Unsupervised Training of Bayesian Networks for Data Clustering
Author(s): Duc Truong Pham and Gonzalo A. Ruz
Source: *Proceedings: Mathematical, Physical and Engineering Sciences*, Vol. 465, No. 2109 (Sep. 8, 2009), pp. 2927-2948
Published by: The Royal Society
Stable URL: http://www.jstor.org/stable/30243416
Accessed: 22/02/2014 11:05

PROCEEDINGS
—OF—
THE ROYAL
SOCIETY A

*Proc. R. Soc. A* (2009) **465**, 2927–2948
doi:10.1098/rspa.2009.0065
*Published online* 8 July 2009

# Unsupervised training of Bayesian networks for data clustering

BY DUC TRUONG PHAM AND GONZALO A. RUZ*†

*Manufacturing Engineering Centre, Cardiff University, Cardiff CF24 3AA, UK*

This paper presents a new approach to the unsupervised training of Bayesian network classifiers. Three models have been analysed: the Chow and Liu (CL) multinets; the tree-augmented naive Bayes; and a new model called the simple Bayesian network classifier, which is more robust in its structure learning. To perform the unsupervised training of these models, the classification maximum likelihood criterion is used. The maximization of this criterion is derived for each model under the classification expectation–maximization (EM) algorithm framework. To test the proposed unsupervised training approach, 10 well-known benchmark datasets have been used to measure their clustering performance. Also, for comparison, the results for the $k$-means and the EM algorithm, as well as those obtained when the three Bayesian network classifiers are trained in a supervised way, are analysed. A real-world image processing application is also presented, dealing with clustering of wood board images described by 165 attributes. Results show that the proposed learning method, in general, outperforms traditional clustering algorithms and, in the wood board image application, the CL multinets obtained a 12 per cent increase, on average, in clustering accuracy when compared with the $k$-means method and a 7 per cent increase, on average, when compared with the EM algorithm.

**Keywords: Bayesian networks; clustering; unsupervised training; classification expectation–maximization algorithm; machine learning**

## 1. Introduction

The learning of Bayesian network classifiers from data is commonly performed in a *supervised* manner, meaning that a training set containing examples that have been previously classified by an expert are used to generate the *directed acyclic graph* (DAG) and its *conditional probability table* (CPT). In practice, this can be viewed as having a class label assigned to each example (row) of the dataset. Unfortunately, in many industrial applications of machine learning, it is difficult to obtain a large dataset with classified examples. This is generally due to the fact that a human expert is needed to manually classify each example, of which in many cases there can be thousands, making it an exhausting and time-consuming task. With this in mind, it is desirable to have an alternative

*Author for correspondence (gonzalo.ruz@uai.cl).
†Present address: Faculty of Engineering and Sciences, University Adolfo Ibáñez, Diagonal las Torres 2640, Peñalolén, Santiago, Chile.

*Received* 6 February 2009
*Accepted* 8 June 2009

way of training a classifier with data that has no class label assigned to each example. This approach is known as *unsupervised* training or learning, which can be used for *clustering* purposes. The main goal of clustering is to find the natural groupings of the data. Well-known clustering algorithms are the $k$-means (MacQueen 1967), the fuzzy $C$-means (a fuzzy version of $k$-means; Bezdek 1981), the information theory-based clustering (Roberts *et al.* 2000), the expectation–maximization (EM) algorithm (Dempster *et al.* 1977) and neural network models such as Kohonen's self-organizing maps (SOM; Kohonen 1995), the adaptive resonance theory (ART; Carpenter & Grossberg 1987) and the fuzzy min–max clustering neural network (Simpson 1993). A recent review of clustering techniques can be found in Pham & Afify (2007) and Kerr *et al.* (2008). It is important to mention, at this point, that unsupervised training will usually obtain lower performances when compared with models trained in a supervised way, primarily owing to the lack of the class information during the learning process. However, it is still a very valuable tool for data exploration and preliminary classification, which can be improved later on once a training set with class labels for each example is built. This task can be carried out by a human expert with the assistance of clustering results, making it a less difficult task.

Although Bayesian network classifiers have become an active research topic in recent years (Grossman & Domingos 2004; Acid *et al.* 2005; Cerquides & Lopez de Mantaras 2005; Hwang & Zhang 2005; Li *et al.* 2007; Jing *et al.* 2008), most of the efforts have been concentrated on developing supervised learning algorithms. Less work has been reported on the unsupervised training of Bayesian network classifiers. In what follows, previous work done on the unsupervised training of Bayesian network classifiers is discussed.

In Barash & Friedman (2002), a clustering technique called *context-specific independences* (CSI) is used for clustering genes based on a combination of genomic and genetic data. The CSI is a refinement of the selective Bayesian models, which are essentially a special subclass of Bayesian networks, specifically, a naive Bayes classifier with the difference that not all the attributes are class dependant. Learning for these models is performed by the *Bayesian structural EM* (*BSEM*) *algorithm* (Friedman 1998), which is capable of learning parameters and structures in an unsupervised way using a scoring function such as the *Bayesian information criterion* or the *Cheeseman–Stutz* (Neapolitan 2004).

An extension of model averaging (MA) with naive Bayesian classifiers (Dash & Cooper 2002) for clustering problems is presented in Santafé *et al.* (2006*a*). To accomplish this, the expectation MA (EMA) algorithm is introduced, which incorporates the MA calculations in the maximization step of the EM algorithm. Tests carried out on synthetic data and DNA microarray data show that the EMA algorithm is a powerful learning algorithm that can be useful for clustering problems where there are many attributes and only a few examples. An extension made to the EMA algorithm to learn tree-augmented naive Bayes (TAN) models is presented in Santafé *et al.* (2006*b*).

A heuristic algorithm for learning Bayesian networks for clustering is shown in Peña *et al.* (1999). The approach is based upon improving the naive Bayes model by means of constructive induction, which is the process of changing the representation of the examples in the database by creating new attributes from existing attributes. With this, some violation of conditional independence assumptions made by the naive Bayes model are detected, and dependencies among attributes are included in the model. The parameter search is performed either by the standard EM algorithm or a hybridization of the bound and collapse method and the EM algorithm (BC + EM), which results in a technique that exhibits a faster convergence rate and a more effective behaviour than the EM algorithm. In Peña *et al.* (2000), the BC + EM method is also used to improve the BSEM algorithm for learning Bayesian networks for clustering.

Estimation of distribution algorithms for the unsupervised training of Bayesian networks, both directly and within the framework of the BSEM algorithm, is proposed and empirically evaluated in Peña *et al.* (2004). An application of the proposed method to gene expression data clustering showed that the identified clusters, after being validated, may be biologically meaningful.

In this paper, an unsupervised training approach following the classification EM (CEM) algorithm (Celeux & Govaert 1992) framework is proposed for three types of Bayesian network classifiers: the Chow and Lui (CL) multinets; the TAN model; and the simple Bayesian network (SBN) classifier, which is more robust in its structure, capable of handling the trade-off between complexity (number of edges in the network) and accuracy using a Bayesian approach. The unsupervised training technique maximizes the classification maximum likelihood (CML) of the Bayesian network classifiers, instead of using the traditional EM approach that maximizes the maximum likelihood (ML). Results on 10 benchmark datasets and on a real-world application, the clustering of wood defects, are presented, together with a comparison with traditional clustering algorithms.

The paper is organized as follows. Section 2 presents a description of Bayesian network classifiers. The proposed unsupervised training via the CEM framework is developed in §3. A description of the experiments carried out to test the proposed method as well as the description of the application to an industrial problem is provided in §4. The results are presented and discussed in §5. The conclusions of the paper are stated and suggestions for future work made in §6.

## 2. Background

Bayesian networks were introduced by Pearl (1988). They encode the joint probability distribution of a finite set $\boldsymbol{A} = \{X_1, \ldots, X_n\}$ of $n$ discrete random attributes as a DAG. Because a Bayesian network satisfies the *Markov condition*, the joint probability distribution (jpd) can be computed as

$$P_{\mathrm{B}}(X_1, \ldots, X_n) = \prod_{i=1}^{n} P(X_i | \Pi_{X_i}), \tag{2.1}$$

where $\Pi_{X_i}$ represents the set of parents of $X_i$ in the network.

In classification tasks, the idea in probabilistic classification is to find the class value that maximizes the posterior probability of the class for a given set of assignments to the attributes. In other words, the class value for the $r$th example of the dataset, $X_1 = x_1^r, X_2 = x_2^r, \ldots, X_n = x_n^r$, can be obtained as

$$\text{class\_value } (X_1 = x_1^r, \ldots, X_n = x_n^r) = \arg\max_k P(C = k | X_1 = x_1^r, \ldots, X_n = x_n^r), \tag{2.2}$$

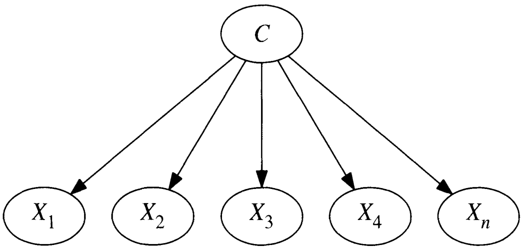

Figure 1. Bayesian network representation of the naive Bayes classifier.

and by using Bayes' theorem, the posterior probability can be computed as

$$P(C=k|X_1,\ldots,X_n)=\frac{P(C=k)P(X_1,\ldots,X_n|C=k)}{\sum_{k'}P(C=k')P(X_1,\ldots,X_n|C=k')}. \tag{2.3}$$

On the r.h.s. of equation (2.3), the denominator is constant with respect to the class and can be expressed as $1/\beta$. So, the main challenge is how to compute the numerator. The simplest approach is to assume that each attribute is conditionally independent of every other attribute. This rather 'naive' assumption yields the well-known naive Bayesian classifier (Duda & Hart 1973; Langley *et al.* 1992),

$$P(C|X_1,\ldots,X_n)=\beta P(C)\prod_{i=1}^{n}P(X_i|C). \tag{2.4}$$

Equation (2.4) has a Bayesian network representation because it is a special case of equation (2.1), where the class node $C$ is the vertex and there is an edge (arc) from $C$ to each attribute $X_i$, as can be seen in figure 1.

Another approach is to use CL's tree algorithm (Chow & Liu 1968). In this case, the training set needs to be partitioned into groups that have examples belonging to the same class. Then, for each one of these groups, a Bayesian network is learned, with the restriction that each attribute has only, at most, one other attribute as a parent, yielding tree structures that have $n-1$ edges. The learning algorithm for this type of Bayesian network is as follows. The initial step is to compute the *mutual information* between each pair of attributes, defined as

$$I(X;Y)=\sum_{x\in X}\sum_{y\in Y}P(x,y)\log\frac{P(x,y)}{P(x)P(y)}. \tag{2.5}$$

Basically, this function measures the amount of information that $Y$ provides about $X$. It is important to point out that, in this work, the marginal and conditional probabilities are estimated by the empirical frequencies from the data.

The following step is to build a complete undirected graph. This is carried out by connecting an edge from each node (attribute) to every other node and assigning the weight of the edge that connects $X_i$ with $X_j$ by $I(X_i;X_j)$. Next, in order to obtain a tree structure, the *maximum-weighted spanning tree* (MWST) is built using any well-known MWST procedures, such as Kruskal's algorithm

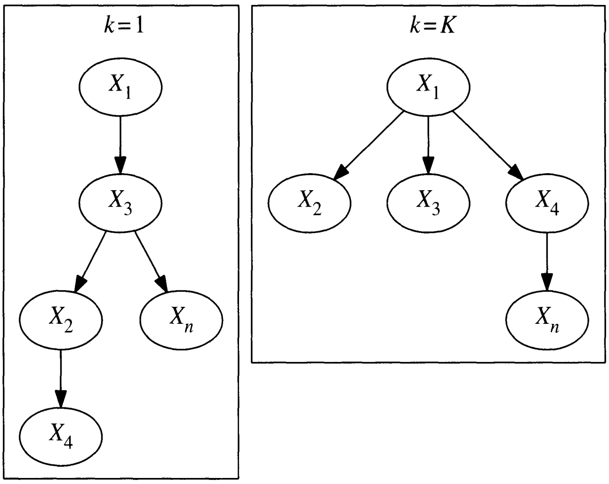

Figure 2. CL multinet classifier representation.

(Kruskal 1956). Finally, directions to the edges of the resulting tree can be added by choosing any attribute as the root and then setting the directions of all edges to be pointing outwards from it.

If there are $K$ classes, then $K$ CL trees are learned, one for each group of data (with the same class value). Each tree distribution, $P_k$ with $k=1,\dots,K$, will be approximating the jpd of the attributes, given a specific class, $P(X_1,\dots,X_n|C=k)=P_k(X_1,\dots,X_n)$. As CL trees are Bayesian networks, the joint probability distribution can be computed using equation (2.1), where $\Pi_{X_i}=\{X_{j\neq i}\}$ will only have one attribute, except for the root attribute node that will have no parents $\Pi_{X_{\text{root}}}=\{\emptyset\}$. Then, the CL multinet classifier can be expressed as

$$P(C=k|X_1,\dots,X_n)=\beta P(C=k)\prod_{i=1}^{n}P_k(X_i|\Pi_{X_i}), \tag{2.6}$$

and a representation can be seen in figure 2.

The next classifier to be analysed is the TAN classifier. This model aims to overcome the strong independence assumption that the naive Bayesian classifier imposes among the attributes in order to obtain equation (2.4). The improvement is accomplished by augmenting the naive Bayes model with edges via a modification of the CL tree algorithm. The main difference with the CL tree procedure is that the TAN model uses the *conditional mutual information*, defined as

$$I(X;Y|Z)=\sum_{x\in X}\sum_{y\in Y}\sum_{z\in Z}P(x,y,z)\log\frac{P(x,y|z)}{P(x|z)P(y|z)}. \tag{2.7}$$

The conditional mutual information measures the information that $Y$ provides about $X$ when the value of $Z$ is known. By using this function, a unique tree structure is obtained, instead of the CL multinet that builds a tree for each class.

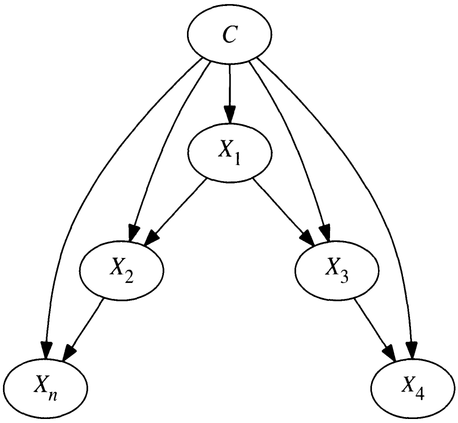

Figure 3. TAN classifier representation.

In this model, each attribute will have $\Pi_{X_i} = \{X_{j \neq i}, C\}$, except for the root attribute node that will have $\Pi_{X_i} = \{C\}$. Then, the TAN classifier can be expressed as

$$P(C|X_1, \ldots, X_n) = \beta P(C) \prod_{i=1}^{n} P(X_i|\Pi_{X_i}), \tag{2.8}$$

and its Bayesian network representation appears in figure 3.

The last model described in this section is a new classifier called the SBN, which follows the TAN model procedure, but is not restricted to tree structures ($n-1$ edges). In fact, the complexity of the network (number of edges) is automatically regulated during the training process, obtaining structures that can have a number of edges $e$ that range from 0 (naive Bayes model) up to $n-1$ (TAN model). The key difference with the standard TAN procedure is that, in each iteration of the MWST, a Bayesian measure with the selected edge to be added must be computed and checked if the stopping criterion for the MWST algorithm is met. If the stopping criterion is met before the MWST algorithm adds the final edge, then the resulting structure will have $e < n-1$ edges. The derivation of the Bayesian measure and the stopping criterion appears in the appendix.

Although the SBN model uses equation (2.8) to carry out the classification, it is worthwhile pointing out that the structure of the network may not be a tree. Thus, if the resulting structure has $e < n-1$ edges, then there are only $e$ attributes with $\Pi_{X_i} = \{X_{j \neq i}, C\}$ and all the rest of the attributes will have $\Pi_{X_i} = \{C\}$, including the root attribute. An illustration of this type of Bayesian network classifier is shown in figure 4.

For the three Bayesian network classifiers described in this section, the learning procedure for each one has time complexity $O(n^2 N)$, associated with the first step (computation of the mutual information/conditional mutual information between each pair of attributes) of the learning procedure; for more details, see Friedman *et al.* (1997).

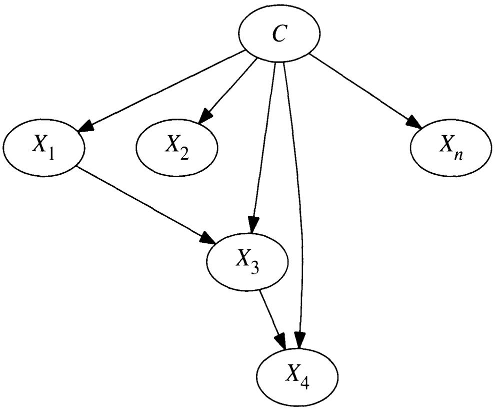

Figure 4. SBN classifier representation.

## 3. Proposed unsupervised training approach

Unsupervised training of the CL multinets, TAN and the SBN classifier described in §2 will be carried out assuming that the data examples have been generated from a mixture of $K$ Bayesian networks. This mixture can be described by

$$\left.\begin{aligned} P(X) &= \sum_{k=1}^{K} \alpha_k f_k(X) \\ \text{and} \quad \sum_{k=1}^{K} \alpha_k &= 1, \quad \alpha_k \geq 0, \end{aligned}\right\} \tag{3.1}$$

where the $\alpha_k$ are the mixing coefficients (weights) and the $f_k$ are the mixing component distributions (defined by the Bayesian networks).

Although CL trees are Bayesian networks, a mixture model of them is not a Bayesian network, whereas by learning networks with the same structure, the resulting mixture model is a unique Bayesian network, which is the case of the TAN model and the SBN classifier. Learning with mixtures of trees for density estimation and classification tasks using the EM algorithm has been developed in Meilă & Jordan (2000), but not for clustering tasks as in this work. As mentioned before, the interest in this work is to learn the Bayesian network classifiers, as well as to cluster the data examples. For this, the mixture parameters $\theta$ (mixing coefficients as well as the parameters of the mixing distributions) and the indicator vectors $\boldsymbol{z}^r = (z_k^r, k = 1, \ldots, K)$, with $z_k^r = 1$ or 0 if $\boldsymbol{x}^r (1 \leq r \leq N)$ has been drawn from the $k$th component or not, will be chosen to maximize the CML criterion (Celeux & Govaert 1995),

$$\mathrm{CL}(\theta, \boldsymbol{z}^1, \ldots, \boldsymbol{z}^N | \boldsymbol{x}^1, \ldots, \boldsymbol{x}^N) = \sum_{k=1}^{K} \sum_{x^r \in \mathrm{P}_k} \log f_k(\boldsymbol{x}^r) + \sum_{k=1}^{K} n_k \log \alpha_k, \tag{3.2}$$

where $\mathrm{P}=\{\mathrm{P}_1,\ldots,\mathrm{P}_k\}$ is a partition (cluster) of the $N$ data examples $\boldsymbol{x}^1,\ldots,\boldsymbol{x}^N$ associated to the indicator vectors $\boldsymbol{z}^1,\ldots,\boldsymbol{z}^N:\mathrm{P}_k=(\boldsymbol{x}^r|z_k^r=1)$ and $n_k=\#\mathrm{P}_k$ $(1\leq k\leq K)$. Equation (3.2) can be optimized by a classification version of the EM algorithm called the CEM algorithm (Celeux & Govaert 1992).

The general outline of the unsupervised training method for the three Bayesian network models, under the CEM framework, is as follows.

Starting from an initial partition $\mathrm{P}^0$, the $m$th $(m>0)$ iteration consists of:

*E-step.* Compute, for $r=1,\ldots,N$ and $k=1,\ldots,K$, the current posterior probabilities that $\boldsymbol{x}^r$ belongs to $\mathrm{P}_k$.
*C-step.* Update the partition by assigning each $\boldsymbol{x}^r$ to the cluster that provides the maximum posterior probability. Let $\mathrm{P}^m$ denote the new partition.
*M-step.* For $k=1,\ldots,K$, in the case of CL multinets, and $\forall k$ simultaneously, in the case of the TAN and the SBN model, compute the ML estimates of $\theta^m$using the cluster $\mathrm{P}_k^m$ as subsamples.

The following subsections will show how to compute the above outline for the CL multinets and the TAN and SBN classifiers.

### (*a*) *Unsupervised training for Chow and Liu multinets classifier*

Starting from an initial partition $\mathrm{P}^0$, the E-step in the $m$th iteration for $r=1,\ldots,N$ and $k=1,\ldots,K$ consists of estimating the probability of each tree generating data point $\boldsymbol{x}^r$, i.e. the posterior probability that $\boldsymbol{x}^r$ belongs to $\mathrm{P}_k$. This posterior probability is computed by

$$t_k^m(\boldsymbol{x}^r)=\frac{\alpha_k^m\prod_{i=1}^n P_k^m(x_i^r|\Pi_{x_i^r})}{\sum_{k'=1}^K\alpha_{k'}^m\prod_{i=1}^n P_{k'}^m(x_i^r|\Pi_{x_i^r})},\tag{3.3}$$

where $\Pi_{x_i^r}$ is the value of $\Pi_{X_i}$ in the $r$th example. If $X_{j(i)}$, with $j\neq i$, is the parent of $X_i$, then $\Pi_{x_i^r}=x_{j(i)}^r$ in equation (3.3). Also, note that the $\alpha_k$ are, in fact, the probability distributions of the variable class $C$, i.e. $P(C=k)$. It is important not to confuse the notation $P_k$, the probability distribution of the $k$th tree, with $\mathrm{P}_k$, which is the $k$th partition (cluster). In the C-step, each $\boldsymbol{x}^r$ is assigned to the cluster $k$ that provides the maximum posteriori probability (3.3), $1\leq k\leq K$. If the maximum posteriori probability is not unique, assign $\boldsymbol{x}^r$ to the cluster with the smallest index. Let the resulting partition be denoted by $\mathrm{P}^m$. The initial iterations of the CEM algorithm may not be so reliable owing to the dependence on the initial partition, causing a convergence to local optima of the CML function. In Celeux & Govaert (1992), a way of reducing this problem was shown by replacing the C-step by a stochastic step called the S-step. In the S-step, each $\boldsymbol{x}^r$ is assigned at random to one of the clusters $\mathrm{P}_1,\ldots,\mathrm{P}_K$ with probability $t_k^m(\boldsymbol{x}^r), k=1,\ldots,K$. The resulting partition is denoted by $\mathrm{P}^m$. The idea, in general, is to start off by using the S-step up to a certain number of iterations, defined by the user, and then to swap the S-step for the C-step for the final iterations. The M-step for $k=1,\ldots,K$ consists of maximizing the CML criteria using the subsamples $\mathrm{P}_k^m$.

Equation (3.2) under the mixtures of CL trees can be expressed as

$$\mathrm{CL} = \sum_{k=1}^{K} \sum_{x^r \in \mathrm{P}_k} \log \prod_{i=1}^{n} P_k(x_i^r | \Pi_{x_i^r}) + \sum_{k=1}^{K} n_k \log \alpha_k. \tag{3.4}$$

The mixture parameters in equation (3.4) are disjoined, so each sum on the r.h.s. of equation (3.4) can be maximized independently with respect to the part of the model on which it depends. The maximization of the second term of equation (3.4), taking into account the constraint

$$\sum_{k=1}^{K} \alpha_k = 1,$$

yields

$$\alpha_k^{m+1} = \frac{n_k}{N} = \frac{\#\mathrm{P}_k^m}{N}, \quad \text{for } k = 1, \ldots, K. \tag{3.5}$$

The maximization of the first term of equation (3.4) will result in a new tree distribution $P_k^{m+1}$(conditional probabilities and tree structure). To achieve this, for each $k$, the following expression needs to be maximized:

$$\mathrm{LL} = \sum_{x^r \in \mathrm{P}_k} \sum_{i=1}^{n} \log P_k(x_i^r | \Pi_{x_i^r}). \tag{3.6}$$

It is recalled that there are $n_k$ data points (examples) in partition $\mathrm{P}_k$, so equation (3.6) is written as

$$\mathrm{LL} = \sum_{r=1}^{n_k} \sum_{i=1}^{n} \log P_k(X_i = x_i^r | \Pi_{X_i} = \Pi_{x_i^r}), \tag{3.7}$$

with $\boldsymbol{x}^r$ for $r = 1, \ldots, n_k$ belonging to partition $\mathrm{P}_k$ . Equation (3.7) is the *log likelihood* (LL) of the model ($k$th tree) given the data, and it is maximized by selecting a tree and its associated conditional probabilities. For any given tree $k$, equation (3.7) is maximized if an empirical distribution $\hat{P}$ defined by the frequencies of events in the data is used as the estimate of the conditional probabilities. This is because $\hat{P}$ is an ML estimator for $P$. By using $\hat{P}$ in equation (3.7), and interchanging the orders of summation, a decomposition of the LL (Friedman *et al.* 1997) according to the $k$th tree can be expressed as

$$\mathrm{LL} = n_k \sum_{i=1}^{n} \sum_{\substack{x_i \in X_i \\ \Pi_{x_i} \in \Pi_{X_i}}} \hat{P}_k(x_i, \Pi_{x_i}) \log \hat{P}_k(x_i | \Pi_{x_i}). \tag{3.8}$$

Then, with some manipulation, equation (3.8) becomes

$$n_k \sum_{i=1}^{n} \sum_{\substack{x_i \in X_i \\ \Pi_{x_i} \in \Pi_{X_i}}} \hat{P}_k(x_i, \Pi_{x_i}) \log \frac{\hat{P}_k(x_i, \Pi_{x_i})}{\hat{P}_k(x_i)\hat{P}_k(\Pi_{x_i})} + n_k \sum_{i=1}^{n} \sum_{\substack{x_i \in X_i \\ \Pi_{x_i} \in \Pi_{X_i}}} \hat{P}_k(x_i, \Pi_{x_i}) \log \hat{P}_k(x_i)$$

$$= n_k \sum_{i=1}^{n} \hat{I}_k(x_i; \Pi_{x_i}) - n_k \sum_{i=1}^{n} \hat{H}_k(x_i), \tag{3.9}$$

where the second expression on the r.h.s. of equation (3.9) is independent of the tree structure and $\hat{I}_k$ is the mutual information computed by the empirical distributions defined by the frequencies of events in the data of $\mathrm{P}_k$. By defining $X_{j(i)}$, with $j \neq i$ as the parent of $X_i$, the tree that must be chosen in order to maximize equation (3.6) is the one that maximizes the r.h.s. of equation (3.9). This can be accomplished by using the MWST algorithm with $\hat{I}_k(x_i; x_{j(i)})$ as branch (edge) weights.

### (*b*) *Unsupervised training for the tree-augmented naive Bayes classifier*

In this case, the tree distributions $P_k$ for $k = 1, \ldots, K$ have the same structure. The E-step is the same as before. The posterior probabilities can be computed using equation (3.3). Also, the S-step and the C-step remain the same. The M-step consists of maximizing the CML criterion (3.4). As before, the mixture parameters are disjoined, so maximization of the second term of equation (3.4) gives the same results as equation (3.5). The maximization of the first term is different from before because all the trees in the mixture model now have the same structure. This constraint implies that the maximization procedure cannot be performed separately for each one of the $K$ trees. The maximization needs to be carried out simultaneously for all the $K$ trees.

Using the result obtained previously in equation (3.9), the first term of equation (3.4) can be expressed as

$$\sum_{k=1}^{K} n_k \left[ \sum_{i=1}^{n} \hat{I}_k(x_i; \Pi_{x_i}) - \sum_{i=1}^{n} \hat{H}_k(x_i) \right]$$

$$= \sum_{k=1}^{K} N\alpha_k \sum_{i=1}^{n} \hat{I}_k(x_i, \Pi_{x_i}) + \text{a constant term independent of the structure.} \tag{3.10}$$

Now, let $C$ be defined as a variable whose values, for $r = 1, \ldots, N$, are obtained from the indicator vector as $\boldsymbol{c}^r = (k|z_k^r = 1)$. In other words, $C$ contains the cluster label for each data point. Also, it is recalled that the $\alpha_k$ are the probability distributions of the cluster/class variable $C$, i.e. $P(C = k)$

and that $P_k(\boldsymbol{X}) = P(\boldsymbol{X}|C=k)$; then, equation (3.10) can be written as

$$N\sum_{i=1}^{n}\sum_{\substack{x_i\in X_i\\ \Pi_{x_i}\in\Pi_{X_i}\\ c\in C}}\hat{P}(c)\frac{\hat{P}(x_i,\Pi_{x_i},c)}{\hat{P}(c)}\log\frac{\hat{P}(x_i,\Pi_{x_i}|c)}{\hat{P}(x_i|c)\hat{P}(\Pi_{x_i}|c)}+\text{const.}$$

$$=N\sum_{i=1}^{n}\hat{I}(x_i;\Pi_{x_i}|c)+\text{const.},\qquad(3.11)$$

where the second expression on the r.h.s. of equation (3.11) is independent of the tree structure and $\hat{I}$ is the conditional mutual information computed by the empirical distributions defined by the frequencies of events in the entire dataset. By defining $X_{j(i)}$, with $j\neq i$, as the parent of $X_i$, the tree that must be chosen is the one that maximizes the r.h.s. of equation (3.11). This can be accomplished by using the MWST algorithm with the conditional mutual information $\hat{I}(x_i;x_{j(i)}|c)$ as branch (edge) weights.

### (*c*) *Unsupervised training for the simple Bayesian network classifier*

This model follows the same procedure as the TAN classifier. The only difference is when maximizing equation (3.11), because the MWST is constructed branch by branch, and for every branch added it needs to check if the stopping criterion is met (see appendix).

## 4. Experimental methods

To test the three Bayesian network classifiers using the unsupervised training approach described in the previous section, 10 benchmark datasets from the University of California, Irvine (UCI) machine learning repository (Asuncion & Newman 2007) were used, as well as a dataset from a real industrial application: wood defect classification (Estévez *et al.* 2003).

### (*a*) *Benchmark datasets*

A brief description of the 10 datasets used in this study appears in table 1, and although the UCI machine learning repository maintains more than 100 datasets, the 10 selected for this paper are commonly found in other machine learning studies, such as Friedman *et al.* (1997) and Gurwicz & Lerner (2006). Adopting those datasets allowed comparisons with previous studies.

Because Bayesian networks are trained using discrete random variables, continuous attributes must be converted to discrete values. It is important to point out that the number of bins used (this will define the number of values for each attribute) in the discretization process has a direct effect on computational costs, the more bins, the greater the computational effort needed. This is because the number of parameters of a Bayesian network, CPTs, increases with the number of parents per node and the number of values that each node can take, therefore requiring more memory storage. Thus, there is a trade-off between the accuracy (number of bins) and the computational tractability,

Table 1. Description of the benchmark datasets.

| dataset | no. of attributes | no. of clusters | no. of examples |
|---|---|---|---|
| corral | 6 | 2 | 128 |
| crx | 15 | 2 | 653 |
| diabetes | 8 | 2 | 768 |
| flare | 10 | 2 | 1066 |
| glass | 9 | 6 | 214 |
| iris | 4 | 3 | 150 |
| lymphography | 18 | 4 | 148 |
| vote | 16 | 2 | 435 |
| wine | 13 | 3 | 178 |
| zoo | 16 | 7 | 101 |

which the user must decide. There are several discretization techniques. Most are supervised methods; among them, probably the most used for Bayesian networks is the entropy-based discretization technique by Fayyad & Irani (1993). On the other hand, fewer unsupervised methods have been reported, equal-width interval binning and equal-frequency interval binning being the only two well-known unsupervised techniques (Dougherty *et al.* 1995). A description of supervised and unsupervised discretization techniques, including the ones mentioned above, can be found in Dougherty *et al.* (1995). More advanced unsupervised discretization techniques can be found in Biba *et al.* (2007) and in Schmidberger & Frank (2005). In this work, because the Bayesian networks use an unsupervised training method, equal-width interval binning, which is considered one of the simplest discretization methods available, was employed, to discretize the continuous attributes into equally sized bins. The number of bins was chosen as 5. This gave sufficient resolution without creating undue computational burdens.

As mentioned before, the CEM algorithm can converge to a local optimum, which sometimes is far from the optimum solution. In order to reduce this problem, the algorithm starts off by using the S-step for 200 iterations. Then, the best solution (maximum value) according to the CML function is selected as the starting point for the CEM algorithm using the C-step, and this second stage usually converges in no more than 10 iterations. In addition to that described previously, a multi-start strategy was also used to reduce convergence to local optima solutions. This strategy consists of starting from $\rho$ different initial points (partition $\mathrm{P}^0$) and then selecting the model that finally scores the highest CML value. In the experiments conducted in this work, $\rho = 10$ was used. The performance of the models is evaluated by computing the correct clustering percentage, and this is carried out by voting using the original class label information. For each dataset, the performance is obtained as the mean value of five repetitions. These results will be compared with the standard clustering techniques: $k$-means and the EM algorithm (both with 10 repetitions). Also, the results obtained by the three Bayesian network classifiers with supervised learning will be included. The CL trees and TAN results are the ones reported in Friedman *et al.* (1997) with no smoothing parameters, and the results for the datasets wine and zoo were the ones reported in Gurwicz & Lerner (2006). The three Bayesian network models, trained in a supervised way, use five-fold cross validation.

### (*b*) *Wood defect classification*

These data were generated from a low-cost automatic visual inspection (AVI) system for wood defect detection (Estévez *et al.* 2003). The AVI systems usually include the following stages (Pham & Alcock 2003): (i) *image acquisition*: to obtain an image of the object to be inspected, (ii) *image enhancement*: to improve the quality of the acquired image, which facilitates later processing, (iii) *image segmentation*: to divide the image into areas of interest and background. The result of this stage is called the segmented image, where *objects* represent the areas of interest, (iv) *feature extraction*: to calculate the values of parameters that describes each object, and (v) *classification*: to determine what is represented by each object.

The feature extraction module from the AVI system in Estévez *et al.* (2003) extracted features from objects and windows of $64 \times 64$ pixels centred in the object geometrical centre. The features used in this work include: 7 object geometrical features measured on the binarized grey image (e.g. area, perimeter, average radius, aspect ratio, etc.); 96 object colour features (24 features measured in each of the four channels, red, green, blue and grey); 46 window colour features (e.g. mean and variance of window histograms, mean and variance at the edge of windows); and 16 co-occurrence features (which contain texture information) that were added recently in Ruz *et al.* (2009). In total, there are 165 features computed from the segmented defects. The dataset used in this work consists of 2247 examples (wood board images of $320 \times 240$ pixels) that have been manually labelled/classified into one of the following 10 defect categories (see Ruz *et al.* 2005, 2009 for details): birdseye; pockets; wane; split; stain; blue stain; pith; dead knot; live knot; and hole. Also, the clear wood category was added, which corresponds to non-defective wood board images. In total, there are 11 classes, with approximately 200 examples per class.

For this dataset, the CL multinet clustering performance was analysed and compared with the $k$-means and the EM algorithm. As this dataset contains a significantly larger amount of attributes compared with the standard benchmark dataset described before, to speed up the learning process for the CL multinet, the results presented are the average value of five repetitions with $\rho = 5$ and 50 iterations for the S-step and 5 iterations for the C-step.

### (*c*) *Initialization*

To obtain the initial partition $\mathrm{P}^0$, $K$ points are selected randomly from the dataset, where $K$ is the number of clusters that will be formed. Then, by computing the Euclidean distance between each data point from the dataset with each one of the $K$ points, assign the data point to the partition (one of the $K$ points) that has the smallest distance.

Another common practice when using the EM framework is to initialize using the $k$-means (Ueda & Nakano 1998; Roberts *et al.* 2000), but because in this work, the performance of the Bayesian network classifiers will be compared with the $k$-means, only the random start was considered.

## 5. Results and discussion

The results using the benchmark datasets are shown in table 2 and figure 5. In general, the three Bayesian network classifier models outperform the traditional clustering algorithms (crosses above the diagonal line in figure 5), except for the wine dataset, where the EM algorithm performs better than the three models, and the $k$-means is better than the CEM-CL trees and similar to CEM-TAN. This particular case could be due to the simple discretization method used for the continuous attributes in the benchmark datasets, and it is possible that more bins are needed for the wine dataset, whereas the $k$-means and EM use the original continuous attributes. Finding the optimum number of bins to use in the unsupervised discretization method for each dataset was not considered because, in general, for the 10 datasets tested, a discretization into five bins obtained good results.

For the flare, iris and vote datasets, the unsupervised training of the three Bayesian networks classifiers obtained similar results to the three models trained in a supervised way (last three columns of table 2). Datasets crx, diabetes, lymphography, wine and zoo have less than 10 per cent difference with the supervised results, whereas corral and glass perform significantly worse than the supervised results, but much better than the traditional clustering algorithms, especially $k$-means. The EM algorithm for datasets diabetes, flare, iris, vote and wine converged always to the same suboptimal solution in the 10 repetitions, and although the results obtained by the Bayesian network models are suboptimal as well, convergence to the same local optima is avoided due to the S-step. Another way to avoid this problem is to incorporate a simulated annealing approach described in Celeux & Govaert (1992) and Ueda & Nakano (1998). The effects of the SBN classifier, which automatically regulates the number of edges (complexity of the network), can be appreciated by analysing the first dataset in table 2. The corral dataset contains six attributes and the class variable is a Boolean function of only four of the attributes: $(1 \wedge 2) \vee (3 \wedge 4)$. The fifth attribute is entirely irrelevant, and the sixth attribute is 'correlated' with the class variable in that it matches the class label 75 per cent of the time. The structures for the best results obtained out of the five repetitions for the unsupervised training of the CL trees, TAN and SBN are shown in figures 6–8, respectively. Note that the edges from $C$ to the attributes are dotted because, in clustering, the $C$ variable is latent. From figure 8, it is clear that attribute 5 does not depend probabilistically on the other attributes, given $C$, and the jpd using equation (2.1) is

$$P(1,2,3,4,5,6,C) = P(C)P(1|C)P(5|C)P(2|1,C)P(3|1,C)P(4|3,C)P(6|3,C)$$

once the model is trained and the $C$ variable contains cluster labels. Because attribute 5 is irrelevant, in fact, $P(5|C=0)=0.5$ and $P(5|C=1)=0.5$ for any value that attribute 5 takes, $P(5|C)$ has no effect on the jpd computation, which is what it is expected. On the other hand, the TAN model is restricted to building tree structures, and in this case, attribute 5 participates in the jpd by $P(5|1,C)$. This conditional probability also remains constant and equal to 0.5, regardless of the values that 5, 1 and $C$ take, but requires unnecessary extra computation compared with $P(5|C)$. Also, from figure 7, it is not clear that attribute 5 is 'different' from the rest of the attributes. It has the same conditioning (parents)

Table 2. Experimental results using the benchmark datasets.

| dataset | CEM-CL trees | CEM-TAN | CEM-SBN | *k*-means | EM | CL trees | TAN | SBN |
|---|---|---|---|---|---|---|---|---|
| corral | 83.12 ± 2.79 | 77.50 ± 3.42 | 80.00 ± 2.79 | 67.81 ± 9.43 | 74.21 ± 1.98 | 99.23 ± 0.77 | 95.32 ± 2.26 | 96.80 ± 3.35 |
| crx | 79.57 ± 2.32 | 78.19 ± 3.53 | 80.82 ± 2.13 | 77.04 ± 3.78 | 74.42 ± 11.09 | 83.92 ± 1.05 | 83.77 ± 1.34 | 87.54 ± 1.74 |
| diabetes | 70.44 ± 3.38 | 70.80 ± 2.97 | 71.38 ± 2.13 | 66.79 ± 0.00 | 66.01 ± 0.00 | 74.35 ± 1.43 | 75.13 ± 0.98 | 78.30 ± 3.18 |
| flare | 81.57 ± 1.56 | 80.35 ± 0.75 | 81.03 ± 1.04 | 64.42 ± 8.98 | 79.08 ± 0.00 | 81.90 ± 1.51 | 82.74 ± 1.60 | 83.38 ± 1.27 |
| glass | 48.50 ± 2.17 | 52.33 ± 3.32 | 51.68 ± 3.23 | 43.73 ± 3.52 | 44.71 ± 1.37 | 69.17 ± 1.29 | 69.18 ± 2.64 | 70.00 ± 2.71 |
| iris | 90.66 ± 3.68 | 92.00 ± 2.30 | 92.40 ± 2.43 | 88.26 ± 0.34 | 90.66 ± 0.00 | 93.33 ± 1.05 | 93.33 ± 1.05 | 96.00 ± 3.65 |
| lymphography | 62.97 ± 4.83 | 62.83 ± 5.12 | 62.16 ± 5.50 | 44.32 ± 8.19 | 48.98 ± 5.71 | 64.11 ± 4.77 | 66.87 ± 3.37 | 68.28 ± 4.50 |
| vote | 89.28 ± 1.56 | 88.18 ± 1.57 | 88.00 ± 0.86 | 86.71 ± 0.14 | 87.81 ± 0.00 | 89.42 ± 1.72 | 89.20 ± 1.61 | 90.34 ± 2.09 |
| wine | 93.37 ± 3.58 | 94.83 ± 2.07 | 95.05 ± 1.08 | 94.66 ± 0.29 | 97.19 ± 0.00 | 98.27 ± 1.65 | 98.03 ± 1.55 | 97.14 ± 2.02 |
| zoo | 86.93 ± 1.90 | 87.72 ± 2.58 | 88.71 ± 2.27 | 83.16 ± 2.64 | 83.26 ± 3.66 | 93.09 ± 5.03 | 95.08 ± 4.25 | 94.00 ± 4.18 |

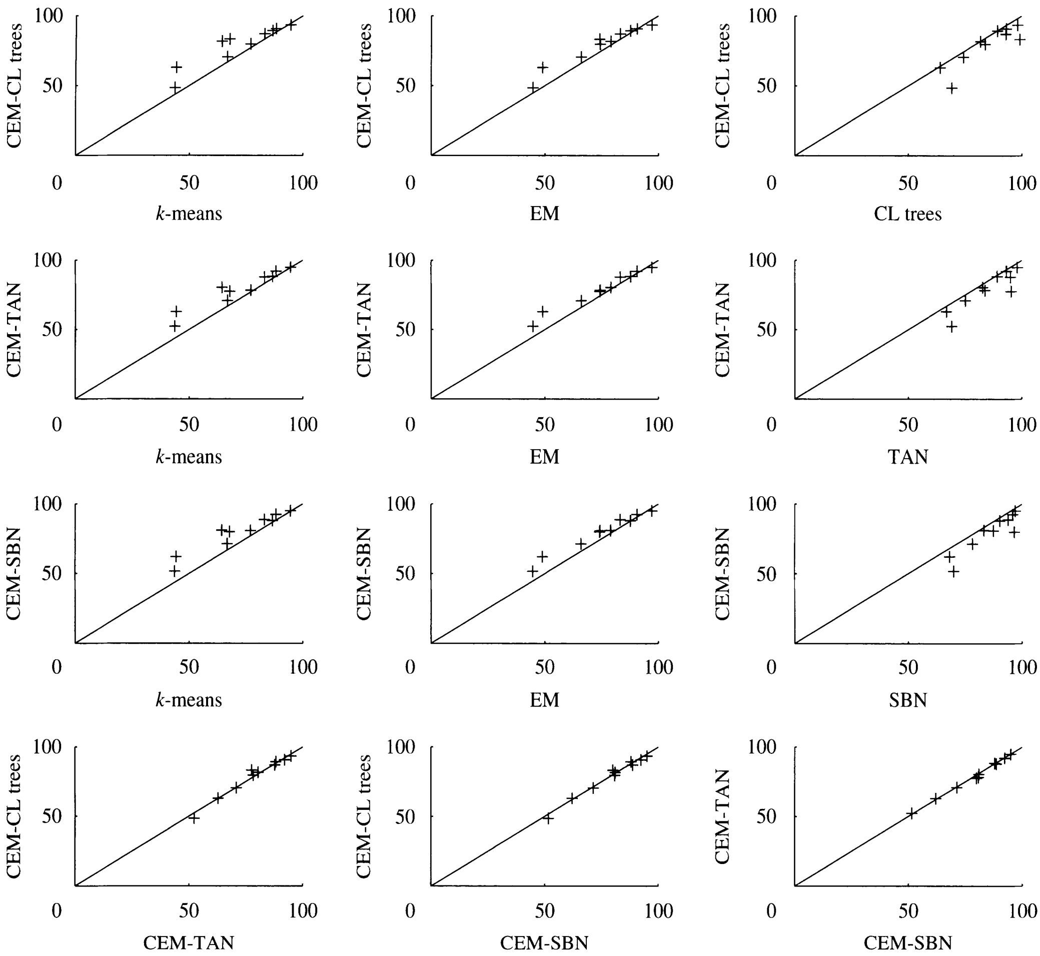

Figure 5. Scatter plots comparing the performance of: first row, CEM-CL trees ($y$-axis) versus $k$-means ($x$-axis); CEM-CL trees ($y$-axis) versus EM ($x$-axis); and CEM-CL trees ($y$-axis) versus CL trees ($x$-axis). Second row, CEM-TAN ($y$-axis) versus $k$-means ($x$-axis), CEM-TAN ($y$-axis) versus EM ($x$-axis) and CEM-TAN ($y$-axis) versus TAN ($x$-axis). Third row, CEM-SBN ($y$-axis) versus $k$-means ($x$-axis), CEM-SBN ($y$-axis) versus EM ($x$-axis) and CEM-SBN ($y$-axis) versus SBN ($x$-axis). Fourth row: CEM-CL trees ($y$-axis) versus CEM-TAN ($x$-axis), CEM-CL trees ($y$-axis) versus CEM-SBN ($x$-axis) and CEM-TAN ($y$-axis) versus CEM-SBN ($x$-axis). In these plots, each cross represents a dataset, and the coordinates correspond to the correct clustering percentage of each of the methods compared. Crosses above the diagonal line correspond to datasets where the $y$-axis method is more accurate, and crosses below the diagonal line correspond to datasets where the $x$-axis method is more accurate.

as attributes 3, 2 and 6, whereas in figure 8, it is clear that attribute 5 is different from the other attributes. Also, from table 2, the performance of the SBN, both in unsupervised and supervised learning, in most of the cases is superior to the TAN model, due to its more realistic representation of the data, which is not restricted to a tree structure all the time.

The performance of the CL multinets on the wood dataset is given in table 3, and although the computational cost is high for the 165-dimensional input space considering that each tree learning procedure in the multinet has

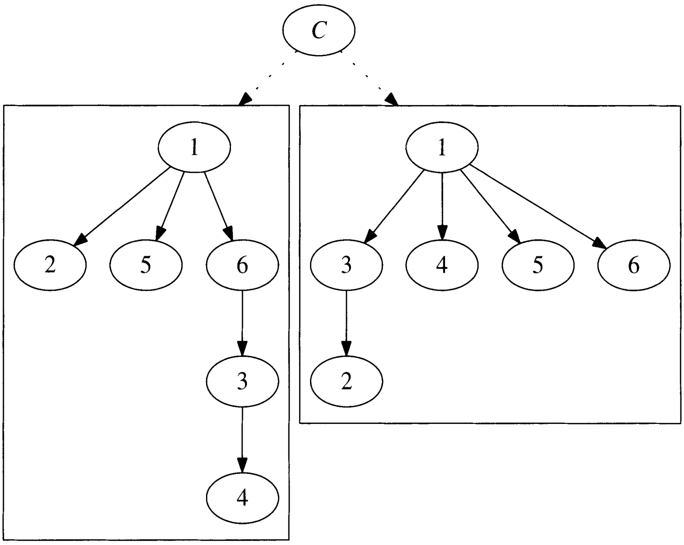

Figure 6. CL trees of the corral dataset.

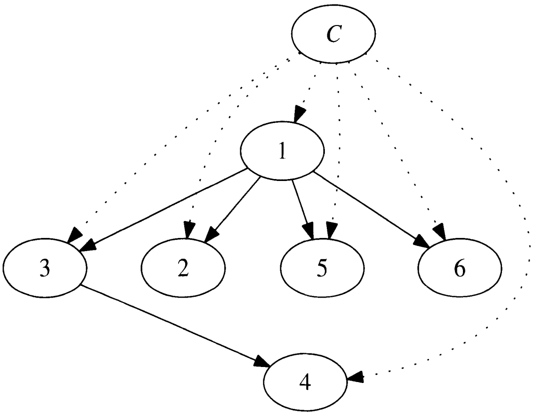

Figure 7. TAN of the corral dataset.

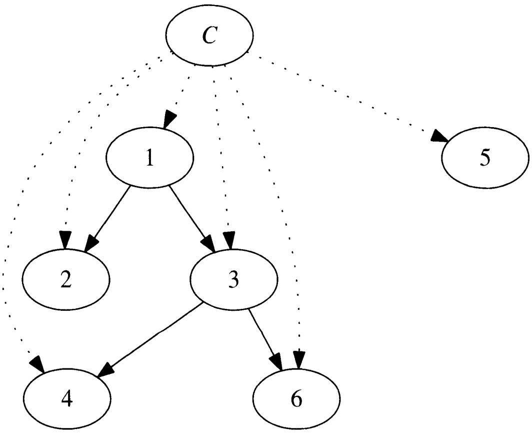

Figure 8. SBN of the corral dataset.

Table 3. Experimental results using the wood dataset.

| dataset | CEM-CL trees | $k$-means | EM |
|---|---|---|---|
| wood data | $65.60 \pm 1.30$ | $52.78 \pm 3.57$ | $57.74 \pm 1.81$ |

time complexity $O(n^2 n_k)$, the results obtained are considerably better than the traditional algorithms, especially when compared with the $k$-means. In these cases where there are a large number of attributes, the CL multinets, as well as the other two Bayesian network models, can be initialized using $k$-means in order to reduce the learning time. To prove this, a simple experiment was carried out using the best solution of the $k$-means (55.09%) as the starting partition $\mathrm{P}^0$, and the CL multinet reached a performance of 65.42 per cent in only 15 iterations (10 in the S-step and 5 in the C-step).

Although a clustering accuracy of 65 per cent might seem low, the best classification accuracy obtained for the same dataset in another study (Ruz *et al.* 2009) using carefully selected and finely tuned multi-layer perceptron (MLP) neural networks was only 83 per cent . This difference of 18 per cent in clustering/classification accuracies is not so significant if one considers the painstaking manual experimentation with several MLP architectures before the best model could be found and the subsequent lengthy supervised training (using class labels) to reach the 83 per cent correct classification. Also, for this particular application, it was observed that some of the defect categories, especially the stain and blue stain defect categories, overlap with the clear wood (defect free). This is because clear wood contains dark grain lines that can be mistaken for defects.

One of the limitations of the proposed unsupervised training method for the three Bayesian network models is that the number of clusters must be known *a priori*. This drawback is common for several clustering algorithms ($k$-means, EM, etc.), but can usually be overcome by learning models with different numbers of clusters and then selecting the best model according to some cluster validity index (Bezdek & Pal 1998).

## 6. Conclusion

Clustering using Bayesian network classifiers has been addressed in this paper. To accomplish the unsupervised training of the Bayesian network classifiers, which includes structure and parameter learning, as well as the clustering of the data, the maximization of the CML was used. This maximization was carried out by formulating the three Bayesian network classifiers—CL multinets, TAN and the SBN classifier—into the CEM algorithm framework. Results using benchmark datasets, as well as real-world applications, show that Bayesian network classifiers are promising models for clustering tasks. The SBN classifier generally outperforms the TAN model when trained in a supervised or an unsupervised way. This is due to its more robust structure learning process, which does not limit its structure to a tree and regulates the number of edges according to the amount of training data available that determines the level of complexity in the network structure. An important advantage of Bayesian network classifiers trained in an unsupervised way is that the resulting structure can give additional information on how the features are related (probabilistic dependencies) in each cluster, information that the traditional methods mentioned previously do not have. The proposed clustering technique can help a human expert identify and explain what each cluster means (represents), given the network representation for each cluster.

For future work, first, two of the main limitations of the proposed method should be researched. These are the difficulty in determining the correct number of clusters during the learning process and the need for a more effective unsupervised technique for transforming continuous attributes into discrete ones. Second, comparisons with other techniques such as the SOM and ART neural networks would be of interest to assess the performance of the Bayesian networks for clustering. Third, test the proposed models in other real-world applications, for example, in the biological sciences for genetic regulatory network modelling using gene expression data.

The authors would like to thank the EU-funded I*PROMS Network of Excellence and the ORS Award for financially supporting this research.

## Appendix A. Derivation of the Bayesian measure

Given a dataset $\boldsymbol{D}$, a Bayesian criterion, usually called the *Bayes factor*, for model selection says that, for a simple model $B_{\mathrm{s}}$ to be replaced by a more complex one $B_{\mathrm{c}}$, the Bayes factor $\lambda^*$ needs to satisfy the following:

$$\lambda^* = \frac{P(\boldsymbol{D}|B_{\mathrm{s}})P(B_{\mathrm{s}})}{P(\boldsymbol{D}|B_{\mathrm{c}})P(B_{\mathrm{c}})} < 1, \qquad (\mathrm{A}\,1)$$

applying the base 2 logarithm,

$$\log_2 \lambda^* = \lambda = \log_2 P(\boldsymbol{D}|B_{\mathrm{s}}) - \log_2 P(\boldsymbol{D}|B_{\mathrm{c}}) + \log_2 P(B_{\mathrm{s}}) - \log_2 P(B_{\mathrm{c}}), \qquad (\mathrm{A}\,2)$$

with the condition being $\lambda < 0$ to replace $B_{\mathrm{s}}$ by $B_{\mathrm{c}}$.

The prior of the Bayesian network can be expressed using an encoding system of the network. For a Bayesian network with $n$ attribute nodes, let the representation of the network be constructed using $n+1$ symbols (one for each attribute node plus an additional symbol to represent the stop command) of length $m = \log_2(n+1)$ bits. Let a Bayesian network $B$ encoding be represented by a sequence of ordered pairs: $(n_1, n_2)(n_3, n_4)\ldots(n_{\mathrm{s}-1}, n_{\mathrm{s}})\mathit{Stop}$, where an ordered pair $(n_i, n_j)$ is a code $C_{ij} = C(n_i)C(n_j)$ formed by the concatenation of the code representing the node $n_i$ and the code representing $n_j$. Each ordered pair in the sequence represents the nodes that have an edge between them.

It is known, from the coding theory (MacKay 2003), that probabilities can be mapped to optimal codelengths, a uniquely decodeable code that minimizes the expected codelength. The expected length is minimized only if the codelengths $l(M_i)$ for a given model $M_i$ are equal to the *Shannon information* contents, $l(M_i) = \log_2(1/P(M_i))$. So, for a Bayesian network $B$ with description length $l(B)$, the prior can be estimated by

$$P(B) = 2^{-l(B)}, \tag{A 3}$$

and if the data $\boldsymbol{D} = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\}$ is independent and identically distributed, then equation (A 2) can be written as

$$\lambda = l(B_c) - l(B_s) + \sum_{r=1}^{N} \sum_{i=1}^{n} \log_2 P(x_i^r | B_s) - \log_2 P(x_i^r | B_c). \tag{A 4}$$

Then, let the structure of $B_c$ be the same as $B_s$, but with one additional edge. This means that the description length of $B_c$ uses two more symbols than $B_s$, i.e. $l(B_c) - l(B_s) = 2m$. Thus, if the extra edge in $B_c$ is due to attribute $X_\omega$ being the parent of attribute $X_\upsilon$, and if the decomposability of a Bayesian network given by equation (2.1) is considered, equation (A 4) can be expressed as

$$\lambda = 2m + \sum_{r=1}^{N} \log_2 P(x_\upsilon^r | c^r) - \log_2 P(x_\upsilon^r | x_\omega^r, c^r). \tag{A 5}$$

Equation (A 5) is the Bayesian measure that presents a way to measure the effect of adding an extra edge to a naive Bayesian network classifier. Negative values of $\lambda$ indicate that there are enough data to support that extra edge. The next step is to find a stopping condition for augmenting the naive Bayes classifier. For this, comparing the effect of having no edges between attributes (naive Bayes) against having $e$ augmenting edges is required. Because of the logarithmic property (multiplicative terms become additive ones), the value to compute is the cumulative Bayesian measure, $\Lambda_e$, which indicates if there is enough data to support $e$ edges compared with 0 edges (naive Bayes); this is given by

$$\Lambda_e = \sum_{i=1}^{e} \lambda_i, \tag{A 6}$$

where $\lambda_i$ represents the $\lambda$ value for the $i$th edge being considered. Then, the adding of edges will continue while $\Lambda_e < 0$. When $\Lambda_e$ becomes positive, the network does not have sufficient data to support the adding of any more edges.

## References

Acid, S., de Campos, L. M. & Castellano, J. G. 2005 Learning Bayesian network classifiers: search in a space of partially directed acyclic graphs. *Mach. Learn.* **59**, 213–235. (doi:10.1007/s10994-005-0473-4)

Asuncion, A. & Newman, D. J. 2007 *UCI machine learning repository.* Irvine, CA: University of California, School of Information and Computer Science. See http://www.ics.ci.edu/~mlearn/MLRepository.html.

Barash, Y. & Friedman, N. 2002 Context-specific Bayesian clustering for gene expression data. *J. Comput. Biol.* **9**, 169–191. (doi:10.1089/10665270252935403)

Bezdek, J. C. 1981 *Pattern recognition with fuzzy objective function algorithms.* New York, NY: Plenum Press.

Bezdek, J. C. & Pal, N. R. 1998 Some new indexes of cluster validity. *IEEE Trans. Syst. Man. Cybern. B* **28**, 301–315. (doi: 10.1109/3477.678624)

Biba, M., Esposito, F., Ferilli, S., Di Mauro, N., Basile, T. M. A. 2007 Unsupervised discretization using kernel density estimation. In *Proc. Int. Joint Conf. on Artificial Intelligence* (*IJCAI 2007*), *Hyderabad, India.*

Carpenter, G. & Grossberg, S. 1987 A massively parallel architecture for a self-organizing neural pattern recognition machine. *Comput. Vision Graph. Image Understanding* **37**, 54–115. (doi:10.1016/S0734-189X(87)80014-2)

Celeux, G. & Govaert, G. 1992 A classification EM algorithm for clustering and two stochastic versions. *Comput. Stat. Data Anal.* **14**, 315–332. (doi:10.1016/0167-9473(92)90042-E)

Celeux, G. & Govaert, G. 1995 Gaussian parsimonious clustering models. *Pattern Recognit.* **28**, 781–793. (doi:10.1016/0031-3203(94)00125-6)

Cerquides, J. & Lopez de Mantaras, R. 2005 TAN classifiers based on decomposable distributions. *Mach. Learn.* **59**, 323–354. (doi:10.1007/s10994-005-0470-7)

Chow, C. K. & Liu, C. N. 1968 Approximating discrete probability distributions with dependence trees. *IEEE Trans. Inform. Theory* **IT-14**, 462–467. (doi:10.1109/TIT.1968.1054142)

Dash, D. & Cooper, G. F. 2002 Exact model averaging with naive Bayesian classifiers. In *Proc. 19th Int. Conf. Machine Learning.* San Francisco, CA: Morgan Kauffman.

Dempster, A. N., Laird, N. M. & Rubin, D. B. 1977 Maximum likelihood from incomplete data via the EM algorithm. *J. R. Statist. Soc. Ser.* **39**, 1–38.

Dougherty, J., Kohavi, R. & Sahami, M. 1995 Supervised and unsupervised discretization of continuous features. In *Proc. 12th Int. Conf. Machine Learning.* San Francisco, CA: Morgan Kaufmann.

Duda, R. O. & Hart, P. E. 1973 *Pattern classification and scene analysis.* New York, NY: John Wiley & Sons.

Estévez, P. A., Perez, C. A. & Goles, E. 2003 Genetic input selection to a neural classifier for defect classification of radiata pine boards. *Forest Prod. J.* **53**, 87–94.

Fayyad, U. M. & Irani, K. B. 1993 Multi-interval discretization of continuous-valued attributes for classification learning. In *Proc. 13th Int. Joint Conf. on Artificial Intelligence.* San Francisco, CA: Morgan Kaufmann.

Friedman, N. 1998 The Bayesian structural EM algorithm. In *14th Proc. Conf. on Uncertainty in Artificial Intelligence.* San Francisco, CA: Morgan Kaufmann.

Friedman, N., Geiger, D. & Goldszmidt, M. 1997 Bayesian network classifiers. *Mach. Learn.* **29**, 131–163. (doi:10.1023/A:1007465528199)

Grossman, D. & Domingos, P. 2004 Learning Bayesian network classifiers by maximizing conditional likelihood. In *Proc. 21st Int. Conf. on Machine Learning.* New York, NY: ACM.

Gurwicz, Y. & Lerner, B. 2006 Bayesian class-matched multinet classifier. *Lect. Notes Comput. Sci.* **4109**, 145–153. (doi:10.1007/11815921)

Hwang, K. & Zhang, B. 2005 Bayesian model averaging of Bayesian network classifiers over multiple node-orders: application to sparse datasets. *IEEE Trans. Syst. Man. Cybern. B, Cybern.* **35**, 1302–1310. (doi:10.1109/TSMCB.2005.850162)

Jing, Y., Pavlović, V. & Rehg J. M. 2008 Boosted Bayesian network classifiers. *Mach. Learn.* **73**, 155–184. (doi:10.1007/s10994-008-5065-7)

Kerr, G., Ruskin, H. J., Crane, M. & Doolan, P. 2008 Techniques for clustering gene expression data. *Comput. Biol. Med.* **38**, 283–293. (doi:10.1016/j.compbiomed.2007.11.001)

Kohonen, T. 1995 *Self-organizing maps.* Berlin, Germany: Springer-Verlag.

Kruskal, J. B. 1956 On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Am. Math. Soc.* **7**, 48–50. (doi:10.2307/2033241)

Langley, P., Iba, W. & Thompson, K. 1992 An analysis of Bayesian classifiers. In *Proc. 10th National Conf. on Artificial Intelligence.* Menlo Park, CA: AAAI Press.

Li, J., Zhang, C., Wang, T. & Zhang Y. 2007 Generalized additive Bayesian network classifiers. In *Proc. Int. Joint Conf. on Artificial Intelligence* (*IJCAI 2007*), *Hyderabad, India.*

MacKay, D. J. C. 2003 *Information theory, inference, and learning algorithms.* Cambridge, UK: Cambridge University Press.

MacQueen, J. 1967 Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symp. on Mathematical Statistics and Probability.* Berkeley: University of California Press.

Meilă, M. & Jordan, M. I. 2000 Learning with mixtures of trees. *J. Mach. Learn. Res.* **1**, 1–48. (doi:10.1162/153244301753344605)

Neapolitan, R. E. 2004 *Learning Bayesian networks.* Upper Saddle River, NJ: Pearson Prentice Hall.

Pearl, J. 1988 *Probabilistic reasoning in intelligent systems: networks of plausible inference.* San Francisco, CA: Morgan Kaufmann.

Peña, J. M., Lozano, J. M. & Larrañaga, P. 1999 Learning Bayesian networks for clustering by means of constructive induction. *Pattern Recogn. Lett.* **20**, 1219–1230. (doi:10.1016/S0167-8655(99)00089-6)

Peña, J. M., Lozano, J. M. & Larrañaga, P. 2000 An improved Bayesian structural EM algorithm for learning Bayesian networks for clustering. *Pattern Recogn. Lett.* **21**, 779–786. (doi:10.1016/S0167-8655(00)00038-6)

Peña, J. M., Lozano, J. M. & Larrañaga, P. 2004 Unsupervised learning of Bayesian networks via estimation of distribution algorithms: an application to gene expression data clustering. *Int. J. Uncertain. Fuzziness Knowledge Based Syst.* **12**(Suppl. 1), 63–82.

Pham, D. T. & Afify, A. A. 2007 Clustering techniques and their applications in engineering. *Proc. IMechE C, J. Mech. Eng. Sci.* **221**, 1445–1459. (doi:10.1243/09544062JMES508)

Pham, D. T. & Alcock R. J. 2003 *Smart inspection systems.* London, UK: Academic Press.

Roberts, S. J., Everson, R. & Rezek I. 2000 Maximum certainty data partitioning. *Pattern Recogn.* **33**, 833–839. (doi:10.1016/S0031-3203(99)00086-2)

Ruz, G. A., Estévez, P. A. & Perez, C. A. 2005 A neurofuzzy color image segmentation method for wood surface defect detection. *Forest Prod. J.* **55**, 52–58.

Ruz, G. A., Estévez, P. A. & Ramírez, P. A. 2009 Automated visual inspection system for wood defect classification using computational intelligence techniques. *Int. J. Syst. Sci.* **40**, 163–172. (doi:10.1080/00207720802630685)

Santafé G., Lozano, J. A. & Larrañaga, P. 2006*a* Bayesian model averaging of naive Bayes for clustering. *IEEE Trans. Syst. Man. Cybern. B* **36**, 1149–1161. (doi:10.1109/TSMCB.2006.874132)

Santafé G., Lozano, J. A. & Larrañaga, P. 2006*b* Bayesian model averaging of TAN models for clustering. In *Proc. European Workshop on Probabilistic Graphical Models (PGM 2006), Prague, Czech Republic.*

Schmidberger, G. & Frank, E. 2005 Unsupervised discretization using tree-based density estimation. In *Proc. 9th European Conf. on Principles and Practice of Knowledge Discovery in Databases, Porto, Portugal.* Berlin, UK: Springer. (doi:10.1007/11564126_26)

Simpson P. K. 1993 Fuzzy min–max neural networks—part 2: clustering. *IEEE Trans. Fuzzy Syst.* **1**, 32–45. (doi:10.1109/TFUZZ.1993.390282)

Ueda, N. & Nakano, R. 1998 Deterministic annealing EM algorithm. *Neural Netw.* **11**, 271–282. (doi:10.1016/S0893-6080(97)00133-0)