Tractable Bayesian Learning of Tree Augmented Naive Bayes Models

Jesús Cerquides CERQUIDE@MAIA.UB.ES
Dept. de Matemàtica Aplicada i Anàlisi, Universitat de Barcelona
Gran Via 585, 08007 Barcelona, Spain

Ramon López de Màntaras MANTARAS@IIIA.CSIC.ES
Institut d'Investigació en Intel.ligència Artificial, Consejo Superior de Investigaciones Científicas
Campus UAB, 08193 Bellaterra, Spain

Abstract

Bayesian classifiers such as *Naive Bayes* or *Tree Augmented Naive Bayes* (TAN) have shown excellent performance given their simplicity and heavy underlying independence assumptions. In this paper we introduce a classifier taking as basis the TAN model and taking into account uncertainty in model selection. To do this we introduce decomposable distributions over TANs and show that they allow the expression resulting from the Bayesian model averaging of TAN models to be integrated into closed form. With this result we construct a classifier with a shorter learning time and a longer classification time than TAN. Empirical results show that the classifier is, most of the cases, more accurate than TAN and approximates better the class probabilities.

1. Introduction

Bayesian classifiers as *Naive Bayes* (Langley et al., 1992) or *Tree Augmented Naive Bayes* (TAN) (Friedman et al., 1997) have shown excellent performance in spite of their simplicity and heavy underlying independence assumptions.

Furthermore, it has been shown (Cerquides & López de Màntaras, 2003a; Kontkanen et al., 1998) that *Naive Bayes* predictions and probability estimations can benefit from incorporating uncertainty in model selection by means of Bayesian model averaging. In the case of TAN, a development inspired in this same idea is presented in (Cerquides, 1999), where to overcome the difficulty of exactly calculating the averaged classifier the idea of local Bayesian model averaging is introduced to calculate an approximation. In this case predictions are also improved.

In this paper we show that, under suitable assumptions, the Bayesian model averaging of TAN can be integrated in closed form and that it leads to improved classification performance. The paper is organized as follows. In section 2 we review *Tree Augmented Naive Bayes* and the notation that we will use in the rest of the paper. In section 3 we develop the closed expression for the Bayesian model averaging of TAN and we construct a classifier based on this result which we will name TBMATAN (from Tractable Bayesian Model Averaging of Tree Augmented Naive-Bayes). In section 4 we notice that TBMATAN has a major drawback that makes difficult its usage for large datasets because it depends on the calculation of an ill-conditioned determinant that requires the floating point precision to increase with the dataset size and hence increases the computing time. To solve this drawback we introduce SSTBMATAN, an approximation of TBMATAN. In section 5 we study the empirical characteristics of TBMATAN and show that it leads to improving classification accuracy and to a better approximation of the class probabilities with respect to TAN. We also show that the empirical results for SSTBMATAN do not differ significantly from the ones obtained by TBMATAN while allowing to deal with large datasets. We end up with some conclusions and future work in section 6.

2. Tree Augmented Naive Bayes

Tree Augmented Naive Bayes (TAN) appears as a natural extension to the *Naive Bayes* classifier (Kontkanen et al., 1998; Langley et al., 1992; Domingos & Pazzani, 1997). TAN models are a restricted family of Bayesian networks in which the class variable has no parents and each attribute has as parents the class variable and at most another attribute. An example of TAN model can be seen in Figure 1(c).

In this section we start introducing the notation to be used in the rest of the paper. After that we discuss

Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC, 2003.

the TAN induction algorithm presented in (Friedman et al., 1997). Finally in this section we present also the improvements introduced to TAN in (Cerquides, 1999).

2.1. Formalization and Notation

The notation used in the paper is an effort to put together the different notations used in (Cerquides, 1999; Heckerman et al., 1995; Friedman et al., 1997; Meila & Jaakkola, 2000) and some conventions in the machine learning literature.

2.1.1. The Discrete Classification Problem

A *discrete attribute* is a finite set. A *discrete domain* is a finite set of discrete attributes. We will note $\Omega = \{X_1, \ldots, X_m\}$ for a discrete domain, where $X_1, \ldots, X_m$ are the attributes in the domain. A *classified discrete domain* is a discrete domain where one of the attributes is distinguished as "class". We will use $\Omega_C = \{A_1, \ldots, A_n, C\}$ for a classified discrete domain. In the rest of the paper we will refer to an attribute either as X_i (when it is considered part of a discrete domain), A_i (when it is considered part of a classified discrete domain and it is not the class) and C (when it is the class of a classified discrete domain). We will note as $V = \{A_1, \ldots, A_n\}$ the set of attributes in a classified discrete domain that are not the class.

Given an attribute A, we will note $\#A$ as the number of different values of A. We define $\#\Omega = \prod_{i=1}^{m} \#X_i$ and $\#\Omega_C = \#C \prod_{i=1}^{n} \#A_i$.

An *observation* x in a classified discrete domain Ω_C is an ordered tuple $x = (x_1, \ldots, x_n, x_C) \in A_1 \times \ldots \times A_n \times C$. An *unclassified observation* S in Ω_C is an ordered tuple $S = (s_1, \ldots, s_n) \in A_1 \times \ldots \times A_n$. To be homogeneous we will abuse this notation a bit noting s_C for a possible value of the class for S. A *dataset* $\mathcal{D}$ in Ω_C is a multiset of classified observations in Ω_C.

We will note N for the number of observations in the dataset. We will also note $N_i(x_i)$ for the number of observations in $\mathcal{D}$ where the value for A_i is x_i, $N_{i,j}(x_i, x_j)$ the number of observations in $\mathcal{D}$ where the value for A_i is x_i and the value for A_j is x_j and similarly for $N_{i,j,k}(x_i, x_j, x_k)$ and so on. We note similarly $f_i(x_i), f_{i,j}(x_i, x_j), \ldots$ the frequencies in $\mathcal{D}$. It is worth noticing that f defines a probability distribution over $A_1 \times \ldots \times A_n \times C$.

A *classifier* in a classified discrete domain Ω_C is a procedure that given a dataset $\mathcal{D}$ in Ω_C and an unclassified observation S in Ω_C assigns a class to S.

2.1.2. Bayesian Networks for Discrete Classification

Bayesian networks offer a solution for the discrete classification problem. The approach is to define a random variable for each attribute in Ω (the class is included but not distinguished at this time). We will note $\mathbf{U} = \{\mathcal{X}_1, \ldots, \mathcal{X}_m\}$ where each $\mathcal{X}_i$ is a random variable over its corresponding attribute X_i. We extend the meaning of this notation to $\mathcal{A}_i$, $\mathcal{C}$ and $\mathcal{V}$. A *Bayesian network* over $\mathbf{U}$ is a pair $B = \langle G, \Theta \rangle$. The first component, G, is a directed acyclic graph whose vertices correspond to the random variables $\mathcal{X}_1, \ldots, \mathcal{X}_m$ and whose edges represent direct dependencies between the variables. The graph G encodes independence assumptions: each variable $\mathcal{X}_i$ is independent of its non-descendants given its parents in G. The second component of the pair, namely Θ, represents the set of parameters that quantifies the network. It contains a parameter $\theta_{i|\Pi_i}(x_i, \Pi_{x_i}) = P_B(x_i|\Pi_{x_i})$ for each $x_i \in X_i$ and $\Pi_{x_i} \in \Pi_{X_i}$, where Π_{X_i} denotes the Cartesian product of every X_j such that $\mathcal{X}_j$ is a parent of $\mathcal{X}_i$ in G. Π_i is the list of parents of $\mathcal{X}_i$ in G. We will note $\overline{\Pi_i} = \mathbf{U} - \{\mathcal{X}_i\} - \Pi_i$. A Bayesian network defines a unique joint probability distribution over $\mathbf{U}$ given by

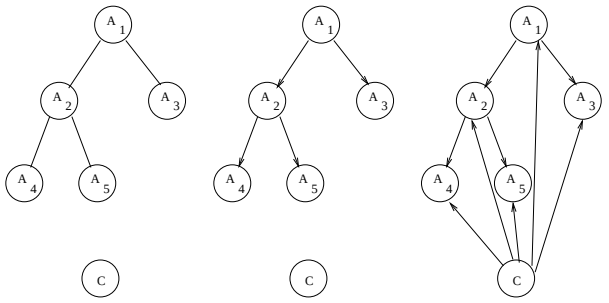
$$P_B(x_1, \ldots, x_m) = \prod_{i=1}^{m} P_B(x_i|\Pi_{x_i}) = \prod_{i=1}^{m} \theta_{i|\Pi_i}(x_i|\Pi_{x_i}) \tag{1}$$

The application of Bayesian networks for classification can be very simple. For example suppose we have an algorithm that given a classified discrete domain Ω_C and a dataset $\mathcal{D}$ over Ω_C returns a Bayesian network B over $\mathbf{U} = \{\mathcal{A}_1, \ldots, \mathcal{A}_n, \mathcal{C}\}$ where each $\mathcal{A}_i$ (resp. $\mathcal{C}$) is a random variable over A_i (resp. C). Then if we are given a new unclassified observation S we can easily classify S into class $\underset{s_C \in C}{argmax}(P_B(s_1, \ldots, s_n, s_C))$. This simple mechanism allows us to see any Bayesian network learning algorithm as a classifier.

2.1.3. Learning with Trees

Given a classified domain Ω_C we will note $\mathcal{E}$ the set of undirected graphs E over $\{\mathcal{A}_1, \ldots, \mathcal{A}_n\}$ such that E is a tree (has no cycles). We will use $u, v \in E$ instead of $(\mathcal{A}_u, \mathcal{A}_v) \in E$ for compactness. We will note as $\overline{E}$ a directed tree for E. Every $\overline{E}$ uniquely determines the structure of a Tree Augmented Naive Bayes classifier, because from $\overline{E}$ we can construct $\overline{E}^* = \overline{E} \cup \{(\mathcal{C}, \mathcal{A}_i) | 1 \leq i \leq n\}$ as can be seen in an example in Figure 1. We note the root of a directed tree $\overline{E}$ as $\rho_{\overline{E}}$ (i.e. in Figure 1(b) we have that $\rho_{\overline{E}} = A_1$).

We will note as $\Theta_{\overline{E}^*}$ the set of parameters that quantify the Bayesian network $M = \langle \overline{E}^*, \Theta_{\overline{E}^*} \rangle$. More con-



(a) E (b) $\overline{E}$ (c) $\overline{E}^*$

Figure 1. Notation for learning with trees

cretely:
$\Theta_{\overline{E}^*} = (\boldsymbol{\theta}_C, \boldsymbol{\theta}_{\rho_{\overline{E}}|C}, \{\boldsymbol{\theta}_{v|u,C} | u, v \in \overline{E}\})$
$\boldsymbol{\theta}_C = \{\theta_C(c) | c \in C\}$ where $\theta_C(c) = P(\mathcal{C} = c|M)$
$\boldsymbol{\theta}_{\rho_{\overline{E}}|C} = \{\theta_{\rho_{\overline{E}}|C}(i,c) | i \in A_{\rho_{\overline{E}}}, c \in C\}$ where
$\theta_{\rho_{\overline{E}}|C}(i,c) = P(\mathcal{A}_{\rho_{\overline{E}}} = i | \mathcal{C} = c, M)$
For each $u, v \in \overline{E}$:
$\boldsymbol{\theta}_{v|u,C} = \{\theta_{v|u,C}(j,i,c) | j \in A_v, i \in A_u, c \in C\}$
where $\theta_{v|u,C}(j,i,c) = P(\mathcal{A}_v = j | \mathcal{A}_u = i, \mathcal{C} = c, M)$.

2.2. Learning Maximum Likelihood TAN

One of the measures used to learn Bayesian networks is the *log likelihood*. An interesting property of the TAN family is that we have an efficient procedure (Friedman et al., 1997) for identifying the structure of the network which maximizes likelihood. To learn the maximum likelihood TAN we should use the following equation to compute the parameters.

$$\theta_{i|\Pi_i}(x_i, \Pi_{x_i}) = \frac{N_{i,\Pi_i}(x_i, \Pi_{x_i})}{N_{\Pi_i}(\Pi_{x_i})} \tag{2}$$

It has been shown (Friedman et al., 1997) that equation 2 leads to "overfitting" the model. Also in (Friedman et al., 1997) Friedman et al. propose to use the parameters as given by

$$\theta_{i|\Pi_i}(x_i, \Pi_{x_i}) = \frac{N_{i,\Pi_i}(x_i, \Pi_{x_i})}{N_{\Pi_i}(\Pi_{x_i}) + N^0_{i|\Pi_i}} + \\ + \frac{N^0_{i|\Pi_i}}{N_{\Pi_i}(\Pi_{x_i}) + N^0_{i|\Pi_i}} \frac{N_i(x_i)}{N} \tag{3}$$

and suggest setting $N^0_{i|\Pi_i} = 5$ based on empirical results. Using equation 3 to fix the parameters improves the accuracy of the classifier. In our opinion, no well founded theoretical justification is given for the improvement. In the following section we revisit the results in (Cerquides, 1999) and show that we can get an alternative parameter fixing equation with a well founded theoretical justification and equivalent classification accuracy.

2.3. Learning Multinomial Sampling TAN

In (Cerquides, 1999) we introduced an alternative approach to learning Bayesian networks which we named "multinomial sampling approach" based on assuming that our dataset is a sample of a multinomial distribution over $A_1 \times \ldots \times A_n \times C$.

This multinomial sampling approach was applied to TAN with the result that we should estimate the parameters using:

$$\theta_{i|\Pi_i}(x_i, \Pi_{x_i}) = \frac{N_{i,\Pi_i}(x_i, \Pi_{x_i}) + \lambda \frac{\#\overline{\Pi_i}}{\#\Omega}}{N_{\Pi_i}(\Pi_{x_i}) + \lambda \frac{\#X_i \#\overline{\Pi_i}}{\#\Omega}} \tag{4}$$

where $\#\overline{\Pi_i} = \prod_{\mathcal{X}_j \in \overline{\Pi_i}} \#X_j$ and we remind that $\overline{\Pi_i}$ stands for the set of variables which are not parents of $\mathcal{X}_i$ in the network excluding $\mathcal{X}_i$.

In (Cerquides, 1999) the usage of $\lambda = 10$ was found to be a good value after empirical tests, and the multinomial sampling approach was compared to the maximum likelihood (equation 2) and softened maximum likelihood (equation 3) parameter estimations. The results were that multinomial sampling is clearly better than maximum likelihood. When compared to softened maximum likelihood, it was observed that multinomial sampling provides an equivalent classification accuracy but improves the quality of the probabilities assigned to the class.

3. Development of the Averaged Tree Augmented Naive Bayes

In the previous section we have reviewed different ways of learning a single TAN model from data. In this section we will develop a classifier based on the TAN model that does also take into account the uncertainty in model selection by means of decomposable distributions over TANs. We start by introducing Bayesian model averaging, then we explain decomposable distributions over tree structures and parameters built upon the idea of decomposable priors as proposed by Meila and Jaakola (Meila & Jaakkola, 2000) to end up showing that given a decomposable distribution it is possible to calculate the probability of an unseen observation and that given a prior decomposable distribution, the posterior distribution after observing a set of data is also a decomposable distribution. We conclude the section by putting together these results to create TBMATAN.

3.1. BMA Classification

We are faced with the problem of defining a good classifier for a classified dataset. If we accept that there is a probabilistic model behind the dataset, we have two alternatives:

1. We know the model M (both structure and parameters) that is generating the data in advance. In this case it is a matter of probabilistic computation. We should be able to calculate $P(\mathcal{C} = s_C|\mathcal{V} = S, M)$ and to choose the class s_C with the highest probability. No learning is performed, because we knew the model in advance.

2. We are given a set of possible models $\mathcal{M}$. In this situation probability theory tell us we should take a weighted average where each model prediction is weighted by the probability of the model given the data. More formally, assuming ξ represents the hypothesis that the model underlying the data is known to be in $\mathcal{M}$ we have that:

$$P(\mathcal{V} = S, \mathcal{C} = s_C|\mathcal{D}, \xi) = \\ = \int_{M \in \mathcal{M}} P(\mathcal{V} = S, \mathcal{C} = s_C|M) P(M|\mathcal{D}, \xi) \quad (5)$$

   Applying this equation is commonly known as Bayesian model averaging (Hoeting et al., 1998).

In the following we prove that if we fix the set of models $\mathcal{M}$ to TAN models and assume a decomposable distribution as prior probability distribution over the set of models, the integral for $P(\mathcal{V} = S, \mathcal{C} = s_C|\mathcal{D}, \xi)$ in equation 5 can be integrated in closed form.

3.2. Decomposable Distributions over TANs

In order to apply Bayesian model averaging, it is necessary to have a prior probability distribution over the set of models $\mathcal{M}$. Decomposable priors were introduced by Meila and Jaakola in (Meila & Jaakkola, 2000) where it was demonstrated for tree belief networks that if we assume a decomposable prior, the posterior probability is also decomposable and can be completely determined analytically in polynomial time.

In this section we introduce decomposable distributions over TANs, which are probability distributions in the space $\mathcal{M}$ of TAN models and an adaptation of decomposable priors, as they appear in (Meila & Jaakkola, 2000), to the task of learning TAN.

Decomposable distributions are constructed in two steps. In the first step, a distribution over the set of different undirected tree structures is defined. Every directed tree structure is defined to have the same probability as its undirected equivalent. In the second step, a distribution over the set of parameters is defined so that it is also independent on the structure. In the rest of the paper we will assume ξ implies a decomposable distribution over $\mathcal{M}$ with hyperparameters $\boldsymbol{\beta}, \mathbf{N}'$ (these hyperparameters will be explained along the development). Under this assumption, the probability for a model $M = \langle \overline{E}^*, \Theta_{\overline{E}^*} \rangle$ (a TAN with fixed tree structure $\overline{E}^*$ and fixed parameters $\Theta_{\overline{E}^*}$) is determined by:

$$P(M|\xi) = P(\overline{E}^*, \Theta_{\overline{E}^*}|\xi) = P(\overline{E}^*|\xi) P(\Theta_{\overline{E}^*}|\overline{E}^*, \xi) \quad (6)$$

In the following sections we specify the value of $P(\overline{E}^*|\xi)$ (decomposable distribution over structures) and $P(\Theta_{\overline{E}^*}|\overline{E}^*, \xi)$ (decomposable distribution over parameters).

3.2.1. Decomposable Distribution over TAN Structures

One of the hyperparameters of a decomposable distribution is an $n \times n$ matrix $\boldsymbol{\beta} = (\beta_{u,v})$ such that $\forall u, v : 1 \leq u, v \leq n \ : \beta_{u,v} = \beta_{v,u} \geq 0 \ ; \ \beta_{v,v} = 0$. We can interpret $\beta_{u,v}$ as a measure of how possible is under ξ that the edge $(\mathcal{A}_u, \mathcal{A}_v)$ is contained in the TAN model underlying the data.

Given ξ, the probability of a TAN structure $\overline{E}^*$ is defined as:

$$P(\overline{E}^*|\xi) = \frac{1}{Z_\beta} \prod_{u,v \in E} \beta_{u,v} \quad (7)$$

where Z_β is a normalization constant with value:

$$Z_\beta = \sum_{E \in \mathcal{E}} \prod_{u,v \in E} \beta_{u,v} \quad (8)$$

It is worth noting that $P(\overline{E}^*|\xi)$ depends only on the underlying undirected tree structure E.

3.2.2. Decomposable Distribution over TAN Parameters

Applying equation 1 to the case of TAN we have that

$$P(\Theta_{\overline{E}^*}|\overline{E}^*, \xi) = P(\boldsymbol{\theta}_C|\overline{E}^*, \xi) P(\boldsymbol{\theta}_{\rho_{\overline{E}}|C}|\overline{E}^*, \xi) \times \\ \times \prod_{u,v \in \overline{E}} P(\boldsymbol{\theta}_{v|u,C}|\overline{E}^*, \xi) \quad (9)$$

A decomposable distribution has a hyperparameter set $\mathbf{N}' = \{N'_{v,u,C}(j, i, c)|1 \leq u \neq v \leq n \ ; \ j \in A_v \ ; \ i \in$

$A_u\ ;\ c \in C\}$ with the constraint that exist $N'_{u,C}(i,c)$, $N'_C(c)$, N' such that for every u,v:

$$N'_{u,C}(i,c) = \sum_{j \in A_v} N'_{v,u,C}(j,i,c) \tag{10}$$

$$N'_C(c) = \sum_{i \in A_u} N'_{u,C}(i,c) \tag{11}$$

$$N' = \sum_{c \in C} N'_C(c) \tag{12}$$

Given ξ, a decomposable probability distribution over parameters with hyperparameter $\mathbf{N}'$ is defined by equation 9 and the following set of Dirichlet distributions:

$$P(\boldsymbol{\theta}_C|\overline{E},\xi) = D(\theta_C(.); N'_C(.)) \tag{13}$$

$$P(\boldsymbol{\theta}_{\rho_{\overline{E}}|C}|\overline{E},\xi) = \prod_{c \in C} D(\theta_{\rho_{\overline{E}}|C}(.,c); N'_{\rho_{\overline{E}},C}(.,c)) \tag{14}$$

$$P(\boldsymbol{\theta}_{v|u,C}|\overline{E},\xi) = \prod_{c \in C} \prod_{i \in A_u} D(\theta_{v|u,C}(.,i,c); N'_{v,u,C}(.,i,c)) \tag{15}$$

If the conditions in equations 6, 7, 8, 9, 10, 11, 12, 13, 14 and 15 hold, we will say that $P(M|\xi)$ follows a decomposable distribution with hyperparameters $\boldsymbol{\beta}, \mathbf{N}'$.

3.3. Calculating Probabilities with Decomposable Distributions

Assume that the data is generated by a TAN model and that $P(M|\xi)$ follows a decomposable distribution with hyperparameters $\boldsymbol{\beta}, \mathbf{N}'$. We can calculate the probability of an observation S, s_C given ξ by averaging over the set of TAN models (equation 5).

Let $Q : \mathbb{R}^{n \times n} \rightarrow \mathbb{R}^{n-1 \times n-1}$. For any real $n \times n$ matrix $\boldsymbol{\tau}$ we define $Q(\boldsymbol{\tau})$ as the first $n-1$ lines and columns of the matrix $\overline{Q}(\boldsymbol{\tau})$ where

$$\overline{Q}_{u,v}(\boldsymbol{\tau}) = \overline{Q}_{v,u}(\boldsymbol{\tau}) = \begin{cases} -\tau_{u,v} & 1 \leq u < v \leq n \\ \sum_{v'=1}^{n} \tau_{v',v} & 1 \leq u = v \leq n \end{cases} \tag{16}$$

The integral for $P(\mathcal{V} = S, \mathcal{C} = s_C|\xi)$ can be calculated in closed form by applying the matrix tree theorem and expressed in terms of the previously introduced Q as:

$$P(\mathcal{V} = S, \mathcal{C} = s_C|\xi) = h_0^{S,s_C} |Q(\boldsymbol{\beta}\ \mathbf{h}^{S,s_C})| \tag{17}$$

where

$$h_0^{S,s_C} = \frac{1}{Z_\beta} \frac{1}{N'} \prod_{A_u \in V} N'_{u,C}(s_u, s_C) \tag{18}$$

$\mathbf{h}^{S,s_C} = \left(h_{u,v}^{S,s_C}\right)$ where

$$h_{u,v}^{S,s_C} = \frac{N'_{v,u,C}(s_v, s_u, s_C)}{N'_{u,C}(s_u, s_C) N'_{v,C}(s_v, s_C)} \tag{19}$$

The proof for this result appears in (Cerquides & López de Màntaras, 2003b).

3.4. Learning with Decomposable Distributions

Assume that the data is generated by a TAN model and that $P(M|\xi)$ follows a decomposable distribution with hyperparameters $\boldsymbol{\beta}$, $\mathbf{N}'$. Then, $P(M|\mathcal{D},\xi)$, the posterior probability distribution after observing a dataset $\mathcal{D}$ is a decomposable distribution with parameters $\boldsymbol{\beta}^*$, $\mathbf{N}'^*$ given by:

$$\beta^*_{u,v} = \beta_{u,v} W_{u,v} \tag{20}$$

$$N'^*_{u,v,C}(j,i,c) = N'_{u,v,C}(j,i,c) + N_{u,v,C}(j,i,c) \tag{21}$$

where

$$W_{u,v} = \prod_{c \in C} \prod_{i \in A_u} \frac{\Gamma(N'_{u,C}(i,c))}{\Gamma(N'_{u,C}(i,c) + N_{u,C}(i,c))} \prod_{c \in C} \prod_{j \in A_v} \frac{\Gamma(N'_{v,C}(j,c))}{\Gamma(N'_{v,C}(j,c) + N_{v,C}(j,c))} \prod_{c \in C} \prod_{i \in A_u} \prod_{j \in A_v} \frac{\Gamma(N'_{v,u,C}(j,i,c) + N_{v,u,C}(j,i,c))}{\Gamma(N'_{v,u,C}(j,i,c))} \tag{22}$$

The proof appears in (Cerquides & López de Màntaras, 2003b).

3.5. Putting it all Together

Putting together the results from sections 3.3 and 3.4 we can easily design a classifier based on decomposable distributions over TANs. The classifier works as follows: when given a dataset $\mathcal{D}$, it assumes that the data is generated from a TAN model and assumes a decomposable distribution as prior over the set of models. Applying the result from section 3.4, the posterior distribution over the set of models is also a decomposable distribution and applying the result of section 3.3 this decomposable posterior distribution can be used to calculate the probability of any observation S, s_C. When given an unclassified observation S, it can just calculate the probability $P(\mathcal{V} = S, \mathcal{C} = s_C|\mathcal{D},\xi)$ for each possible class $s_C \in C$ and classify S in the class with highest probability.

We have mentioned that the classifier assumes a decomposable distribution as prior. Ideally, this prior will be fixed by an expert that knows the classification

domain. Otherwise, we have to provide the classifier with a way for fixing the prior distribution hyperparameters without knowledge about the domain. In this case the prior should be as "non-informative" as possible in order for the information coming from $\mathcal{D}$ to dominate the posterior by the effects of equations 20 and 21. We have translated this requisite into equations 23 and 24:

$$\forall u, v \ ; \ 1 \leq u \neq v \ \leq n \ ; \ \beta_{u,v} = 1 \tag{23}$$

$$\forall u, v; 1 \leq u \neq v \leq n; \forall j \in A_v; \forall i \in A_u; \forall c \in C$$
$$N'_{v,u,C}(j, i, c) = \frac{\lambda}{\#C \# A_u \# A_v} \tag{24}$$

Defining $\boldsymbol{\beta}$ as in equation 23 means that we have the same amount of belief for any edge being in the TAN structure underlying the data. Fixed u, v, equation 24 assigns the same probability to any (j, i, c) such that $j \in A_v$, $i \in A_u$ and $c \in C$ and the value assigned is coherent with the multinomial sampling approach. λ is an "equivalent sample size" for the prior in the sense of Heckerman et al. in (Heckerman et al., 1995). In our experiments we have fixed $\lambda = 10$. In the following TBMATAN will refer to the classifier described in this section.

4. Approximating TBMATAN

TBMATAN can theoretically be implemented by an algorithm with $\mathcal{O}(N \cdot n^2)$ learning time and $\mathcal{O}(\#C \cdot n^3)$ time for classifying a new observation. In spite of that, a straightforward implementation of TBMATAN, even when accomplishing these complexity bounds, will not yield accurate results, specially for large datasets. This is due to the fact that the calculations that need to be done in order to classify a new observation include the computation of a determinant (in equation 17) that happens to be ill-conditioned. Even worse, the determinant gets more and more ill-conditioned as the number of observations in the dataset increases. This forces the floating point accuracy that we have to use to calculate these determinants to depend on the dataset size. We have calculated the determinants by means of NTL (Shoup, 2003), a library that allows us to calculate determinants with the desired precision arithmetic. This solution makes the time for classifying a new observation grow faster than $\mathcal{O}(\#C \cdot n^3)$, and hence makes the practical application of the algorithm difficult in situations where it is required to classify a large set of unclassified data.

We analyzed what makes the determinant being ill-conditioned and concluded that it is due to the $W_{u,v}$ factors given by equation 22. The factor $W_{u,v}$ could be interpreted as "how much the dataset $\mathcal{D}$ has changed the belief in that there is a link between u and v in the TAN model generating the data". The problems relies in the fact that $W_{u,v}$ are easily in the order of 10^{-200} for a dataset with 1500 observations. Furthermore, the factors $\frac{W_{u,v}}{W_{u',v'}}$ for such a dataset can be around 10^{-20}, providing the ill-condition of the determinant. In order to overcome this problem, we propose to postprocess the factors $W_{u,v}$ computed by equation 22 by means of a transformation that limits them to lie in the interval $[10^{-K}, 1]$ where K is a constant that has to be fixed depending on the floating point accuracy of the machine. In our implementation we have used $K = 5$. The transformation works as depicted in figure 2 and

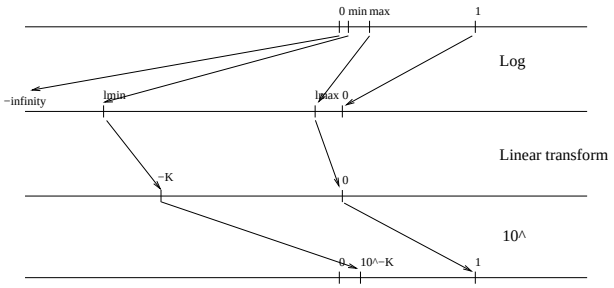


Figure 2. Transformation of weights for SSTBMATAN

is described in detail by the following equations:

$$lmax = \log_{10} \max_{\substack{u \in V \\ v \in V \\ u \neq v}} W_{u,v} \tag{25}$$

$$lmin = \log_{10} \min_{\substack{u \in V \\ v \in V \\ u \neq v}} W_{u,v} \tag{26}$$

$$a = \begin{cases} \frac{K}{lmax - lmin} & lmax - lmin > K \\ 1 & otherwise \end{cases} \tag{27}$$

$$b = -K - a * lmin \tag{28}$$

$$\widetilde{W}_{u,v} = 10^{a \log_{10}(W_{u,v}) + b} \tag{29}$$

Using $\widetilde{W}_{u,v}$ instead of $W_{u,v}$ to calculate the posterior hyperparameters $\beta^*_{u,v}$ has the following properties:

1. It is harder to get get ill-conditioned determinants, because for all u, v $\widetilde{W}_{u,v}$ is bound to the interval $[10^{-K}, 1]$.
2. It preserves the relative ordering of the $W_{u,v}$. That is, if $W_{u,v} > W_{u',v'}$ then $\widetilde{W}_{u,v} > \widetilde{W}_{u',v'}$.
3. It does not exaggerate relative differences in belief. That is, for all u, v, u', v' we have that
   - If $\frac{W_{u,v}}{W_{u',v'}} \geq 1$ then $\frac{W_{u,v}}{W_{u',v'}} \geq \frac{\widetilde{W}_{u,v}}{\widetilde{W}_{u',v'}}$.

- If $\frac{W_{u,v}}{W_{u',v'}} \leq 1$ then $\frac{W_{u,v}}{W_{u',v'}} \leq \frac{\widetilde{W}_{u,v}}{\widetilde{W}_{u',v'}}$.

The posterior hyperparameters $\beta^*_{u,v}$ can be interpreted as a representation of the a posteriori belief in the existence of an edge (u, v) in the TAN structure. Using $\widetilde{W}_{u,v}$, given the properties stated, means being more conservative in the structure learning process, because the beliefs will be confined to the interval $[10^{-K}, 1]$ which impedes the representation of extreme probability differences between edges. We can interpret the transformation as applying some stubbornness to the structure learning process. Applying this transformation allows us to implement an approximation of TBMATAN that does not require the use of special floating point accuracy computations. We will refer to this approximation of TBMATAN as SSTBMATAN (from Structure Stubborn TBMATAN).

It is worth noting that the problem described in this section does only affect the classification time. The learning process for TBMATAN does not need high precision arithmetics. The learning time complexity for TBMATAN, $\mathcal{O}(N \cdot n^2)$, is the same as the one for TAN. In spite of that, in practice, TAN learning time would be somewhat longer because the learning stage for TBMATAN (calculating every $N_{v,u,C}(j, i, c)$) is only the first step of the TAN learning process.

5. Empirical Results

We tested four algorithms over 16 datasets from the Irvine repository (Blake et al., 1998). The dataset characteristics are described in Table 1. To discretize continuous attributes we used equal frequency discretization with 5 intervals. For each dataset and algorithm we tested both accuracy and *LogScore*. *LogScore* is calculated by adding the minus logarithm of the probability assigned by the classifier to the correct class and gives an idea of how well the classifier is estimating probabilities (the smaller the score the better the result). If we name the test set $\mathcal{D}'$ we have

$$LogScore(M, \mathcal{D}') = \\ = \sum_{(S,s_C)\in\mathcal{D}'} -\log(P(\mathcal{C} = s_C|\mathcal{V} = S, M)) \quad (30)$$

For the evaluation of both error rate and *LogScore* we used 10 fold cross validation. We tested the algorithm with the 10%, 50% and 100% of the learning data for each fold, in order to get an idea of the influence of the amount of data in the behaviors of both error rate and *LogScore* for the algorithm. In order to evaluate the statistical significance of the differences we performed a paired t-test at 5%.

Detailed experimental results can be found in (Cerquides & López de Màntaras, 2003b).

The classifiers under comparison are:

- TAN+MS: Single TAN induction using the *multinomial sampling approach* (Cerquides, 1999).
- TBMATAN: The method described in section 3.5.
- SSTBMATAN: The method presented in section 4.

TBMATAN classification times are very large for datasets with a large number of instances. For datasets over 5000 instances we have skipped the execution of TBMATAN.

Table 1. Datasets information

Dataset	Attributes	Instances	Classes	Missing
ADULT	14	48842	2	some
BREAST	10	699	2	16
CAR	6	1728	4	no
CHESS	36	3196	2	no
CLEVE	13	303	2	some
CRX	15	690	2	few
FLARE	10	323	4	no
GLASS	10	214	2	none
HEP	19	155	2	some
IRIS	4	150	3	none
LETTER	16	20000	26	none
MUSHROOM	22	8124	2	some
NURSERY	8	12960	5	no
PIMA	8	768	2	no
SOYBEAN	35	316	19	some
VOTES	16	435	2	few

5.1. Interpretation of the Results

Statistical significance results can be seen in tables 2 and 3 where each entry in the table contains the number of datasets for which the error rate (resp. *LogScore*) for the classifer in the left column was better than the same measure for the classifier on the top row in a statistically significant way. For example, the improvement in *LogScore* provided by SSTBMATAN with respect to TAN+MS is statistically significant for 14 datasets when taking the 10% of the training data, for 11 datasets when taking the 50% and for 6 datasets when taking all of the learning data.

In many cases TBMATAN improves both accuracy and *LogScore* with respect to TAN+MS. The average relative improvement is around 10% for error rate and slightly higher for *LogScore*. The percentage of improvement is higher as we reduce the amount of learning data. This is understandable, because it is reasonable to think that if we have enough data, the posterior is likely to be concentrated around the tree learned by TAN+MS.

SSTBMATAN performs even slightly better than TBMATAN for many datasets, so we can accept that the

Table 2. Statistically significant differences in error rate

	TBMATAN			SSTBMATAN			TAN+MS		
	.1	.5	1	.1	.5	1	.1	.5	1
TBMATAN	-			1	1	1	5	5	3
SSTBMATAN	2	2	1	-			8	8	4
TAN+MS	0	0	1	1	2	2	-		

Table 3. Statistically significant differences in *LogScore*

	TBMATAN			SSTBMATAN			TAN+MS		
	.1	.5	1	.1	.5	1	.1	.5	1
TBMATAN	-			1	1	4	11	7	5
SSTBMATAN	7	7	6	-			14	11	6
TAN+MS	0	0	2	2	3	4	-		

approximation introduced in section 4 is good for datasets of this size. Finally, if we compare SSTBMATAN and TAN+MS we can see that its relative behavior is very similar to the one of TBMATAN with TAN+MS.

6. Conclusions and Future Work

We have introduced TBMATAN a classifier based on TAN, decomposable distributions and Bayesian model averaging. We have seen that its implementation leads to the calculation of ill-conditioned determinants and have proposed to use an approximated implementation: SSTBMATAN .

SSTBMATAN is, to the best of our knowledge, the most accurate classifier reported with a learning time linear on the number of observations of the dataset. The accuracy increase comes at the price of increasing the classification time, making it cubic on the number of attributes. The algorithm is anytime and incremental: as long as the dataset observations are processed randomly, we can stop the learning stage anytime we need, perform some classifications and then continue learning at the only (obvious) cost of the lower accuracy of the classifications performed in the middle of the learning process. These characteristics make the algorithm very suitable for datasets with a large number of instances.

Being able to calculate some measure of the concentration of the posterior distribution around the TAN learned by TAN+MS (that is, some sort of "variance") will probably allow us to determine beforehand whether TBMATAN will provide significant improvement over TAN+MS in a dataset.

Finally, we think that all of the classifiers reviewed in (Friedman et al., 1997) that are based on the Chow and Liu algorithm (Chow & Liu, 1968) can benefit from an improvement similar to the one seen here by the use of decomposable distributions and Bayesian model averaging. Formalizing the development for these classifiers and performing the empirical tests remains as future work.

References

Blake, C., Keogh, E., & Merz, C. (1998). UCI repository of machine learning databases.

Cerquides, J. (1999). Applying General Bayesian Techniques to Improve TAN Induction. *Proceedings of the International Conference on Knowledge Discovery and Data Mining, KDD99*.

Cerquides, J., & López de Màntaras, R. (2003a). The indifferent naive bayes classifier. *Proceedings of the 16th International FLAIRS Conference*.

Cerquides, J., & López de Màntaras, R. (2003b). *Tractable bayesian learning of tree augmented naive bayes classifiers* (Technical Report IIIA-2003-04). http://www.iiia.csic.es/∼mantaras/ReportIIIA-TR-2003-04.pdf.

Chow, C., & Liu, C. (1968). Aproximatting Discrete Probability Distributions with Dependence Trees. *IEEE Transactions on Information Theory*, *14*, 462–467.

Domingos, P., & Pazzani, M. (1997). On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. *Machine Learning*, *29*, 103–130.

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

Heckerman, D., Geiger, D., & Chickering, D. (1995). Learning bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, *20*, 197–243.

Hoeting, J., Madigan, D., Raftery, A., & Volinsky, C. (1998). *Bayesian model averaging* (Technical Report 9814). Department of Statistics. Colorado State University.

Kontkanen, P., Myllymaki, P., Silander, T., & Tirri, H. (1998). Bayes Optimal Instance-Based Learning. *Machine Learning: ECML-98, Proceedings of the 10th European Conference* (pp. 77–88). Springer-Verlag.

Langley, P., Iba, W., & Thompson, K. (1992). An Analysis of Bayesian Classifiers. *Proceedings of the Tenth National Conference on Artificial Intelligence* (pp. 223–228). AAAI Press and MIT Press.

Meila, M., & Jaakkola, T. (2000). *Tractable bayesian learning of tree belief networks* (Technical Report CMU-RI-TR-00-15). Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Shoup, V. (2003). NTL: A library for doing number theory. http://www.shoup.net/ntl.