum. Classification with hybrid generative/discriminative models. In *Advances in Neural Information Processing Systems 16*, 2004.

T. Roos, H. Wettig, P. Grünwald, P. Myllymäki, and H. Tirri. On discriminative Bayesian network classifiers and logistic regression. *Machine Learning*, 59:267–296, 2005.

B. Schölkopf and A.J. Smola. *Learning with kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.

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

M. Teyssier and D. Koller. Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *21th Conference on Uncertainty in AI (UAI)*, pages 584 – 590, 2005.

R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc B.*, 58(1): 267–288, 1996.

V. Vapnik. *Statistical Learning Theory*. Wiley & Sons, 1998.

X. Zhang, L. Song, A. Gretton, and A. Smola. Kernel measures of independence for non-iid data. In *Advances on Neural Information Processing Systems 22*, 2009.