



A targeted Bayesian network learning for classification

A. Gruber & I. Ben-Gal

To cite this article: A. Gruber & I. Ben-Gal (2017): A targeted Bayesian network learning for classification, Quality Technology & Quantitative Management, DOI: [10.1080/16843703.2017.1395109](https://doi.org/10.1080/16843703.2017.1395109)

To link to this article: <https://doi.org/10.1080/16843703.2017.1395109>



Published online: 26 Oct 2017.



Submit your article to this journal [↗](#)



Article views: 31



View related articles [↗](#)



View Crossmark data [↗](#)



A targeted Bayesian network learning for classification

A. Gruber and I. Ben-Gal

Department of Industrial Engineering, Tel-Aviv University, Tel-Aviv, Israel

ABSTRACT

A targeted Bayesian network learning (TBNL) method is proposed to account for a classification objective during the learning stage of the network model. The TBNL approximates the expected conditional probability distribution of the class variable. It effectively manages the trade-off between the classification accuracy and the model complexity by using a discriminative approach, constrained by information theory measurements. The proposed approach also provides a mechanism for maximizing the accuracy via a Pareto frontier over a complexity–accuracy plane, in cases of missing data in the data-sets. A comparative study over a set of classification problems shows the competitiveness of the TBNL mainly with respect to other graphical classifiers.

ARTICLE HISTORY

Accepted 17 October 2017

KEYWORDS

Bayesian classifiers; complexity–accuracy trade-off; information theory; AI; machine learning; target-oriented learning

1. Introduction

Bayesian belief networks (BNs) constitute a family of graphical probabilistic models, aiming to efficiently encode the joint probability distribution of some domain (Pearl, 1988). A BN model qualitatively and quantitatively represents the relations among random variables in a fairly intuitive manner, and at the same time provides a compact mathematical representation of the joint probability distribution (Ben-Gal, 2007; Heckerman, Geiger, and Chickering, 1995). Due to this generality, BNs have been used for various data analytics tasks, including prediction, anomaly detection, clustering, decision-making and classification. They have been applied to various systems and areas, such as retail, e-commerce, agriculture, homeland security, industry, meteorology, education, surveillance and more (Ben-Gal, 2007; Bouchaffra, 2012; Heckerman et al., 1995; Pourret, Naïm, & Marcot, 2008).

In many of these applications, the final underlying objective is to optimize some system output against operational, environmental and economic considerations. With the growing demand to model and analyze such systems directly from real data, automated BN learning approaches are essential and can be beneficial (Bouchaffra, 2012; Gruber & Ben-Gal, 2012; Gruber, Yanovski, & Ben-Gal, 2013; Zhu, Liu, & Jiang, 2016). We follow this direction and propose an engineering approach for learning a BN classifier that controls the accuracy–complexity trade-off during the learning stage, and provides a useful descriptive model for classification purposes. Unlike the latter two works that focus on specific reliability and maintenance applications, this work presents the use of a targeted Bayesian network learning (TBNL) method for a general classification purpose and benchmarks it with respect to other generalized graphic classifiers.

Learning a BN, and particularly its structure, is an NP-hard problem (Chickering, Geiger, & Heckerman, 1994). In the realm of BN classifiers, one can consider two main approaches for structure learning: a *canonical* approach and a *target-oriented* approach.

The canonical approach first applies a general Bayesian network (GBN) learning to depict the dependencies among the domain's variables, without focusing on any particular variable. Having learnt the GBN, the canonical approach then uses the variables in the networks as attributes of the class variable to carry out the classification. Thus, the learning phase of the network is conducted independently of the final classification task.

The target-oriented approach, in a different manner, sets the class variable a priori and constructs around it a model of the attributes. For example, the simple and well-known naïve-Bayes (NB) model (Duda, Hart, & Stork, 1973; Ng & Jordan, 2002) and the *tree augmented network* (TAN) (Friedman, Geiger, & Goldszmidt, 1997), are two cornerstone methods in the realm of target-oriented Bayesian classifiers. Following such an approach, the BN is initially constructed toward the classification task (Kelner & Lerner, 2012), focusing the modeling efforts around the class variable. Accordingly, the structural complexity of these classifiers is often fixed a priori, since there is very little 'room' for a BN structure search, and the trade-off between the classification accuracy and the model complexity is not fully managed.

The proposed TBNL method exploits advantages of the above two approaches. Following the canonical approach, it controls the model complexity during the learning stage by using information-theoretic measurements. On the other hand, similar to the target-oriented approach, it focuses on the class variable modeling. Thus, the TBNL manages the accuracy–complexity trade-off with respect to the expected conditional probability distributions of the class variable in the domain. It is shown that the accuracy–complexity trade-off is efficiently manageable by using measurements of Information Theory (Cover & Thomas, 2012), following the principles suggested by the *adding-arrows* method (Williamson, 2000).

The TBNL is compared with several directed graphical classifiers, such as the NB, the C4.5, the BNC algorithm and the TAN, in which the relations among random variables are represented graphically. Although the focus of this paper is on graphical classifiers, the benchmark study also includes the *support vector machine* (SVM) classifier as a popular non-graphical model in the considered industry. The paper focuses on graphical models since, unlike several 'black-box' classifiers (e.g. neural networks and deep-learning models), graphical models such as the BN and the proposed TBNL classifiers yield descriptive and interpretable networks. These networks can support better understanding and intuition to domain experts on the selected features and their interactions. For example, two specific use cases in which the TBNL provided insights to domain-experts by relying on its graphical interface, are *suspects identification* in cellular networks and *condition-based maintenance* (CBM) in production processes.

As seen in Section 4, the TBNL was compared against the above-mentioned classifiers over several known data-sets (taken from publicly available sources, such as Frank & Asuncion, 2010; Kohavi & John, 1997). Over the 25 tested data-sets, the TBNL dominated all the compared classifiers in 8/25 cases, while in 14/25 of the cases both the TBNL and the TAN were found to obtain the best classification results. These models were further compared to each other and benchmarked against the popular NB model. The results reflect the TBNL usability. First, in general it is well known that there is not one classifier that outperforms all the other classifiers over all data-sets. The literature recognizes that each classifier can have its own contribution and properties. Second, the TBNL graph has a convenient descriptive structure, which enables a more intuitive understanding of the explanatory variables and their interactions. This is the reason we focus mainly on graphical classifiers. The visual advantage of graphical models is unquestioned and is emphasized by other types of graphical models as in Wang, Xie, Liu, and Yu (2012) for example. Third, unlike generative Bayesian classifiers, such as NB and TAN, the TBNL draws these interactions in a form that might imply causality, which is one of the profound properties of Bayesian networks (Pearl, 2003). Although causality is not unique for Bayesian networks (Wang & Chan, 2012), it goes very well with the graphical representation of the

network. Modeling these interactions, along with the control of the accuracy–complexity trade-off, enables the TBNL to handle missing data effectively, as shown in Section 4.

The rest of this paper is organized as follows. Section 2 discusses the BN learning approaches. It gives an overview of some learning methods for classification, both of canonical and target-oriented approaches. Section 3 defines the problem of constrained BN learning and formulates the proposed TBNL approach and discusses some of its properties. Section 4 provides a numerical study comparing the TBNL against other BN methods and decision trees. Section 5 concludes the paper.

2. Learning BN classifiers

BN learning often comprises two main phases: structure learning in the form of a directed acyclic graph (DAG), and parameters learning, normally in a form of conditional probability tables (CPTs). Most of the challenge of BN learning lies in the structure learning. A BN learning method, of which the objective is to encode the domain's joint probability distribution, is referred to as a GBN learning (Cheng & Greiner, 1999). A comprehensive collection of works in this field can be found in Koller and Friedman (2009) and Spirtes, Glymour, and Scheines (2000).

Theoretically, if the true joint probability distribution is known, it is said to be consistent with any complete BN (Friedman, Mosenzon, Slonim, & Tishby, 2001; Tishby, Pereira, & Bialek, 2000). In practical situations, when the true distribution is unknown, and only observed data points are given, a complete BN model may result in overfitting the data and an increased prediction error.

Over the years, several approaches which are not limited to BN learning were suggested for tackling overfitting and the accuracy–complexity trade-off. One well-known approach is the Minimum Description Length (MDL) (Rissanen, 1978). The MDL uses the Kullback–Leibler (KL) divergence that is also used in this work. The KL divergence provides a measure of fit between two distributions, often one is considered as the true unknown distribution, while the second is its estimated distribution. The KL divergence, denoted here by $d_{\text{KL}}(p||q)$, can be thought of as a penalty term (measured in bits of code length), for relying on the distribution q , while the true distribution is in fact p . As such, a model that minimizes the KL divergence also minimizes the description length. For example, the C4.5 algorithm that applies the MDL principle on decision trees attempts to maximize the information about the class variable (Quinlan & Rivest, 1989). Also, in unsupervised tasks it is used for clustering and dimension reduction (Xia, Zong, Hu, & Cambria, 2013; Zhao et al., 2015).

A general problem formulation of such scoring functions and modeling was proposed by de Campos (2006) as:

$$g(B, D) = LL_D(B) - C(B)f(m), \quad (1)$$

where B represents any candidate model (a GBN in our case), D is the observed data, m is the number of instances in D , $LL_D(B)$ is the corresponding log-likelihood given the candidate model, $C(B)$ is the complexity measure of the candidate model and f is the information criteria. For example, if $f = 0$, the scoring function becomes a maximum likelihood which results in a complete BN. If $f = 1$, the score is based on *Akaike's information criteria* (Akaike, 1974), and if $f = 0.5\log(m)$, the score is based on *Bayesian information criteria* (Schwarz, 1978), which coincides with the MDL score (de Campos, 2006). The information criteria penalize the scoring function, while aiming at maximizing the log-likelihood term.

Equation (1) sets a goal for GBN learning and is being used in the first step of the canonical classification. The second step is applying classification on the basis of the learned GBN (Cheng & Greiner, 1999; Chickering, 2002; Cooper & Herskovits, 1992; Heckerman et al., 1995; Spirtes et al., 2000; Wang & Chan, 2012; Williamson, 2000). Such an approach that we follow here appears in Williamson (2000). The author proposed the *adding-arrows* algorithm, which learns a GBN by attempting to maximize the *total information weight*, or equivalently the *multi-information* (Friedman et al., 2001), of the learned BN, subject to some arbitrary structural constraints. The *adding-arrows* method uses Equation (1) implicitly by maximizing the multi-information of the BN, which is identical to maximizing the

log-likelihood measure (Buntine, 1996; Friedman et al., 1997). The *adding-arrows* can be adapted to Equation (1), by conditioning $f = 0$ as long as the constraints are satisfied, and $f = \infty$ if they are not. Williamson (2000) generalized the arguments of Chow and Liu (1968) from a Bayesian tree to a BN. In general, maximizing the log-likelihood is equivalent to maximizing the total mutual information (MI) of the learned BN. Similarly, maximizing the conditional log-likelihood of the class variable is equivalent to minimizing the KL divergence between the true conditional distribution (of the class variable) and the modeled one (Friedman et al., 1997).

While GBN learning methods attempt to best approximate the joint probability distribution, they address the trade-off between the BN complexity versus the prediction accuracy of the learned BN. Unlike GBN learning methods, the *target-oriented* methods form the structure learning specifically for classification. The NB model, for instance, does not require any structure learning. Instead, the structure is fixed a priori, where the node representing the class variable is predetermined as the common parent of all nodes that represent the rest of the variables (attributes).

Friedman et al. (1997) presented the TAN model, as well as the Bayesian augmented Bayesian network classifier (BAN), each of which poses an improvement upon the NB model in classification. The TAN can be considered as a combination of the NB classifier and an internal spanning tree, based on Chow & Liu algorithm (CLA). Thus, the TAN produces a BN in which every attribute has another parent in addition to the class node. The structure that the TAN suggests is predetermined to the extent of the class variable, while the scoring function can take the form of the log-likelihood, for example, to learn the tree structure among the attributes. Such a target-oriented model (including models that are based on it, e.g. Keogh & Pazzani, 1999), leaves the complexity component in Equation (1) with very little influence, as with the trade-off considerations.

Grossman and Domingos (2004) proposed the BNC, which substitutes in Equation (1) the conditional log-likelihood of the class variable as the primary objective function. The BNC offers two versions for cutting down the BN's complexity, either by bounding the maximum number of parents (referred to as BNC-NP, where N stands for the maximum number of parents of each node) or by applying the MDL principle. The authors claim that for realistic data-sets of medium or large sizes, discriminative classifiers (that model directly the conditional probability distribution of the class variable) are often preferable over generative models (that are learned by maximizing the log-likelihood of the data being generated by the model). This issue has been discussed in other works as well (Kelner & Lerner, 2012; Madden, 2009; Ng & Jordan, 2002; Roos, Wetteg, Grünwald, Myllymäki, & Tirri, 2005). In our benchmark study, we focus on graphical models including popular generative models such as the NB and the TAN, yet we also consider the SVM model as a predominant discriminative classifier (Cortes & Vapnik, 1995; Ghanty, Paul, & Pal, 2009). The proposed TBNL classifier is discriminative, while the applied parameter learning scheme is generative, as proposed by Grossman and Domingos (2004), for each of the considered Bayesian network classifiers.

An important feature of the TBNL that is associated with Bayesian networks in general is the handling of missing data values. In general, there are several approaches for dealing with missing values. García-Laencina, Sancho-Gómez, and Figueiras-Vidal (2010) associate each method for handling missing data to one of four types of approaches: case deletion, imputation, model-based procedures and machine learning methods. Most of these approaches consider missing data values during the training time. Saar-Tsechansky and Provost (2007) state that the literature is quite poor when dealing with missing values at prediction time (i.e. during the testing phase), as we do here. Moreover, considering these approaches, the paper indicates that 'there are almost no comparisons of existing approaches nor analyses or discussions of the conditions under which the different approaches perform well or poorly'.

In this work, we propose the TBNL classifier that effectively manages the trade-off between the classification accuracy and the model complexity by using a discriminative approach, constrained by information theory measurements. The proposed model provides a mechanism for maximizing classification accuracy via a Pareto frontier over a complexity-accuracy plane, in cases of missing data in the tested data-sets, as shown in later sections.

3. Targeted Bayesian network learning

In this section, we present the used BN notation, discuss the problem formulation and introduce the TBNL method.

3.1. Bayesian network notation

Denote by $B(\mathcal{G}, \Theta)$, a Bayesian network that represents the joint probability distribution of a vector of random variables $\mathbf{X} = (X_1, \dots, X_N)$. The structure $\mathcal{G}(\mathbf{X}, \mathbf{E})$ is a DAG composed of a set of nodes \mathbf{X} and a set of directed edges \mathbf{E} , connecting the nodes. An edge $E_{ji} = X_j \rightarrow X_i$ manifests dependence between the variables X_j and X_i . A directed edge E_{ji} connects a parent node X_j to its child node X_i (Heckerman et al., 1995). We denote by $\mathbf{Z}_i = \{X_i^1, \dots, X_i^{L_i}\}$ the set of parents of X_i in $\mathcal{G}(\mathbf{X}, \mathbf{E})$ where $L_i = |\mathbf{Z}_i|$ is the size of \mathbf{Z}_i and $\mathbf{Z}_i \subset \mathbf{X}$. The set of parameters Θ holds the local CPTs over \mathbf{X} , where $p(x_i|\mathbf{z}_i)$ is the conditional probability of x_i given \mathbf{z}_i (a conjunction of states $x_i^1 \cap \dots \cap x_i^{L_i}$) of \mathbf{Z}_i .

3.2. Problem definition

Denote a given target or class variable by $X_t \in \mathbf{X}$ and the rest of the domain's variables $\mathbf{X} \setminus \{X_t\}$ by $\bar{\mathbf{X}}_t$. $p(X_t)$ can be expressed by $E[p(X_t|\bar{\mathbf{X}}_t)]$, the expected conditional probability distribution of X_t over the rest of the domain's variables, thus, $p(X_t) = \sum_{\bar{\mathbf{x}}_t \in \bar{\mathbf{X}}_t} p(X_t|\bar{\mathbf{x}}_t)p(\bar{\mathbf{x}}_t)$ where $\bar{\mathbf{x}}_t$ denotes the atomic states of $\bar{\mathbf{X}}_t$. Ideally, $p(X_t)$ could be obtained by expectation over the joint probability distribution $p(\mathbf{X})$ using a full BN. When using a constrained BN following Equation (1), $p(X_t)$ can be approximated by $q(X_t)$, where $q(X_t) = \sum_{\bar{\mathbf{x}}_t \in \bar{\mathbf{X}}_t} p(X_t|\mathbf{z}_t)p(\bar{\mathbf{x}}_t)$ and \mathbf{z}_t denote a state of $\mathbf{Z}_t \subseteq \bar{\mathbf{X}}_t$ (the parents of X_t).

The problem definition stems from the general BN construction approach that is generally given by Equation (1). Yet, we use alternative expressions for both the log-likelihood and the complexity terms by means of information theory measurements (Bouckaert, 1995; Chow & Liu, 1968; Williamson, 2000).

For the first term in Equation (1), we follow Williamson (2000) and Chow and Liu (1968) and instead of maximizing the log-likelihood, we aim at minimizing the KL divergence between the true unknown probability distribution of the variables and the modeled (approximated) distribution by the BN. The KL divergence between the true joint probability distribution $p(\mathbf{X})$ and the modeled distribution $q(\mathbf{X})$ is given by:

$$d_{\text{KL}}(p||q) = \sum_{x_1, \dots, x_N \in \mathbf{X}} p(x_1, \dots, x_N) \log \frac{p(x_1, \dots, x_N)}{q(x_1, \dots, x_N)}. \quad (2)$$

When using the BN factorization to express q , one obtains $d_{\text{KL}}(p(\mathbf{X})||q(\mathbf{X})) = -H(\mathbf{X}) - \sum_i I(X_i; \mathbf{Z}_i) + \sum_i H(X_i)$, where $H(X_i)$ is the entropy ascribed to the distribution of X_i , $H(\mathbf{X})$ is the joint entropy ascribed to the joint distribution of \mathbf{X} , and $I(X_i; \mathbf{Z}_i)$ is the MI between the variable X_i and its conditioning variables, namely its parents in the network (see Williamson, 2000). Since the two entropy terms are independent of the BN model, the only term that could minimize the above KL divergence is the negative sum of the MI weights. This leads to the observation that maximizing the total MI weight results in a BN that best approximates $p(\mathbf{X})$, where edges pointing from parents to a node in the network are weighted by the information gain (IG) in the parents about the corresponding random variable. Based on these observations, Williamson (2000) proposed the *adding-arrows* algorithm, which is a straightforward generalization of the work presented by Chow and Liu (1968) for Bayesian trees (see also Gruber & Ben-Gal, 2012). The MI between a variable and its parents is given in Equation (3):

$$I(X_i; \mathbf{Z}_i) = \sum_{x_i \in X_i, \mathbf{z}_i \in \mathbf{Z}_i} p(x_i, \mathbf{z}_i) \log \frac{p(x_i, \mathbf{z}_i)}{p(x_i)p(\mathbf{z}_i)}. \quad (3)$$

The term for the total MI weight in a complete BN can then be expressed by using the chain rule of information, such that $I(X_i; \mathbf{Z}_i)$ for all $i = 1, \dots, N$ can be decomposed to a sum of IG components, thus:

$$\sum_{i=1}^N I(X_i; \mathbf{Z}_i) = \sum_{i=1}^{N-1} \sum_{j=i+1}^N I(X_i; X_j | X_{j-1}, \dots, X_{i+1}), \quad (4)$$

where $(X_{j-1}, \dots, X_{i+1}) = \emptyset$ if $j = i + 1$.

Note that each of the IG components in the above summation corresponds to an edge in the network. When the BN is unconstrained, the total MI weight, expressed by Equation (4), reaches the global maximum. Since there exist $N!$ different complete BN structures, each of which encodes the same probability distribution by a unique corresponding structure (also known as an I-Map of the distribution), the total MI weight can be decomposed to as many terms as the different ordering of the IG components in Equation (4). This observation is also known as the *score equivalence* (Koller & Friedman, 2009), implying that for a complete BN, the direction of the DAG does not change the probabilistic model, or its likelihood, given a data-set. A BN in which edges are not drawn for components that contribute zero IG is a *minimal* I-Map of the distribution.

The second term in Equation (1), which refers to the complexity of a BN, can also be expressed in terms of information theory measurements (Cover & Thomas, 2012). In particular, we propose a set of upper bounds on the MI about each node in the network, and use them to constrain the learned BN. Explicitly, we define $\{\eta_t, \eta_r\}$, such that $I(X_t; \mathbf{Z}_t) \leq \eta_t$ and $I(X_r; \mathbf{Z}_r) \leq \eta_r$, where the subscript r denotes a non-target variable (of the ‘rest’ of the variables), $X_r \in \bar{\mathbf{X}}_r$. In addition, similar to other works, one can also constrain the number of free parameters in the learned model (de Campos, 2006; Williamson, 2000). Thus, an explicit measure of the complexity can be given by the parameter:

$$k = \sum_{i=1}^N (|X_i| - 1) \prod_{j=1}^{L_i} |X_i^j|, \quad (5)$$

where N is the number of random variables, $|X_i|$ is the number of values that X_i can take and $|X_i^j|$ are all possible values that the parent variable X_i^j in \mathbf{Z}_i can take (X_i has L_i parents). Other approaches to complexity measurements could be easily integrated in the model (e.g. see Lam & Bacchus, 1994).

The general problem definition that was given in Equation (1) can now be expressed in terms of information measurements as follows.

Given $X_t \in \mathbf{X}$, find the BN $B(\mathcal{G}, \Theta)$ (namely find the set $\{\mathbf{Z}_i\}_{i=1}^N$) which minimizes $d_{\text{KL}}(p(X_t) || q(X_t))$, subject to an upper bound η_t on $I(X_t; \mathbf{Z}_t)$ and an upper bound η_r on each of the elements in $\{I(X_r; \mathbf{Z}_r)\}$ where $X_r \in \bar{\mathbf{X}}_r$.

Note that a differential consideration between the target variable and the rest of the variables can result in a different trade-off between accuracy and complexity in information theoretic terms. This is implemented in the proposed TBNL method which is discussed next.

3.3. The proposed TBNL method

The TBNL aims at maximizing the MI weights in the BN model, while constraining the learned BN by applying conditions on information-related measurements. Instead of modeling explicitly the joint probability distribution over all the variables, the TBNL models the expected conditional probability distribution of the target (class) variable over the ‘rest’ of the variables of the domain. During the modeling stage, constraints are imposed on each node, in the form of information bounds, yielding an efficient trade-off between the model accuracy and the model complexity, as given in Equation (5). The TBNL aims at representing the true and unknown distribution $p(X_t) = \sum_{\bar{\mathbf{x}}_t \in \bar{\mathbf{X}}_t} p(X_t | \bar{\mathbf{x}}_t) p(\bar{\mathbf{x}}_t)$, by a BN-based distribution $q(X_t) = \sum_{\mathbf{z}_t \in \mathbf{Z}_t} p(X_t | \mathbf{z}_t) p(\mathbf{z}_t)$, where \mathbf{z}_t denotes a state of the parents of X_t , $\mathbf{Z}_t \subseteq \bar{\mathbf{X}}_t \subset \mathbf{X}$.

The TBNL method forms a unique dependence structure among related variables. It restricts X_i to have no children, while restricting its parents' set to \mathbf{Z}_i . Accordingly, \mathbf{Z}_i can be roughly considered as a 'diluted' Markov blanket of X_i (containing its parents and their children). Other examples of BN classifiers that have 'degenerated' Markov blanket structures are the NB, the TAN and the BAN. In general, a BN classifier depends only on the Markov blanket of the class node (Koller & Friedman, 2009), hence, learning the interactions among the parents of X_i is redundant. However, since the TBNL evaluates the corresponding probabilities $p(\mathbf{z}_i)$, for each $\mathbf{z}_i \in \mathbf{Z}_i$, the method is found to efficiently handle classification tasks in cases of missing data. Numerical experiments that support this statement are discussed in Section 4.

Next, we present the main procedure of the TBNL algorithm. The procedure extends Williamson's (2000) observation to account for the expected conditional distribution of the class variable in a BN.

The TBNL algorithm uses a recursive procedure that can be applied on any node that represents a variable. The procedure, called the *AddParents* (described below), adds edges from candidate nodes to the node to which the procedure is currently applied – each time it adds the edge from the node with the highest IG. Essentially, the *AddParents* is a greedy, forward feature-selection (see also Friedman, Nachman, & Peér, [1999] that adds 'candidate parents' iteratively), which is similar to the feature selection scheme in the *adding-arrows* (Williamson, 2000). The main difference is that the TBNL algorithm starts with the class variable and then proceeds recursively to the selected parent nodes. In particular, the TBNL algorithm starts by applying the *AddParents* procedure to the target node to select its parents. The *AddParents* is then applied to each parent to select its own parents from the set of the target's parents. Thus, any node in the network is a parent of the target node (i.e. corresponding to a limited form of a Markov blanket), while maintaining the DAG structure.

Note that although the *AddParents* can be viewed as a modification of the feature-selection procedure in the *adding-arrows*, the TBNL algorithm results in a totally different network model. A possible explanation is that the *adding-arrows* follows the canonical approach, while the *AddParents* is a target-based procedure. Thus, despite the similarity of the procedures, the resulting networks of the two methods are profoundly different by structure, parameters and performance. This contribution is consistent with other algorithms in the literature that were somewhat modified to yield conceptually different networks. For example, Grossman and Domingos (2004) used HGC's hill climbing procedure from Heckerman et al., (1995), with the CLL in place of the LL, leading to the BNC classifier. Another example is the *adding-arrows* itself, that modified the CLA by using IG measurements of higher orders.

The input parameters of the *AddParents* are: X_i representing the current node; \mathbf{T}_i representing the set of the candidate parents of X_i ; \mathbf{C} representing the set of arbitrary constraints on the network, such as the number of permitted parameters; η_i representing a constraint of the maximum allowed MI about X_i ; β_i representing the minimum IG 'step size' upon adding a parent to X_i in the network.

The output of the *AddParents* procedure, once applied to node X_i is the set of parent nodes \mathbf{Z}_i if one of the following conditions is fulfilled: (1) any of the \mathbf{C} constraints is not met; (2) $I(X_i; \mathbf{Z}_i | \mathbf{Z}_i) / H(X_i) < \beta_i$; (3) the set of candidate parents \mathbf{T}_i is empty. The *AddParents* is presented in Figure 1.

Carefully looking at the procedure, the penultimate line implies that it is a quasi-recursive procedure. Namely, the TBNL algorithm actually calls *AddParents* only once. Having obtained \mathbf{Z}_p it iteratively calls $\mathbf{Z}_j = \text{AddParents}(X_j, \mathbf{Z}_j, \mathbf{C}, \eta_j, \beta_j)$ for each $X_j \in \mathbf{Z}_p$. Note that the order of the iterations is well defined – the output parents from each iteration affect directly the input of the next step. Such a procedure generates different outputs to those obtained, had the algorithm iterated the procedure only after obtaining the full set of parents. Thus, the TBNL calls $\mathbf{Z}_i = \text{AddParents}(X_i, \bar{\mathbf{X}}_i, \mathbf{C}, \eta_i, \beta_i)$, which ultimately results in a DAG $\mathcal{G} = \{\mathbf{Z}_1, \mathbf{Z}_2, \dots, \mathbf{Z}_N\}$.

In the comparative section, the advantage of controlling the complexity–accuracy trade-off is illustrated, where the differential complexity parameters β and η are used for such a control. Note, however, that such a control requires further computation time. As long as there are no missing data, modeling the dependencies among the attributes may be irrelevant for the classification task. However, in case

Input:
 \mathcal{G} : a DAG
 X_i : the current node
 \mathbf{T}_i : the set of candidate parents of X_i
 \mathbf{C} : a set of constraints on the BN
 η_i : maximum MI about X_i
 β_i : minimum IG about X_i given \mathbf{Z}_i

Output:
 \mathbf{Z}_i : the set of parents of X_i

Procedure:
Set $\mathbf{Z}_i = \{\emptyset\}$
Set $\mathbf{T}'_i = \mathbf{T}_i$
Stop = FALSE
While $\mathbf{T}'_i \neq \{\emptyset\}$ And Not(Stop)
 $X_j = \operatorname{argmax}_{j: X_j \in \mathbf{T}'_i} \{I(X_i; X_j | \mathbf{Z}_i) / H(X_i)\}$
 Add an arrow $E_{ji} = V_j \rightarrow V_i$ in \mathcal{G}
 If \mathbf{C} is not met
 Stop = TRUE
 If $I(X_i; \mathbf{Z}_i \cup \{X_j\}) \geq \eta_i$
 Stop = TRUE
 If $I(X_i; X_j | \mathbf{Z}_i) / H(X_i) < \beta_i$
 Stop = TRUE
 If Stop
 Remove E_{ji}
 Else
 $\mathbf{Z}_i = \mathbf{Z}_i \cup \{X_j\}$
 $\mathbf{T}'_i = \mathbf{T}'_i \setminus \{X_j\}$
End {While}
For each $X_k \in \mathbf{Z}_i$
 $\mathbf{Z}_k = \operatorname{AddParents}(\mathcal{G}, X_k, \mathbf{Z}_i, \mathbf{C}, \eta_k, \beta_k)$
Return \mathbf{Z}_i

Figure 1. The AddParents procedure.

of missing data at the testing phase, the classification accuracy often decreases and the attributes' dependencies can then be used to improve the approximation of the missing values and (partially) compensate for the reduced accuracy.

4. Experimental results

In this section, the TBNL is compared with known classifiers and its accuracy–complexity trade-off is discussed. The numerical comparison is focused on graphical classifiers such as BNs and a Decision Tree following Grossman and Domingos (2004) and Friedman et al. (1997). For completeness we also show the classification results of the SVM which is known as a dominant and popular non-graphical classifier.

4.1. Data and setup

Twenty-five data-sets taken from the UCI Repository (Frank & Asuncion, 2010; Kohavi & John, 1997) were used for the comparison. These data-sets are detailed in Table 1, including the numbers of variables, classes and instances. We followed previous studies (Friedman et al., 1997; Wang & Chan, 2012) in implementing the used testing method, i.e. either a holdout of a fixed number of instances (listed as ‘Holdout’) or five stratified folds with cross validation (listed as ‘CV5’). The comparison is based on the results reported in Friedman et al. (1997), Keogh and Pazzani (1999), Grossman and Domingos (2004) and Ben-Gal, Dana, Shkolnik, and Singer (2014) with the most competent results of each classifier being included in the study. These studies considered generative parameter learning, which we follow in this paper.

The TBNL algorithm was implemented in Matlab including procedures from the Bayes Net Toolbox (Mallows, 1973; ‘MATLAB’, 2010, Murphy, 2001). Continuous variables were discretized using the supervised method proposed by Ching, Wong, and Chan (1995). Although discretization is directly applied using off-the-shelf methods, it is a critical aspect for the classification task and should be considered in accordance with the applied classifier (Ching et al., 1995). Essentially, the used discretization method joins the values of each attribute, such that it maximizes the normalized (with respect to their joint entropy) MI about the target variable. The discretization was applied each time exclusively to the training set, while transforming continuous values on the corresponding testing set (each test was applied directly to the original values in the testing set). The TBNL parameters were tuned manually (with a default value $\beta = \{0.02, 0.05\}$ and $\eta = \{1, 1\}$). Note that a small β_t value can add more parents to the target node, whereas a large β_t value can stop the algorithm too early (before obtaining a significant IG). This setting normally yields a different network structure for each training set, with a variable number of parents to each node.

The TBNL was compared against the following graphical benchmark models: BNC-2P (recall that ‘2P’ implies up to two parents per each node); NB; C4.5; TAN; HGC (as a canonical benchmark model while relying on the original BN search algorithm of Heckerman et al., 1995); and the SVM which is a non-graphical yet a popular model in the industry.

Table 1. List of used UCI repository data-sets, including properties and test procedure.

	Data-set	#attributes	#classes	#instances (training)	Test methodology
1	Australian	14	2	690	CV5
2	breast	10	2	683	CV5
3	chess	36	2	3196	Holdout
4	cleave	13	2	296	CV5
5	corral	6	2	128	CV5
6	crx	15	2	653	CV5
7	diabetes	8	2	768	CV5
8	flare	10	2	1066	CV5
9	German	20	2	1000	CV5
10	glass	9	7	214	CV5
11	glass2	9	2	163	CV5
12	heart	13	2	270	CV5
13	hepatitis	19	2	80	CV5
14	iris	4	3	150	CV5
15	letter	16	26	20,000	Holdout
16	lymphography	18	4	148	CV5
17	mofn-3-7-10	10	2	1324	Holdout
18	pima	8	2	768	CV5
19	satimage	36	6	6435	Holdout
20	segment	19	7	2310	Holdout
21	shuttle-small	9	7	5800	Holdout
22	soybean-large	35	19	562	CV5
23	vehicle	18	4	846	CV5
24	vote	16	2	435	CV5
25	waveform-21	21	3	5000	Holdout

4.2. Comparable results

The figures shown in Table 2 display the classification accuracy (in percent) of the corresponding method listed in the column header. For each data-set, the most accurate and the least accurate results are marked in bold or italic letters, respectively. The same marking was applied to the results that are within 5% difference from the above, following the convention.

As seen, the TBNL obtained the best accuracy in 8/25 (32%) of these data-sets, while the second-best classifier, the TAN, obtained the highest accuracy in 4/25 (16%) of the cases. In 9/25 (36%), the TBNL obtained a bold mark, as it was found to be within the 5% range off the most accurate classifier in 'shuttle-small'.

The SVM results imply that in these cases the performance of the graphical classifiers is competent. In fact, it is found that in 14 out of 24 data-sets, the SVM results in lower classification accuracy with respect to the leading graphical classifier (i.e. below 5% of the best classification results). Figure 2 depicts six pairwise comparisons of classification accuracy between two competing models over all 25 data-sets. In each comparison, a Wilcoxon signed-rank test is used, as recommended by Demšar (2006) for a comparison between two classifiers over different data-sets. The following p -values designate the significance of the comparisons (the lower the p -value is, the more significant is the outperforming classifier). In Figure 2(a), the TBNL outperforms the BNC-2P in 14/25 of the cases (56%, $p < 0.4376$), in Figure 2(b), the TBNL outperforms the NB in 16/25 of the cases (64%, $p < 0.0263$), in Figure 2(c), the TBNL outperforms the TAN in 11/25 of the cases (44%, $p < 0.4772$), in Figure 2(d), the TBNL outperforms the C4.5 in 17/25 of the cases (68%, $p < 0.059$), and in Figure 2(e), the TBNL outperforms HGC in 19/25 of the cases (76%, $p < 0.001$). For the forthcoming discussion, a comparison between the BNC-2P and the TAN is also depicted in Figure 2(f), where the BNC-2P outperforms the TAN in 11/25 of the cases (44%, $p < 0.666$). Thus, for a critical significance level of 0.05–0.06, the above pairwise p -values indicate a significant dominance of the TBNL relative to the NB, the HGC and closely the C4.5, but not relative to the BNC-2P and the TAN. These three predominant classifiers (the TBNL, the BNC-2P and the TAN) do not significantly dominate one another, implying that model selection should be undertaken for each case exclusively, especially in real-world scenarios.

Table 2. Accuracy comparison of the TBNL with selected graphical or discriminative classifiers.

Data-set	TBNL	BNC-2P	NB	TAN	C4.5	HGC	SVM
Australian	85.7	87.0	86.5	86.1	84.9	85.6	55.5
breast	95.9	95.8	97.6	96.5	93.9	97.6	96.5
chess	96.9	95.8	<i>87.3</i>	92.4	99.5	95.3	93.8
cleave	81.4	80.0	82.1	<i>78.4</i>	79.4	<i>78.7</i>	54.7
corral	100.0	98.8	<i>87.2</i>	98.6	98.5	100	96.9
crx	86.4	84.2	85.0	83.7	86.1	86.9	65.7
diabetes	77.1	<i>73.3</i>	74.3	76.2	74.1	74.3	65.1
flare	82.0	82.0	<i>79.8</i>	82.2	82.7	82.2	82.3
German	73.2	73.6	75.4	73.9	72.9	<i>72.5</i>	70.0
glass	60.0	58.3	55.9	54.2	59.3	<i>31.2</i>	69.2
glass2	83.4	<i>73.1</i>	<i>77.6</i>	<i>77.5</i>	76.1	<i>53.0</i>	76.7
heart	80.0	81.3	84.5	81.5	78.2	85.2	55.9
hepatitis	88.8	83.8	<i>80.7</i>	87.0	82.5	81.2	83.6
iris	97.0	95.8	93.0	<i>92.4</i>	96.0	95.7	96.7
letter	72.2	81.7	<i>69.3</i>	82.5	87.8	<i>69.1</i>	na
lymphography	81.8	83.7	83.4	82.2	78.4	63.8	80.0
mofn-3-7-10	100.0	91.4	<i>86.7</i>	91.5	<i>84.0</i>	<i>86.7</i>	100
pima	72.9	73.3	74.3	76.2	74.1	74.3	65.1
satimage	82.9	82.6	80.9	86.1	82.3	72.9	84.2
segment	91.4	94.3	<i>87.8</i>	93.3	91.8	88.7	63.9
shuttle-small	99.1	99.5	98.6	99.1	99.4	<i>86.5</i>	89.4
soybean-large	84.3	92.5	91.5	93.6	91.1	35.3	87.2
vehicle	64.0	70.8	61.1	72.8	68.3	49.2	30.5
vote	96.8	95.8	<i>90.1</i>	94.9	94.7	95.4	95.4
waveform-21	64.6	73.3	78.6	74.7	65.1	56.6	86.1

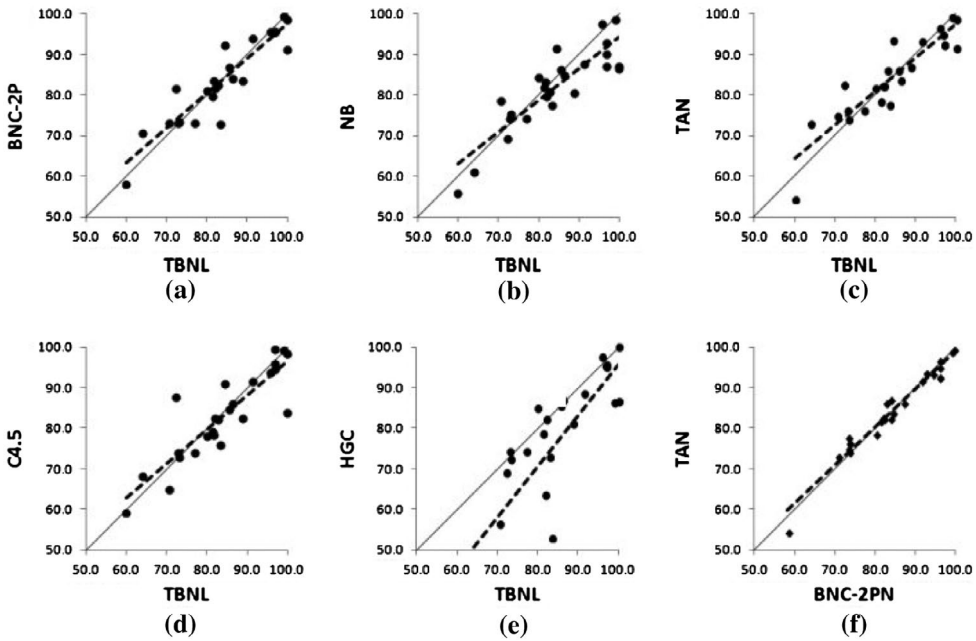


Figure 2. Pairwise comparisons of classification accuracy between two competing models over 25 tested data-sets. The solid line represents equality between the two compared classifiers, while the dashed line represents the tendency of the comparison between them as the accuracy increases.

Accordingly, we claim that the contribution of the TBNL should be evaluated in a manner which is similar to the unique contribution of the BNC despite the fact that the latter does not outperform the TAN on the average, or in the number of tests. Finally, in Section 4, we further show that the TBNL flexibility plays an important role when dealing with missing data values.

In each sub-figure of Figure 2, a linear curve $y = ax + b$ is fitted in order to indicate the tendency of the corresponding comparison. Except for Figure 2(e), in all comparisons, $a < 1$, which implies that the TBNL tends to underperform in cases of low accuracy and to outperform as the accuracy increases (to the extent of the value of b). This explication should not be surprising, since high accuracy is associated in many cases with unbalanced classes, for which the entropy is small. Thus, since the TBNL is a discriminative classifier on a basis of an entropy measure, this rough tendency does make sense. It can be seen in Figure 2(a) where $a = 0.863$, and (d) where $a = 0.846$, that the BNC-2P and the C4.5 accuracies are close to that of the TBNL, while Figure 2(f), where $a = 0.945$, implies that the BNC-2P and the TAN are very close to each other. The slopes for the comparison of the TBNL to generative classifiers (Figure 2(b) and (c)) are $a = 0.781$ and $a = 0.823$, respectively. In Figure 2(e), $a = 1.258$ is counterintuitive since it is expected that a canonical classifier will perform better in balanced cases, rather than in unbalanced cases. Note, however, that in this case the intercept is $b = -29.98$, which implies that the curve does not cross the $y = x$ line below 100% accuracy.

4.3. Accuracy-complexity trade-off

Consider now a robust indicator that measures the number of data-sets in which a classifier's accuracy is among the top two results. The TBNL and the TAN are found to be among the two leading classifiers in 12/25 (48%) of the cases, i.e. for nearly half of the cases. The TBNL algorithm addresses the trade-off between the classification accuracy and the model complexity by using two sets of information-related parameters, $\beta = \{\beta_t, \beta_r\}$ and $\eta = \{\eta_t, \eta_r\}$. The first coordinate of each parameter set controls the complexity that is associated with the class node (designated by subscript t). The second coordinate of

each parameter set controls the complexity associated with each of the rest of the nodes (designated by subscript r). Thereby, one can draw a line of the accuracy versus the complexity of the resulting BN. Figure 3 illustrates three cases from the data-sets ‘corral’, ‘mofn-3-7-10’ and ‘vote’, respectively, in which the TBNL is compared with the NB and the TAN. These three examples were selected for illustrating the advantage of the TBNL in the complexity management through the parameters β and η , compared to the TAN and the NB as classic target-oriented Bayesian network classifiers. Note that the computation time at each learning cycle of these models took up to a few minutes with a conventional 8-cores server. In that sense, the proposed TBNL is competent with respect to the other compared methods. Evidently, the fact that the differential complexity parameters β and η can be used to control the complexity–accuracy trade-off implies a larger computation time that depends on the number of the different models that are constructed.

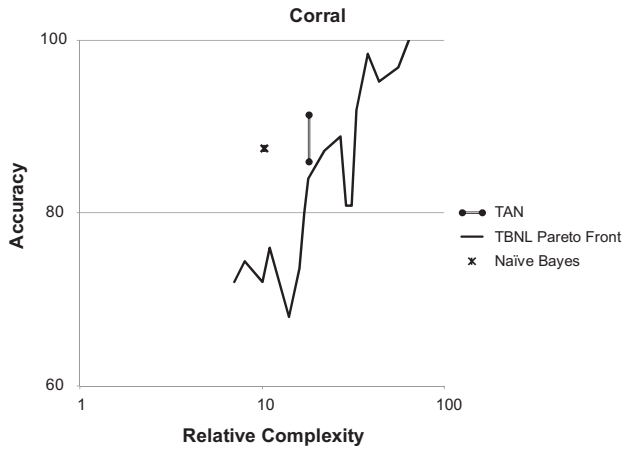
The accuracy of each learned BN is drawn as a function of its relative complexity (normalized with respect to the complete BN), for the corresponding data-set. The relative complexity in the horizontal axis is displayed in percent using a logarithmic scale. A common consistent pattern of the three figures is that the NBs’ accuracy is reflected by a single point – representing a single fixed BN structure. An example of NB Bayesian network, based on ‘corral’ data-set (Kohavi & John, 1997), is shown in Figure 4. Note again that the NB model is popular due to its simplicity, but it bears crude conditional independence assumptions.

The TAN’s accuracy is reflected by a vertical line – representing a number of potential structures, as obtained by the TAN algorithm. For example, six TAN classifiers that were learned from the ‘corral’ data-set and have the same complexity are shown in Figure 5. Note that there are six TAN models in this particular example, as in this data-set there are six attributes, each of which can be set as a root node to build the augmented dependence tree. Typically, each TAN model has a different accuracy due to different log-likelihood measurements, yet the complexity of these models is equal.

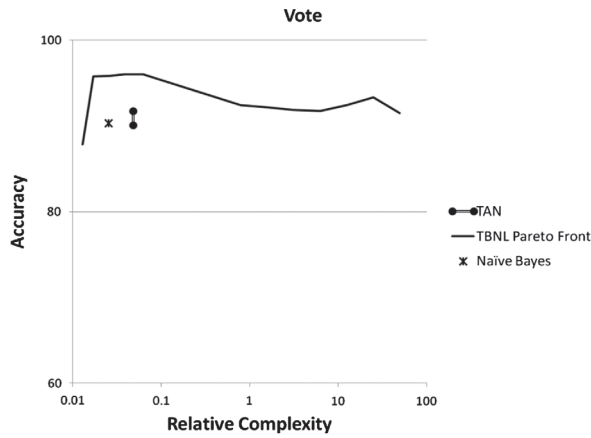
The TBNL’s accuracy is reflected by a curve representing the flexibility of the user to adjust the information-complexity parameters differentially on the class variable, as well as on the attributes. An example of a possible outcome of the TBNL, by fixing β and η to specific values, is shown in Figure 6 for the ‘corral’ data-set.

It can be observed from Figure 3 that the NB network has a relatively low accuracy with a relatively low complexity. Compared to the NB, the TAN has an accuracy range for each data-set, with a fixed complexity. Since the complexity of the TAN is constant, a trade-off study cannot be conducted, and it is appealing to select the model with the highest accuracy as a preferred one. In other words, the TAN’s accuracy can be maximized by finding the root attribute that maximizes the log-likelihood of the model. Compared to the NB and the TAN, the TBNL obtains a clear accuracy–complexity Pareto frontier. The parameters β and η enable regulating the ‘flow’ of information, and controlling the accuracy–complexity trade-off. For the purpose of analysis, η was set between 0 and 100% in intervals of 5% for each coordinate independently. This produced a 21×21 grid of networks drawing a two dimensional accuracy vs. complexity space. Then for each complexity level, the highest accuracy was selected and displayed in Figure 3 in ascending order of the corresponding complexity. Figure 3(a), ‘corral’ data-set, shows that the NB and the TAN achieve higher accuracy levels for a given complexity level, compared with the TBNL. This observation is also found in Figure 3(c), ‘mofn-3-7-10’ data-set, while for Figure 3(b), ‘vote’ data-set the TBNL is found to dominate the NB and the TAN altogether for different complexity levels. However, by adjusting the β and η parameters, the TBNL’s complexity can be controlled differentially to achieve a higher accuracy when possible.

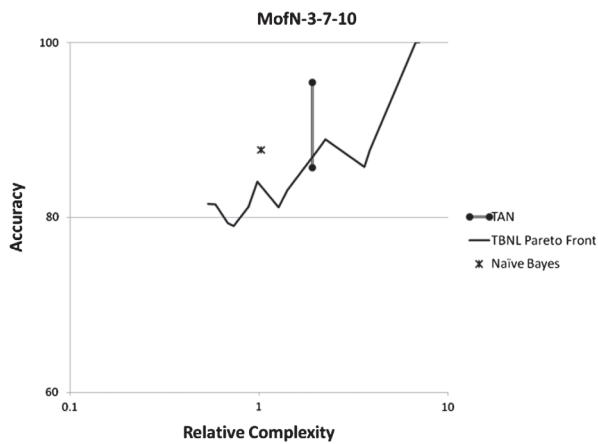
The differential complexity property of the TBNL can play an important role in cases of missing or unreliable data, where outliers appear and are needed to be re-estimated. Additional experiments were conducted for scenarios of missing data, following the arguments stated in Section 3.3 regarding missing-data handling. The method implemented in the present work follows the *missing at complete randomness* type. Description of missing-data mechanisms can be found in García-Laencina et al. (2010). For each data-set, three scenarios of (controlled) missing data were tested: 10, 30 and 60% of the records were omitted from one, two or three of the most important attributes, as shown in



(a) TBNL Pareto front of the 'corral' dataset



(b) TBNL Pareto front of the 'vote' dataset



(c) TBNL Pareto front of the 'mofn-3-7-10' dataset

Figure 3. (a) TBNL Pareto front of the 'corral' data-set. (b) TBNL Pareto front of the 'vote' data-set. (c) TBNL Pareto front of the 'mofn-3-7-10' data-set.

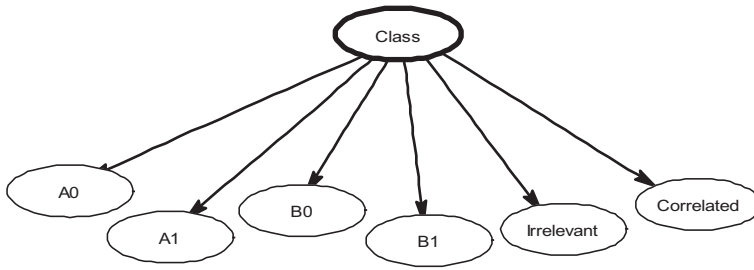


Figure 4. Naïve-Bayes Bayesian classifier for the 'corral' data-set.

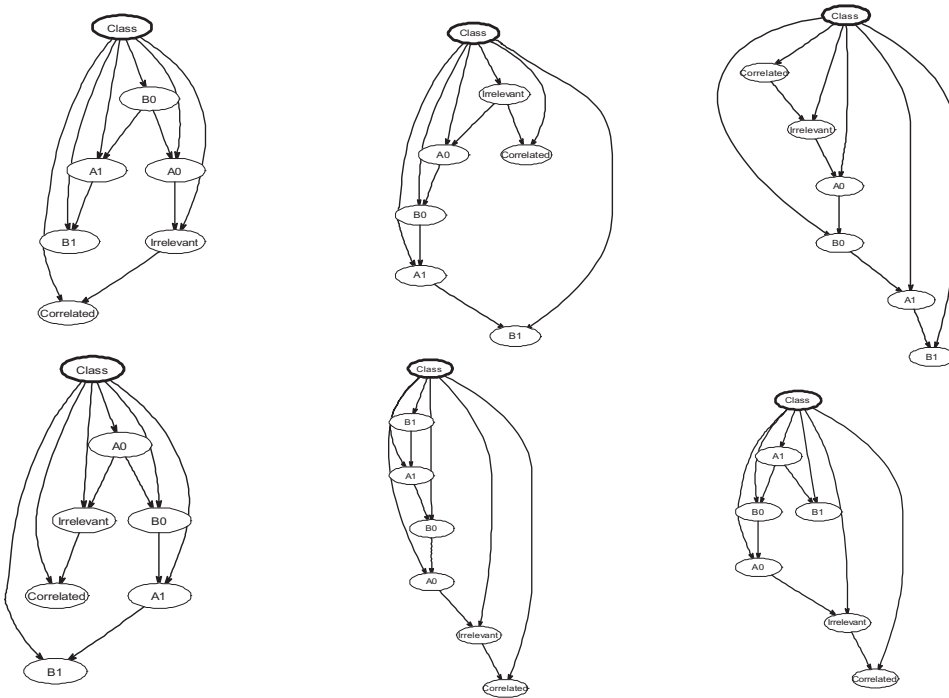


Figure 5. Tree augmented network classifiers for the 'corral' data-set.

Batista and Monard (2003). After dividing the data-set into training and testing sets (following either a holdout or cross validation protocol, as detailed in Table 1), for each of the three most influential attributes, every instance was omitted independently with probability parameter f_m . If n_m represents the number of attributes from which missing data are omitted, the probability for a record to have any missing value over its sampled features is expressed by $P(f_m, n_m) = \sum_{j=1}^{n_m-1} f_m (1 - f_m)^{j-1}$ (or identically by $P(f_m, n_m) = 1 - (1 - f_m)^{n_m-1}$). In this example, $n_m = 3$, and the values of f_m that derive {10%, 30%, 60%} missing rates are $f_m = \{0.0345, 0.112, 0.263\}$, respectively.

Figure 7 shows the result obtained for 'corral', 'mofn-3-7-10' and 'vote' data-sets, in which the accuracies of the TBNL, the TAN and the NB were evaluated as a function of the missing-data rate from 0 to 60%.

In the above illustrative cases, the TBNL maintains the highest accuracy in accordance with the results presented in Table 2, while the TAN and the NB models follow, respectively. Note that the TBNL is consistent with the rate growth, namely, its accuracy is monotonically non-increasing in the

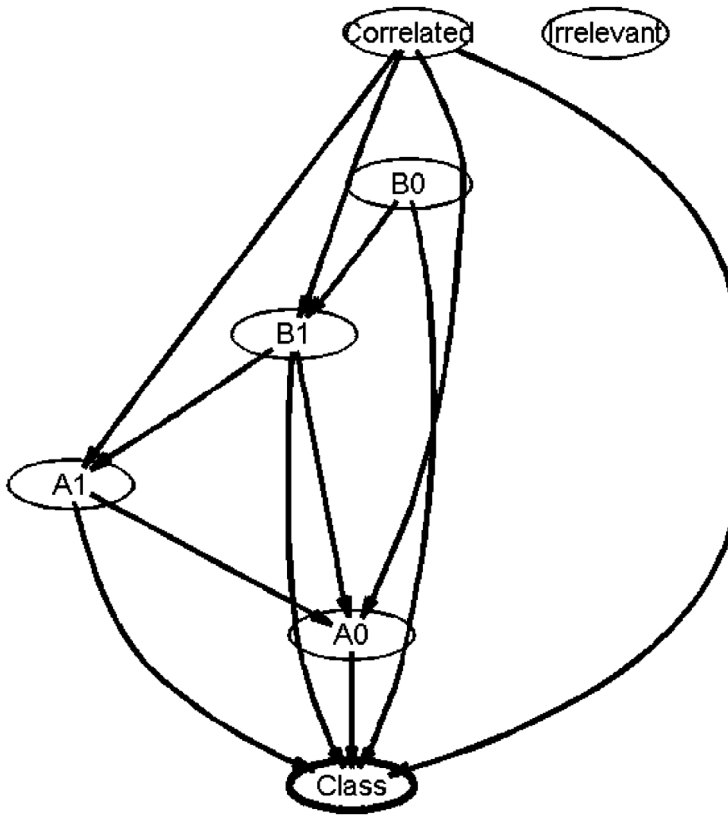


Figure 6. The TBNL classifier for the 'corral' data-set.

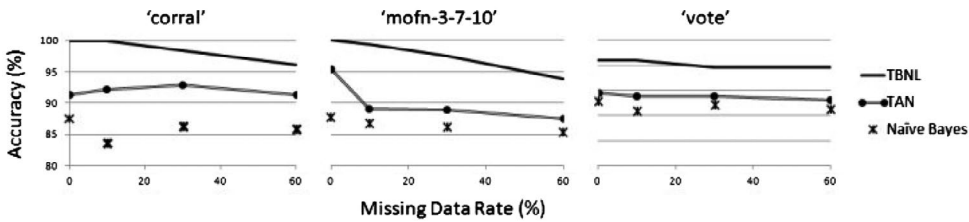


Figure 7. The accuracy of the TBNL, the TAN and the NB models in three data-sets with missing data.

missing-data rate. Counter-intuitively, both the TAN and the NB do not reflect the same consistent behavior.

One possible explanation to this observation is that the TBNL algorithm selects the most informative attributes, due to its flexibility, whereas the TAN and the NB that rely on all the attributes – some of which might become noisy and non-informative in case of missing data. Then, in certain cases of missing data, the noise is reduced, resulting in a higher accuracy of the TAN and the NB. Another possible explanation is that the TBNL exploits the available information efficiently, as it searches for the best structure using the differential complexity approach. Naturally, such efficiency is obtained by investing more computational efforts in learning the structure compared to the TAN or the NB. The latter approaches, on the other hand, have limited structure flexibility, and therefore are more sensitive to the particular available data-set.

4.5. The case of missing data values

A Pareto frontier of the ‘corral’ data-set is given in Figure 8. Similar to Figure 3, it illustrates the classification accuracy vs. the relative complexity of the TBNL, TAN and NB classifiers, but in addition, this comparison considers variable missing-data rates.

For the full data case (0% missing-data rate), the TBNL has two complexity points where it reaches 100% accuracy in a 5-fold cross validation test. In such case, it is preferable to choose the model with the lower complexity for 100% accuracy. In the 10% missing-data rate, the TBNL has still two points with 100% accuracy. But, the lower complexity point in the 10% missing-data rate has a larger complexity vs. the lower complexity point in the 0% missing-data rate. Note that even for the cases of 10, 30 and 60% missing-data rates, there are points for which the TBNL still outperforms the NB and the TAN in their 0% missing-data rate.

Note that generally for any given scenario and any given classifier the greater the rate of missing data, the less accurate the model is. Yet, in many cases the TBNL has a flexibility that can tackle this accuracy loss more effectively, as the control of the model complexity enables to select a different structure that potentially preserves a higher accuracy level. The TBNL, as a discriminative BN classifier, offers its straight-forward imputation method, which can be seen as a collection of local expected conditional probability distributions for each attribute, winding up with the class variable as a leaf. This is in accordance with the observation of Saar-Tsechansky and Provost (2007), who emphasize that the dominance of one form of imputation versus another depends on the statistical dependencies (and lack thereof) between the features.

Finally, let us note that the TBNL is scalable to larger and realistic data-sets. Two real-world uses-cases of a specific implementation of TBNL approach have been conducted and are not included in this paper due to lack of space. The first application considers a CBM framework for a European operator of a freight rail fleet, as presented by Gruber et al. (2013). In this study, the TBNL models the interactions between the mechanical components of a freight rail wagon, along with maintenance-related variables (e.g. cumulated mileage and maintenance policy), and their effects on failure prediction. The second application of the TBNL is related to further sensitivity analyses of CBM policies that are based on system simulations (Gruber et al., 2013). In particular, simulations were used to explore the robustness of various CBM policies under different service scenarios. The Bayesian network, which was learned from the simulation data, was then used as an explanatory compact metamodel for failure predictions.

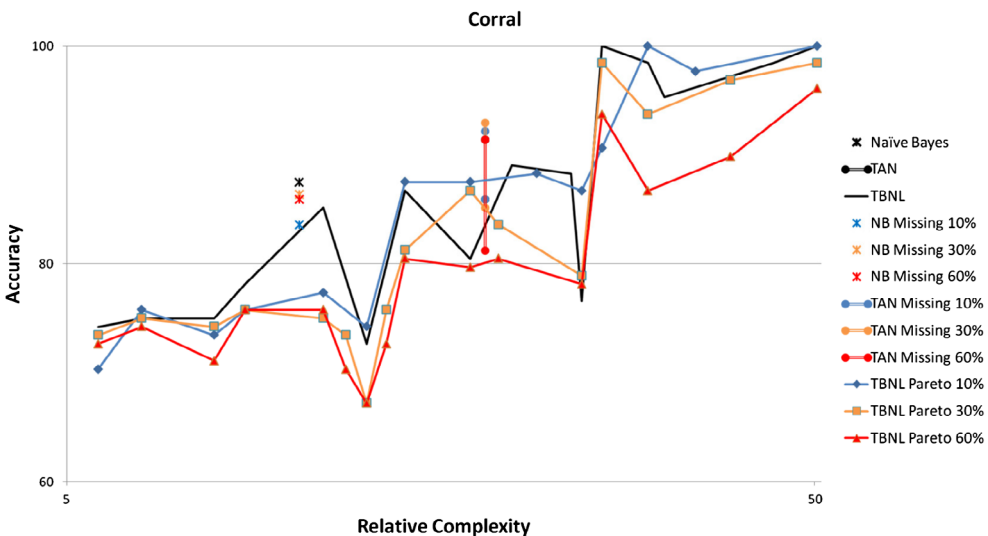


Figure 8. TBNL Pareto front of the ‘corral’ data-set, with missing values.

These preliminary applications of the TBNL approach materialized the TBNL method that is proposed in this paper.

5. Conclusion

The motivation to develop the TBNL arises from the growing demand to model real-world data-driven systems. Such systems are often related to an underlying objective that can often be formulated as the target variable in the model. In this study, we particularly focused on classification as the model target, since there is a firm ground for comparisons versus other graphical classifiers. The TBNL method reflects an engineering approach of target-oriented BN learning, as it puts together a discriminative model with information-related constraints to control the model complexity. Unlike canonical methods that manage the accuracy–complexity trade-off in the learned GBN, the proposed approach differentiates the constraints related to the target variable from those treated to the attributes. Consequently, the TBNL effectively manages the accuracy–complexity trade-off for the classification task. Using a Pareto frontier, it was illustrated how a better classification accuracy can be obtained for a given model complexity. This approach is in contrast to other methods in which the model complexity is fixed or with little flexibility. This flexibility was used to emphasize the ability of the TBNL to handle missing data in the testing phase.

A numerical study over 25 standard data-sets showed that the TBNL obtained the highest accuracy in 8/25 (32%) of these data-sets, while each of the other classifiers obtained the highest accuracy in 4/25 (16%) at most. Moreover, the TBNL was among the two leading classifiers in 12/25 (48%) of the cases. Note that these results should not be surprising since by providing additional degrees of freedom to the model construction, a better accuracy could be extracted from a given data-set.

An important property of the TBNL is the target-oriented graphical representation. Unlike generative BN classifiers, the TBNL literally depicts the most influential attributes on the class variable, as well as their interactions. In many practical situations such a presentation is essential for domain experts, helping them to understand the nature of the interactions and possible hidden patterns in the data (Pearl, 1988).

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This research was partially supported the Israel Science Foundation [grant number 1362/10].

Notes on contributors

Aviv Gruber is a lead data scientist at Huawei Technologies in the cloud-networking group of Huawei's Tel-Aviv Research Center. He is a lecturer at Tel Aviv University, the course Computer Integrated Manufacturing (CIM), that is coupled with practical sessions at the CIM Lab within the undergraduate programme of the Department of Industrial Engineering & Management.

Irad Ben-Gal is a full professor in the Department of Industrial Engineering & Management at Tel Aviv University and the head of the "AI and Business Analytics" Lab in the Faculty of Engineering. He is leading the "Digital Life 2030" research partnership with Stanford University, where he held a visiting professor position. His research interests include, machine learning and AI applications for industrial and service systems and statistical methods for monitoring and analysis of complex processes.

References

- Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6), 716–723.
- Batista, G. E., & Monard, M. C. (2003). An analysis of four missing data treatment methods for supervised learning. *Applied Artificial Intelligence*, 17(5–6), 519–533.
- Ben-Gal, I. (2007). Bayesian networks. In F. Ruggeri, F. Faltin, R. Kenett (Eds.), In *Encyclopedia of statistics in quality and reliability*. Oxford: Wiley & Sons.
- Ben-Gal, I., Dana, A., Shkolnik, N., & Singer, G. (2014). Efficient construction of decision trees by the dual information distance method. *Quality Technology & Quantitative Management*, 11(1), 133–147.
- Bouchaffra, D. (2012). Mapping dynamic Bayesian networks to α -shapes: Application to human faces identification across ages. *IEEE Transactions on Neural Networks and Learning Systems*, 23(8), 1229–1241.
- Bouckaert, R. (1995). Bayesian belief networks from construction to inference. Ph.D. thesis, University of Utrecht.
- Buntine, W. (1996). A guide to the literature on learning probabilistic networks from data. *IEEE Transactions on Knowledge and Data Engineering*, 8(2), 195–210.
- Campos, L. M. D. (2006). A scoring function for learning Bayesian networks based on mutual information and conditional independence tests. *Journal of Machine Learning Research*, 7(Oct), 2149–2187.
- Cheng, J., & Greiner, R. (1999, July). Comparing Bayesian network classifiers. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence* (pp. 101–108). Morgan Kaufmann Publishers Inc.
- Chickering, D. M. (2002). Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov), 507–554.
- Chickering, D. M., Geiger, D., & Heckerman, D. (1994). *Learning Bayesian networks is NP-hard* (Vol. 196) (Technical Report MSR-TR-94-17). Microsoft Research.
- Ching, J. Y., Wong, A. K. C., & Chan, K. C. C. (1995). Class-dependent discretization for inductive learning from continuous and mixed-mode data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(7), 641–651.
- Chow, C., & Liu, C. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3), 462–467.
- Cooper, G. F., & Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4), 309–347.
- Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.
- Cover, T. M., & Thomas, J. A. (2012). *Elements of information theory*. Wiley.
- Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan), 1–30.
- Duda, R. O., Hart, P. E., & Stork, D. G. (1973). *Pattern classification* (Vol. 2). New York, NY: Wiley.
- Frank, A., & Asuncion, A. (2010). *UCI machine learning repository*. Irvine, CA: University of California, School of Information and Computer Science, 213. Retrieved from <https://archive.uci.edu/ml>
- Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, 29(2/3), 131–163.
- Friedman, N., Mosenzon, O., Slonim, N., & Tishby, N. (2001, August). Multivariate information bottleneck. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence* (pp. 152–161). Morgan Kaufmann Publishers Inc.
- Friedman, N., Nachman, I., & Pe'er, D. (1999, July). Learning bayesian network structure from massive datasets: the "sparse candidate" algorithm. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence* (pp. 206–215). Morgan Kaufmann Publishers Inc.
- García-Laencina, P. J., Sancho-Gómez, J. L., & Figueiras-Vidal, A. R. (2010). Pattern classification with missing data: A review. *Neural Computing and Applications*, 19(2), 263–282.
- Ghanty, P., Paul, S., & Pal, N. R. (2009). NEUROSVM: An Architecture to Reduce the Effect of the Choice of Kernel on the Performance of SVM. *Journal of Machine Learning Research*, 10(Mar), 591–622.
- Grossman, D., & Domingos, P. (2004, July). Learning Bayesian network classifiers by maximizing conditional likelihood. In *Proceedings of the twenty-first international conference on machine learning* (p. 46). ACM.
- Gruber, A., & Ben-Gal, I. (2012). Efficient Bayesian network learning for system optimization in reliability engineering. *Quality Technology & Quantitative Management*, 9(1), 97–114.
- Gruber, A., Yanovski, S., & Ben-Gal, I. (2013). Condition-based maintenance via simulation and a targeted Bayesian network metamodel. *Quality Engineering*, 25(4), 370–384.
- Heckerman, D., Geiger, D., & Chickering, D. M. (1995). Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3), 197–243.
- Kelner, R., & Lerner, B. (2012). Learning Bayesian network classifiers by risk minimization. *International Journal of Approximate Reasoning*, 53(2), 248–272.
- Keogh, E. J., & Pazzani, M. J. (1999, January). Learning augmented Bayesian classifiers: A comparison of distribution-based and classification-based approaches. In *AISTATS*.
- Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, 97(1–2), 273–324.
- Koller, D., & Friedman, N. (2009). *Probabilistic graphical models: principles and techniques*. MIT press.

- Lam, W., & Bacchus, F. (1994). Learning Bayesian belief networks: An approach based on the MDL principle. *Computational Intelligence*, 10(3), 269–293.
- Madden, M. G. (2009). On the classification performance of TAN and general Bayesian networks. *Knowledge-Based Systems*, 22(7), 489–495.
- Mallows, C. L. (1973). Some comments on C p. *Technometrics*, 15(4), 661–675.
- MATLAB and Statistics Toolbox Release. (2010). Natick, MA: The MathWorks.
- Murphy, K. (2001). The Bayes net toolbox for Matlab. *Computing Science and Statistics*, 33(2), 1024–1034.
- Ng, A. Y., & Jordan, M. I. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. *Advances in Neural Information Processing Systems*, 2, 841–848.
- Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. San Francisco, CA: Morgan Kaufmann.
- Pearl, J. (2003). Causality: Models, reasoning and inference. *Econometric Theory*, 19(675–685), 46.
- Pourret, O., Naïm, P., & Marcot, B. (Eds.). (2008). *Bayesian networks: A practical guide to applications* (Vol. 73). Wiley.
- Quinlan, J. R., & Rivest, R. L. (1989). Inferring decision trees using the minimum description length principle. *Information and Computation*, 80(3), 227–248.
- Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, 14(5), 465–471.
- Roos, T., Wettig, H., Grünwald, P., Myllymäki, P., & Tirri, H. (2005). On discriminative Bayesian network classifiers and logistic regression. *Machine Learning*, 59(3), 267–296.
- Saar-Tsechansky, M., & Provost, F. (2007). Handling missing values when applying classification models. *Journal of machine learning research*, 8(Jul), 1623–1657.
- Schwarz, G. (1978). Estimating the dimension of a model. *The annals of statistics*, 6(2), 461–464.
- Spirtes, P., Glymour, C. N., & Scheines, R. (2000). *Causation, prediction, and search*. MIT Press.
- Tishby, N., Pereira, F. C., & Bialek, W. (2000). *The information bottleneck method*. arXiv preprint physics/0004057.
- Wang, Z., & Chan, L. (2012). Learning causal relations in multivariate time series data. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(4), 76.
- Wang, G., Xie, S., Liu, B., & Yu, P. S. (2012). Identify online store review spammers via social review graph. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(4), 61.
- Williamson, J. (2000, December). Approximating discrete probability distributions with bayesian networks. In *Proc. Intl. Conf. on AI in Science & Technology*, Tasmania (pp. 16–20).
- Xia, R., Zong, C., Hu, X., & Cambria, E. (2013). Feature ensemble plus sample selection: Domain adaptation for sentiment classification. *IEEE Intelligent Systems*, 28(3), 10–18.
- Zhao, K., Xie, Y., Tsui, K. L., Wei, Q., Huang, W., Jiang, W., ... Shi, J. (2015). System informatics: From methodology to applications. *IEEE Intelligent Systems*, 30(6), 12–29.
- Zhu, M., Liu, S., & Jiang, J. (2016). A hybrid method for learning multi-dimensional Bayesian network classifiers based on an optimization model. *Applied Intelligence*, 44(1), 123–148.