Knowledge-Based Systems 22 (2009) 489–495

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

On the classification performance of TAN and general Bayesian networks

Michael G. Madden *

College of Engineering and Informatics, National University of Ireland, University Road, Galway, Ireland

ARTICLE INFO

Article history:
Available online 9 January 2009

Keywords:
Bayesian networks
TAN
Naïve Bayes
Classification
Inductive learning
Parameter estimation

ABSTRACT

Over a decade ago, Friedman et al. introduced the Tree Augmented Naïve Bayes (TAN) classifier, with experiments indicating that it significantly outperformed Naïve Bayes (NB) in terms of classification accuracy, whereas general Bayesian network (GBN) classifiers performed no better than NB. This paper challenges those claims, using a careful experimental analysis to show that GBN classifiers significantly outperform NB on datasets analyzed, and are comparable to TAN performance. It is found that the poor performance reported by Friedman et al. are not attributable to the GBN per se, but rather to their use of simple empirical frequencies to estimate GBN parameters, whereas basic parameter smoothing (used in their TAN analyses but not their GBN analyses) improves GBN performance significantly. It is concluded that, while GBN classifiers may have some limitations, they deserve greater attention, particularly in domains where insight into classification decisions, as well as good accuracy, is required.

1. Introduction

This paper examines the performance of Bayesian networks as classifiers, comparing their performance to that of the Naïve Bayes (NB) classifier and the Tree Augmented Naïve Bayes (TAN) classifier, both of which make strong assumptions about interactions between domain variables.

In the experiments performed for this work, described below in Section 3, standard Bayesian networks (referred to as General Bayesian Networks, GBNs, to distinguish them from NB and TAN) are compared with NB and TAN classifiers on 28 standard benchmark datasets. Our experiments indicate that the GBN classifier is substantially better than NB, with performance closer to that of TAN. This contrasts with the conclusions drawn in the landmark paper on Bayesian network classifiers by Friedman et al. [14]. That paper presented results on many of the same datasets, showing that GBNs constructed using the minimum description length (MDL) score tend to perform no better than NB. That result has been widely noted by other authors (e.g. [16,18]); in one case the result was interpreted as indicating that NB "easily outperforms" GBN.

Our contention is that it has become 'accepted wisdom' that GBN classification performance is no better than that of NB, and significantly worse than TAN (ignoring other considerations such as computational complexity or interpretability). Our results indicate that GBN's classification performance is superior to that of NB and much closer to that of TAN, when the same parameter estimation procedure is used for all.

It turns out that Friedman et al. used simple frequency counts for parameter estimation in constructing GBN classifiers, whereas they used parameter smoothing in constructing TAN classifiers (see Section 2.3 for details). Our experiments show that if frequency counts are used for both GBN and TAN, neither is much better than NB (Section 3.3, Fig. 5), but if parameter smoothing is used for both, they both perform similarly well (Fig. 4). Furthermore, since GBN classifiers are commonly constructed through heuristic search, it is possible for improved GBN construction algorithms to lead to improved performance.

The structure of the paper is as follows. Section 2 reviews Bayesian networks and the algorithms for constructing GBN and TAN classifiers that are used in this paper. Section 3 presents experiments applying NB, TAN and two GBN algorithms to classification problems on 28 standard datasets, and identifies why the results of this paper are at odds with those of Friedman et al. as mentioned above. Finally, Section 4 draws general conclusions about the suitability of GBNs as classifiers.

2. Bayesian networks and classification

As is well known, a Bayesian network is composed of the network structure and its conditional probabilities. The structure B_S is a directed acyclic graph where the nodes correspond to domain variables $x_1,\ldots,x_n$ and the arcs between nodes represent direct dependencies between the variables. Likewise, the absence of an arc between two nodes x_1 and x_2 represents that x_2 is independent of x_1 given its parents in B_S. Using the notation of Cooper and Herskovits [12], the set of parents of a node x_i in B_S is denoted π_i. The structure is annotated with a set of conditional probabilities, B_P,

* Tel.: +35391493797; fax: +35391444214.
E-mail address: michael.madden@nuigalway.ie.

0950-7051/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.knosys.2008.10.006

containing a term $P(X_i|\Pi_i)$ for each possible value X_i of x_i and each possible instantiation Π_i of π_i.

2.1. Inductive learning of Bayesian networks

Several algorithms have been proposed since the late 1980s for inductive learning of general Bayesian networks. Recent developments include the global optimization approach of Silander and Myllymäki [23], the Greedy Equivalence Search algorithm [9], and the Three-Phase Dependency Analysis algorithm [8], though this latter algorithm has subsequently been shown to be incorrect [10]. We evaluate two approaches to GBN construction, described in the following sub-sections, both of which approaches have relatively low computational complexity:

1. The K2 search procedure [12] in conjunction with the Bayesian BDeu scoring metric [5], which is a refinement of the K2 metric.
2. The approach used by Friedman et al. [14], which combines hill-climbing search with the MDL score.

These are both search-and-score methods for construction of GBNs; a search heuristic is used to propose candidate networks, and a scoring function is used to assess, for any two candidates, which one is more likely given the training data.

The scoring functions and search procedures are described in greater detail in the following sub-sections. Rather can constructing general BN structures, restrictions may be placed on the structures; this is described in Section 2.2. Typically, the conditional probabilities (parameters) associated with a network are not computed from the data until after the structure has been found; parameter estimation is described in Section 2.3.

2.1.1. K2: search with BDeu scoring approach

If D is a database of training cases, Z is the set of variables in each case in D, and B_{Si} and B_{Sj} are two belief network structures containing exactly those variables that are in Z, then the comparison amounts to calculating $P(B_{Si}|D)/P(B_{Sj}|D)$, which in turn reduces to calculating $P(B_{Si},D)/P(B_{Sj},D)$.

Assume that Z is a set of n discrete variables, where a variable x_i in Z has r_i possible value assignments, $(v_{i1},\ldots,v_{iri})$, and that D has N cases, each with a value assignment for each variable in Z. A network structure B_S is assumed to contain just the variables in Z. Each variable x_i in B_S has zero or more parents, represented as a list π_i. Let w_{ij} denote the jth unique instantiation of π_i relative to D, and assume that there are q_i such unique instantiations of π_i. Let N_{ijk} be defined as the number of cases in D in which variable x_i has the value v_{ik} and π_i is instantiated as w_{ij}. Let N'_{ijk} denote a Dirichlet parameter. Let N_{ij} and N'_{ij} be defined as:

$$N_{ij} = \sum_{k=1}^{r_i} N_{ijk}, \quad N'_{ij} \equiv \sum_{k=1}^{r_i} N'_{ijk} \tag{1}$$

With these definitions, the BD metric [17] is defined as:

$$P(B_S, D) = P(B_S) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(N'_{ij})}{\Gamma(N'_{ij} + N_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(N'_{ijk} + N_{ijk})}{\Gamma(N'_{ijk})} \tag{2}$$

Note that Γ is the gamma function, defined as $\Gamma(x+1) = x\Gamma(x)$, which is closely related to the factorial function but defined for real numbers, not just integers. In a practical implementation, the logs of terms in Eq. (2) are computed.

The K2 metric [12] corresponds to Eq. (2) with all Dirichlet exponents set to 'uninformative' values of $N'_{ijk} = 1$. Alternative uninformative values are proposed by Buntine [5]:

$$N'_{ijk} = \frac{N'}{r_i q_i} \tag{3}$$

Using Buntine's values, Eq. (2) becomes what Heckerman et al. [17] term the BDeu metric, which has the additional property of being structure-equivalent. This is the metric used in the current work. Assuming that all structures are equally likely *a priori*, $P(B_S)$ is constant, so to maximize $P(B_S,D)$ just requires finding the set of parents for each node that maximizes the second inner product of Eq. (2).

The K2 search procedure requires a node ordering. It operates by initially assuming that a node has no parents, and then adding incrementally that parent whose addition most increases the probability of the resulting network. Parents are added greedily to a node until the addition of no one parent can increase the structure probability. This is repeated for all nodes in the sequence specified by the node ordering.

In the experiments of Section 3, the node ordering in each dataset is arbitrarily taken to be the order of attributes in the input files, except that the class node is always placed first in the order. In addition, the maximum number of parents a node may have is limited to 4.

2.1.2. MDL scoring approach

In constructing GBNs, Friedman et al. [14] use a scoring function based on the minimum description length (MDL) principle. The MDL score of a network B given a database of training cases D is:

$$\mathrm{MDL}(B|D) = \frac{1}{2}\log N|B| - LL(B|D) \tag{4}$$

where $|B|$ is the number of parameters in the network and $LL(B \mid D)$ denotes the log-likelihood of B given D. To calculate $LL(B \mid D)$, let $\hat{P}_D(\cdot)$ be the empirical probability measure defined by frequencies of events in D. Then:

$$LL(B|D) = N \sum_i \sum_{X_i,\Pi_i} \hat{P}_D(X_i, \Pi_i) \log(\hat{P}_D(X_i|\Pi_i)) \tag{5}$$

The search procedure used by Friedman et al. is to start with the empty network and successively apply local operations that greedily reduce the MDL score maximally until a local minimum is found. The local operations applied are arc insertion, arc deletion and arc reversal.

2.1.3. Classification using a GBN

A Bayesian network may be used for classification as follows. Firstly, any nodes outside of the Markov blanket of the classification node x_c may be deleted. Then, assume that the value of x_c is unknown and the values of all other nodes are known. Then, for every possible instantiation of x_c, calculate the joint probability of that instantiation of all variables in the network given the database D. By the definition of a Bayesian network, the joint probability of a particular instantiation of all n variables is calculated as:

$$P(x_1 = X_1, \ldots, x_n = X_n) = \prod_{i=1}^{n} P(x_i = X_i|\pi_i = \Pi_i) \tag{6}$$

By normalizing the resulting set of joint probabilities of all possible instantiations of x_c, an estimate of the relative probability of each is found. The vector of class probabilities may be multiplied by a misclassification cost matrix, if available. Note that the classification node is not considered 'special' when building the GBN, and in Eq. (6), x_c is just one of the variables $x_1,\ldots,x_n$.

Although arbitrary inference in a GBN with discrete variables is NP-hard [11], the classification procedure just described just requires Eq. (6) to be evaluated once for each possible instantiation of x_c; thus its time complexity is $O(n_m r_c)$, where n_m is the number of nodes in x_c's Markov blanket; $n_m \leqslant n$.

2.2. Restricted Bayesian classifiers

Fig. 1 schematically illustrates the structure of the Bayesian classifiers considered in this paper. The simplest form of Bayesian classifier is Naïve Bayes. When represented as a Bayesian network, a Naïve Bayes (NB) classifier has a simple structure whereby there is an arc from the classification node to each other node, and there are no arcs between other nodes, as illustrated in Fig. 1a. Since NB has a fixed structure, learning simply involves estimating the parameters according to one of the procedures discussed below in Section 2.3.

Several researchers have examined ways of achieving better performance than NB. Friedman et al. [14] in particular consider (among other structures) Tree Augmented Naïve Bayes (TAN), which allows arcs between the children of the classification node x_c as shown in Fig. 1b, thereby relaxing the assumption of conditional independence. In their approach, each node has x_c and at most one other node as a parent, so that the nodes excluding x_c form a tree structure. Optimal TAN structures are constructed by finding the maximum weighted spanning tree within a complete graph connecting the nodes, where arcs are annotated by the conditional mutual information between all pairs of non-class nodes, conditioned on the class node, according to Eq. (7).

$$I(x_i, x_j|c) = \sum_{X_i, X_j, C} P(X_i, X_j, C) \log \frac{P(X_i, X_j|C)}{P(X_i|C)P(X_j|C)} \tag{7}$$

2.3. Parameter estimation

Let θ_{ijk} denote the conditional probability that a variable x_i in B_S has the value v_{ik}, for some k from 1 to r_i, given that the parents of x_i, represented by π_i, are instantiated as w_{ij}. Then $\theta_{ijk} = P(x_i = k|\pi_i = w_{ij})$ is termed a network conditional probability. The simplest form of parameter estimation is based on frequency counts (referred to as *unsmoothed* estimates by Friedman et al.):

$$\theta^f_{ijk} = \frac{N_{ijk}}{N_{ij}} \tag{8}$$

A problem with using Eq. (8) is that it can result in zero estimates for some parameters if not all combinations of variables are well represented in the training data, resulting in a probability of 0 being computed for some instantiations of all variables. One solution is to replace zero estimates by a small positive value.

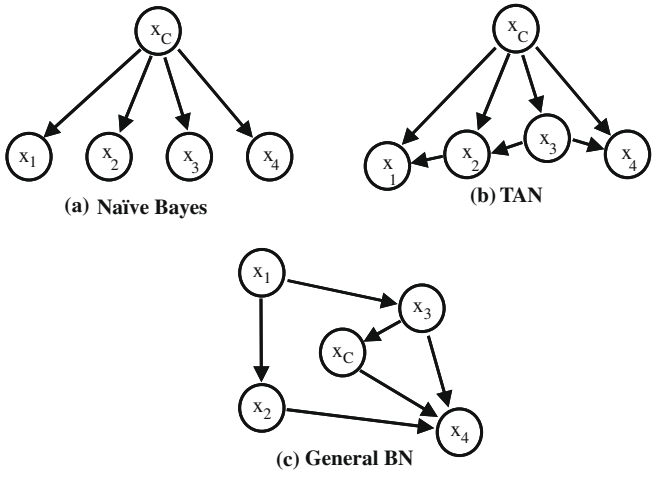

Fig. 1. Illustration of Naïve Bayes, TAN and general BN structures.

As well as using unsmoothed estimates, Friedman et al. use technique based on Dirichlet priors that they term *parameter smoothing*, which boils down to the following calculation:

$$\theta^s_{ijk} = \frac{N_{ijk} + N_0 N_i/N}{N_{ij} + N_0} \tag{9}$$

where $N_i/N = \hat{P}(x_i)$ is the frequency of the given value of x_i observed in the dataset. (Friedman et al. report that, after experimentation, a value of $N_0 = 5$ was chosen.)

As part of our controlled comparisons, the same parameter smoothing is used for all classifiers in the analyses presented below in Section 3.

To avoid any ambiguity, it should be pointed out that smoothed parameter estimates are used only to estimate the conditional probabilities, B_P, after the network structure, B_S, has been determined. TAN and GBN structure learning uses simple frequency counts (Eq. (8)).

3. Experiments

3.1. Methodology

For this work, the Naïve Bayes, TAN and two general BN algorithms were compared using 26 datasets from the UCI repository of Machine Learning datasets [1]. For consistency with previous work in this domain [7,14,18,21], continuous variables were discretized using the discretization utility of MLC++ ([19]) with its default entropy-based setting [13] and any cases with missing values were removed. The two general BN algorithms are those listed earlier:

1. GBN-K2: K2 search procedure with the Bayesian BDeu scoring metric
2. GBN-HC: hill-climbing search with MDL score, following Friedman et al.

Table 1
Classification performance (accuracy ± std dev) of four algorithms as measured on 28 datasets; results in bold are best or joint best, as described in text.

No.	Dataset	Naïve	TAN	GBN-K2	GBN-HC
1	Adult	84.03 ± 0.53	86.15 ± 0.35	**86.16 ± 0.33**	**86.02 ± 0.48**
2	Australian	85.80 ± 4.03	85.06 ± 3.90	86.22 ± 3.83	85.93 ± 4.06
3	Breast cancer	97.38 ± 1.84	96.99 ± 1.88	97.32 ± 1.81	97.15 ± 1.83
4	Car	85.15 ± 2.74	**93.96 ± 1.90**	89.61 ± 2.20	86.36 ± 3.15
5	Chess	87.85 ± 1.70	92.09 ± 1.39	**94.45 ± 1.41**	**94.95 ± 1.47**
6	Cleve	82.87 ± 6.20	81.04 ± 6.77	81.07 ± 6.22	82.33 ± 6.27
7	Connect-4	72.11 ± 0.63	76.43 ± 0.40	**79.08 ± 0.66**	73.88 ± 0.70
8	Corral	87.05 ± 9.46	**99.23 ± 3.19**	**99.62 ± 2.53**	**99.38 ± 2.37**
9	DNA-splice	95.26 ± 0.98	94.92 ± 1.10	**95.93 ± 1.05**	**95.81 ± 1.02**
10	Flare	80.12 ± 3.47	82.65 ± 3.47	82.24 ± 3.39	82.56 ± 3.48
11	German	74.61 ± 4.31	72.07 ± 4.04	74.20 ± 3.97	73.25 ± 4.07
12	Glass2	81.16 ± 8.68	79.37 ± 8.95	79.00 ± 9.35	77.29 ± 9.86
13	Heart	82.74 ± 6.70	83.11 ± 7.30	82.30 ± 7.49	83.04 ± 7.32
14	Hepatitis	86.38 ± 10.97	88.00 ± 11.64	87.00 ± 13.29	86.38 ± 14.22
15	Letter	74.67 ± 1.05	**86.28 ± 0.61**	81.76 ± 0.73	75.12 ± 0.72
16	Lymphography	82.16 ± 10.61	81.07 ± 9.57	77.46 ± 9.47	75.06 ± 10.98
17	Mofin-3-10	85.34 ± 3.43	**91.96 ± 2.63**	86.85 ± 3.56	**93.04 ± 2.86**
18	Nursery	90.29 ± 0.77	**93.30 ± 0.81**	91.18 ± 0.89	91.68 ± 0.82
19	Pima	75.69 ± 4.42	76.37 ± 3.94	76.33 ± 4.26	76.18 ± 4.27
20	Segment	91.27 ± 1.70	**95.27 ± 1.49**	**94.64 ± 1.56**	93.45 ± 1.48
21	Soybean-large	**91.83 ± 3.50**	**92.35 ± 3.08**	89.22 ± 4.22	78.02 ± 6.45
22	Spect	68.53 ± 9.14	**70.29 ± 8.99**	**68.98 ± 8.50**	**74.19 ± 8.89**
23	Tic tac toe	69.76 ± 4.45	**76.32 ± 3.82**	69.26 ± 4.74	68.38 ± 4.83
24	Vehicle	60.62 ± 4.88	**70.36 ± 4.58**	**67.30 ± 5.18**	62.50 ± 5.46
25	Vote	90.27 ± 4.30	**93.84 ± 3.26**	**93.57 ± 3.53**	**95.11 ± 3.03**
26	Waveform-21	**80.90 ± 1.64**	**81.96 ± 1.70**	**81.67 ± 1.56**	79.73 ± 1.96

The GBN-HC implementation used in this work is that in WEKA [3]. The NB, TAN and GBN-K2 algorithms were implemented for this work in Common Lisp (code available by email on request).

Previous comparisons of similar classifiers [7,14,21] have estimated classifier accuracy using holdout sets for the larger datasets and 5-fold cross-validation for smaller datasets. However, it has been shown that such analyses may suffer from high sensitivity to the specific divisions used [4]. Also, previous analyses have compared accuracy figures by simply considering the magnitude of the estimated accuracy without performing statistical significance tests [7,14], or using *t*-tests that are not corrected to account for the overlap in folds from a multi-fold cross-validation run [21]. This latter approach has been shown to have a high Type I error [22].

To avoid such problems, the experimental methodology used in this work follows the 10×10 fold sorted cross-validation approach proposed by Bouckaert [4], with associated *t*-tests to measure significance. This has been shown to have good replicability, thereby facilitating future comparisons, and because by applying it consistently across all datasets and algorithms, coherent comparisons can be drawn.

3.2. Results

Table 1 lists the accuracy (and standard deviation of accuracy) of each of the four classification algorithms being considered, as measured from 10 runs of 10-fold cross-validation on each dataset. In each row, the best of the four classifier results are displayed in bold. Specifically, for each dataset, the classifier with the highest performance is highlighted in bold and compared with that of the other two classifiers, using a paired *t*-test at the 5% significance level based on the 10×10 fold sorted cross-validation results. If another's performance is not significantly different from the best, it is also highlighted, but if the differences between all four classifiers are not statistically significant, then none of them are highlighted.

As these results show, there are no statistical differences between the algorithms on 10 of the 26 datasets, at the 5% significance level. In just 2 other cases, NB is best (including joint best), in 13 cases TAN is best, in 10 cases GBN-K2 is best and in seven cases GBN-HC is best.

Fig. 2 shows two scatter-plots comparing TAN with NB and with GBN-HC. Fig. 2a shows that TAN generally outperforms NB, as was also demonstrated in the experiments of Friedman et al. [14]. Fig. 2b also shows TAN outperforming GBN-HC, though the difference in performance is not as marked as in the results of Friedman et al.

But, what about the claim that GBNs perform as badly as, or even worse than, NB? Fig. 3 shows two scatter-plots comparing GBN-K2 and GBN-HC with NB. In this and subsequent graphs, "A vs. B" indicates that A is plotted on the vertical axis and B is plotted on the horizontal axis. Visually, points above the diagonal are those where classifier A has higher accuracy. Our results do not provide evidence for that claim. They show that the classification perfor-

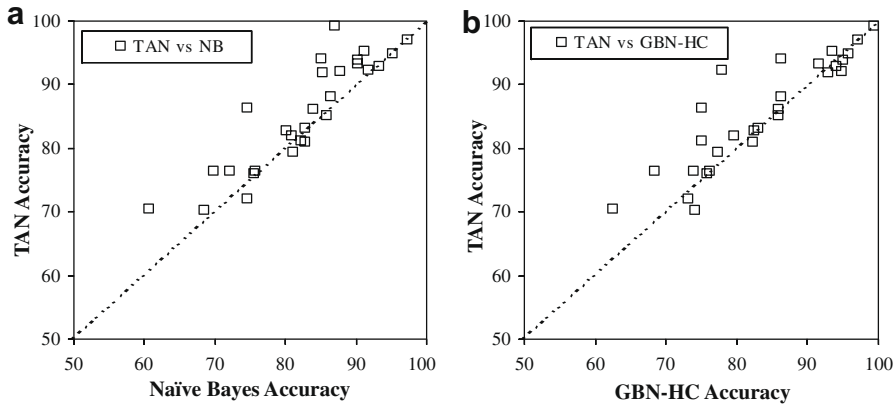

Fig. 2. Relative accuracies of: (a) TAN and NB, (b) TAN and GBN-HC.

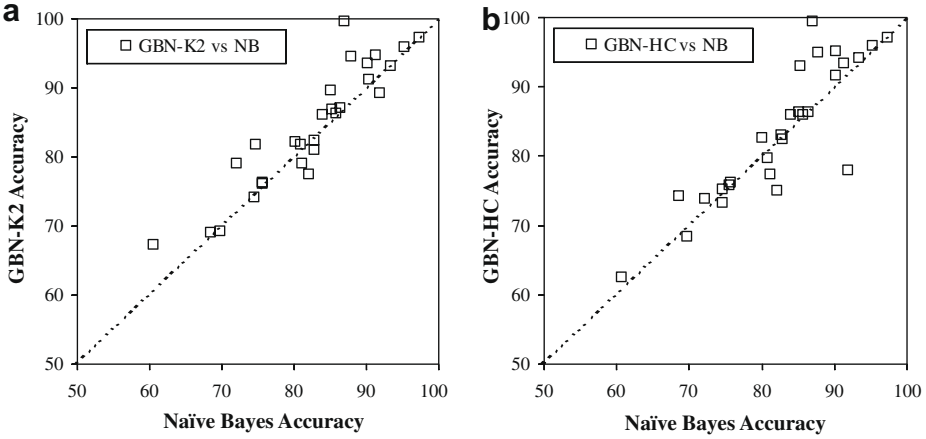

Fig. 3. Relative accuracies of: (a) GBN-K2 and NB, (b) GBN-HC and NB.

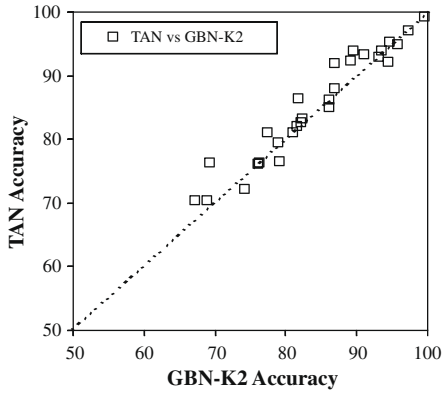

Fig. 4. Relative accuracies of TAN and GBN-K2.

mance of both GBN algorithms is good relative to NB, although the performance of GBN-K2 is a little better than that of GBN-HC. On the basis of paired *t*-tests, it is found that GBN-K2 is better than NB on 11 datasets whereas NB is better than it on just 1; likewise, GBN-HC is better than BN on 9 datasets whereas NB is better on 1.

Furthermore, when GBN-K2, rather than GBN-HC, is compared with TAN, the differences between them are not at all pronounced, as shown in Fig. 4.

3.3. Discussion of results

The results presented in Table 1 and illustrated in Fig. 3 indicate that GBN outperforms NB overall. This conclusion is clearly at variance with the experimental results of Friedman et al., who compared GBN and NB on 25 datasets and reported that GBN was significantly better on 6 and significantly worse on 6. (All of those datasets are included in this study except for CRX and Glass, which are variants of the Australian and Glass2 datasets that are included.) Our GBN-HC algorithm is the same one that they used.

Differences in experimental methodology might account for some of the disparities in conclusions drawn from our work and that of Friedman et al, as their experiments may be more prone to Type I errors and have lower replicability. However, we believe that parameter estimation has a much more significant effect. For the TAN and NB algorithms, they present results using unsmoothed (Eq. (8)) and smoothed (Eq. (9)) parameter estimates. As would be expected, parameter smoothing has little effect on the performance of NB, but it improves the performance of TAN since zero probability estimates are more likely to arise in more complex structures. However, Friedman et al. present results for GBN without smoothing only; they do not present corresponding smoothed GBN results, even though one would expect parameter smoothing to improve the performance of GBN also. In contrast, the results presented above in Table 1 and Figs. 2–4 use parameter smoothing for all classifiers.

To explore this further, we repeated our analyses using unsmoothed parameter estimates. Fig. 5a presents a plot comparing Unsmoothed GBN with Unsmoothed NB. These results are qualitatively similar to those of Friedman et al.; Unsmoothed GBN is not much better than Unsmoothed NB. However, the comparison in Fig. 5b is also interesting, as it shows that Unsmoothed TAN is also no better than Unsmoothed NB.

In a further set of experiments, we used unsmoothed parameter estimates but replaced zero probabilities with small epsilon values. When we did so, the results were quite close to the smoothed result of Table 1. We therefore conclude that the essential cause of the poor performance of the TAN and GBN classifiers relative to NB in Fig. 5 may be attributed to the zero probabilities in the computations.

4. Conclusions: suitability of GBN as a classifier

The results of the preceding section have shown that, when TAN and GBN-K2 classifiers are compared under careful experimental procedures and using the same parameter estimation procedure for both, there is little to distinguish between them in terms of classification accuracy.

An advantage of TAN is its low computational complexity, which is $O(n^2 N)$. However, for a fixed maximum number of parents per node, to complexity of the GBN-K2 algorithm is $O(n^3 N r)$, which is just a factor $(n\ r)$ worse (here, r is the maximum number of different values any node may have).

Nonetheless, if GBN classifiers are more expensive to construct than TAN classifiers and do not offer greater classification accuracy, why use them? There are other possible drawbacks of GBNs as classifiers:

1. It is often observed in Machine Learning that we should not solve a more general problem than required, so why build a full GBN if all that is required is a classifier?
2. GBNs are in general more complex than TAN classifiers, though not necessarily; in fact, as discussed below, A GBN classifier may end up with fewer arcs than a TAN classifier on the same domain, since not all nodes might be within the Markov blanket.

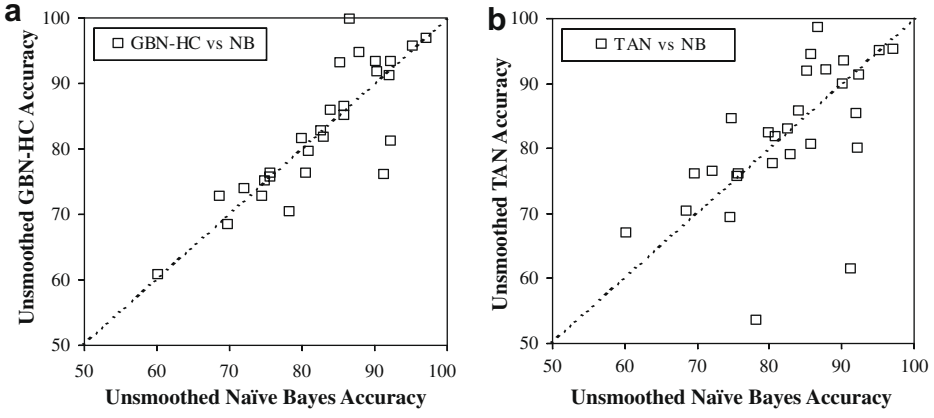

Fig. 5. Relative accuracies of: (a) unsmoothed GBN-HC vs. unsmoothed NB, (b) unsmoothed TAN vs. unsmoothed NB.

3. The GBN that best describes the domain as a whole does not necessarily correspond to the one that best discriminates between classes and the classification node might potentially be unconnected from the network.

While aware of these drawbacks, we propose three reasons for their use:

1. *Insightful analysis:* In many practical domains, particularly where it is required to convince end-users such as scientists or engineers that classification decisions are reasonable and logical, it is as important to gain insight into the problem as it is to achieve high classification accuracy. GBN classifiers support this by modelling the distribution, allowing more complex interactions between nodes to be represented than with TAN and also potentially identifying nodes that are outside the classification node's Markov blanket. They also aid in identifying conditional independencies in the data, which may also be useful for domain insight.
2. *Representational power:* Ling and Zhang [20] have examined the representational power of discrete BNs and have concluded that, if each node has at most u parents, a BN can represent parity functions of maximum order u. This implies, as would be expected, that GBN has greater representational power than TAN which in turn has greater representational power than NB.
3. *Appropriate complexity:* As noted above, a GBN classifier may have fewer arcs than a TAN classifier for the same domain. In TAN, nodes are must have the class node as a parent, and a full tree of arcs between non-class nodes, so all but two nodes have exactly two parents each. In GBN, there are no such constraints; a node may have no parents or several. On the Adult dataset for example, the typical GBN had 13 arcs with 0–3 parents per node, which is the same number of arcs as Naïve Bayes for the dataset, which has exactly one parent per node. The TAN classifier for the Adult dataset was more complex, with 25 arcs. On the Connect 4 dataset, Naïve Bayes has 13 arcs, TAN has 83 arcs and GBN has a median of 74 arcs.

GBN approaches are not as widely used for classification tasks as TAN. Notable exceptions include the work of Cheng and Greiner [7], the application by Baesens et al. [2] of Monte Carlo Markov Chain search to constructing GBN classifiers, and Grossman and Domingos' [16] algorithm for learning GBNs that maximize conditional likelihood.

However, a larger number of researchers have analysed TAN and proposed improvements. Examples include the work of Keogh and Pazzani [18] who proposed the use of classification accuracy rather than conditional mutual information in building TAN-style classifiers; Zhang and Ling [24], who extended Keogh & Pazzani's work by using AUC measures; Cerquides and de Mántaras [6], who identified theoretical weaknesses in the TAN approach and proposed corrections for them; and Garg and Roth [15] who addressed the question of why classifiers such as TAN that make often inaccurate assumptions tend to perform well.

Although the experiments here have shown that the GBN-K2 algorithm has quite good classification performance, it is likely that other algorithms would perform even better. Given the relative complexity of GBN construction compared to TAN construction, improving the performance of GBN classifiers would appear to be a topic with some potential for research. A limitation of GBN-K2 is that it requires an ordering on nodes. In specific applications it may be possible to determine a reasonable node ordering from domain knowledge, but it would be interesting to analyse the performance of other algorithms that do not require node ordering. That being said, GBN-HC does not require node ordering and its performance on the test datasets was slightly weaker than that of GBN-K2, but its simple hill-climbing search without restarts is quite limited.

In the future, it is hoped to analyse more sophisticated algorithms, particularly the algorithm of Silander and Myllymäki [23], which searches for a globally optimal network. In order to address the issue noted earlier in this section that the optimal GBN for a domain is not necessarily the optimal one for classification, it would be necessary to develop an approach that constructs a Markov blanket around the classification node.

Overall, we believe that GBNs may deserve greater attention as classifiers, particularly in problem domains where data is plentiful and insight into the domain, as well as high accuracy, is required, although work remains to be done to optimize them for classification tasks.

Acknowledgements

This research has been supported by a Marie Curie Transfer of Knowledge Fellowship of the EU 6th Framework Programme, contract CT-2005-029611.

References

[1] A. Asuncion, D.J. Newman, UCI Machine Learning Repository, University of California, Irvine, 2007. <http://www.ics.uci.edu/~mlearn/MLRepository.html>.
[2] B. Baesens, M. Egmont-Petersen, R. Castelo, J. Vanthienen, Learning Bayesian network classifiers for credit scoring using Markov Chain Monte Carlo search, in: Proceedings of 2002 International Congress on Pattern Recognition, IEEE Computer Society, 2002.
[3] R.R. Bouckaert, Bayesian networks in Weka, Technical Report 14/2004, Computer Science Department, University of Waikato, 2004.
[4] R.R. Bouckaert, Estimating replicability of classifier learning experiments, in: Proceedings of 21st International Conference on Machine Learning, 2004.
[5] W. Buntine, Theory refinement on Bayesian networks, in: Proceedings of Seventh International Conference on Uncertainty in Artificial Intelligence, 1991.
[6] J. Cerquides, R. de Mántaras, TAN Classifiers Based on Decomposable Distributions, Machine Learning 59 (2005) 323–354.
[7] J. Cheng, R. Greiner, Learning Bayesian belief network classifiers: algorithms and system, in: Proceedings of 14th Canadian Conference on Artificial Intelligence, 2001.
[8] J. Cheng, R. Greiner, J. Kelly, D. Bell, W. Liu, Learning belief networks from data: an information theory based approach, Artificial Intelligence 137 (2002) 43–90.
[9] D.M. Chickering, Optimal structure identification with greedy search, Journal of Machine Learning Research 3 (2002) 507–554.
[10] D.M. Chickering, C. Meek, On the incompatibility of faithfulness and monotone DAG faithfulness, Artificial Intelligence 170 (2006) 653–666.
[11] G.F. Cooper, The computational complexity of probabilistic inference using Bayesian belief networks, Artificial Intelligence 42 (1990) 393–405.
[12] G.F. Cooper, E. Herskovits, A Bayesian method for the induction of probabilistic networks from data, Machine Learning 9 (1992) 309–347. Kluwer Academic Publishers.
[13] J. Dougherty, R. Kohavi, M. Sahami, Supervised and unsupervised discretization of continuous features, in: Proceedings of 12th International Conference on Machine Learning, 1995.
[14] N. Friedman, D. Geiger, M. Goldszmidt, Bayesian network classifiers, Machine Learning 29 (1997) 131–163.
[15] A. Garg, D. Roth, Understanding probabilistic classifiers, in: Proceedings of 12th European Conference on Machine Learning, 2001.
[16] D. Grossman, P. Domingos, Learning Bayesian network classifiers by maximizing conditional likelihood, in: Proceedings of 21st International Conference on Machine Learning, 2004.
[17] D. Heckerman, D. Geiger, D.M. Chickering, Learning Bayesian networks: the combination of knowledge and statistical data, Machine Learning 20 (1995) 197–243.
[18] E. Keogh, M.J. Pazzani, Learning the structure of augmented Bayesian classifiers, International Journal on Artificial Intelligence Tools 11 (4) (2002) 587–601.
[19] R. Kohavi, D. Sommerfield, J. Dougherty, Data mining using MLC++, International Journal on Artificial Intelligence Tools 6 (4) (1997) 537–566.
[20] C.X. Ling, H. Zhang, The representational power of discrete Bayesian networks, Journal of Machine Learning Research 3 (2002).
[21] M.G. Madden, The performance of Bayesian network classifiers constructed using different techniques, in: Proceedings of European Conference on Machine Learning, Workshop on Probabilistic Graphical Models for Classification, 2003.
[22] C. Nadeau, Y. Bengio, Inference for the generalization error, Advances in Neural Information Processing Systems 12 (2000).

[23] T. Silander, P. Myllymäki, A simple approach for finding the globally optimal Bayesian network structure, in: Proceedings of 22nd Conference on Uncertainty in Artificial Intelligence, 2006.

[24] H. Zhang, C.X. Ling, An improved learning algorithm for augmented Naive Bayes, in: Proceedings of Fifth Pacific-Asia Conference on Knowledge Discovery in Databases, 2001.