In *Proceedings of the IADIS International Conference Intelligent Systems and Agents 2007*. Lisbon, Portugal, 2007.

Calculating the Normalized Maximum Likelihood Distribution for Bayesian Forests

Hannes Wettig Petri Kontkanen
Petri Myllymäki
Complex Systems Computation Group (CoSCo)
Helsinki Institute for Information Technology (HIIT)
University of Helsinki & Helsinki University of Technology
P.O.Box 68 (Department of Computer Science)
FIN-00014 University of Helsinki, Finland
{Firstname}.{Lastname}@hiit.fi

ABSTRACT

When learning Bayesian network structures from sample data, an important issue is how to evaluate the goodness of alternative network structures. Perhaps the most commonly used model (class) selection criterion is the marginal likelihood, which is obtained by integrating over a prior distribution for the model parameters. However, the problem of determining a reasonable prior for the parameters is a highly controversial issue, and no completely satisfying Bayesian solution has yet been presented in the non-informative setting. The normalized maximum likelihood (NML), based on Rissanen's information-theoretic MDL methodology, offers an alternative, theoretically solid criterion that is objective and non-informative, while no parameter prior is required. It has been previously shown that for discrete data, this criterion can be computed in linear time for Bayesian networks with no arcs, and in quadratic time for the so called Naive Bayes network structure. Here we extend the previous results by showing how to compute the NML criterion in polynomial time for tree-structured Bayesian networks. The order of the polynomial depends on the number of values of the variables, but neither on the number of variables itself, nor on the sample size.

KEYWORDS

Machine Learning, Bayesian Networks, Minimum Description Length, Normalized Maximum Likelihood.

1 INTRODUCTION

We consider the problem of learning a Bayesian network structure, based on a sample of data collected from the domain to be studied. We focus on the *score-based* approach, where first a model selection score is defined, yielding a goodness criterion that can be used for comparing different model structures, and any search method of choice can then be used for finding the structure with the highest score.

In this paper we study the problem of choosing and computing an appropriate model selection criterion. Naturally, any reasonable criterion must possess some desirable optimality properties. For a Bayesian, the most obvious choice is to use the model structure posterior, given the data and some model structure prior that has to be fixed in advance. Assuming a uniform prior over the possible structures, this leaves us with the *marginal likelihood*, which is the most commonly used criterion for learning Bayesian networks. Calculation of the marginal likelihood requires us to define a prior distribution over the parameters defined by the model structure under consideration. Under certain

assumptions, computing the marginal likelihood is then straightforward, see e.g. [1, 2]. Perhaps somewhat surprisingly, determining an adequate prior for the model parameters of a given class, in an objective manner has turned out to be a most difficult problem.

The uniform parameter prior sounds like the obvious candidate for a *non-informative* prior distribution, but it is not transformation-invariant, and produces different marginal likelihood scores for dependence-equivalent model structures [2]. This is due to the fact that there is no objective way of defining uniformity, but any prior can be uniform at most *with respect to a chosen representation*. The problem of transformation-invariance can be remedied by using the prior distribution suggested in [3], but this still leaves us with a single parameter, the *equivalent sample size*, the value of which is highly critical with respect to the result of the model structure search. Alternatively, one might resort to using the transformation-invariant Jeffreys prior, but although it can in the Bayesian network setting be formulated explicitly [4], computing it appears to be quite difficult in practice.

For the above reasons, in this paper we take the alternative approach of using the information-theoretic *normalized maximum likelihood* (NML) criterion [5, 6] as the model selection criterion. The NML score is – under certain conditions – asymptotically equivalent to the marginal likelihood with the Jeffreys prior [6], but it does not require us to define a prior distribution on the model parameters. Based on the data at hand only, it is fully objective, non-informative and transformation-invariant. What is more, the NML distribution can be shown to be the optimal distribution in a certain intuitively appealing sense. It may be used for selection of a model class among very different candidates. We need not assume a model family of nested model classes or the like, but we may compete against each other any types of model classes for which we can compute the NML distribution. Consequently, the NML score for Bayesian networks is of great importance both as a theoretically interesting problem and as a practically useful model selection criterion.

Although the NML criterion yields a theoretically very appealing model selection criterion, its usefulness in practice depends on the computational complexity of the method. In this paper we consider Bayesian network models for discrete data, where all the conditional distributions between the variables are assumed to be multinomial. For a single multinomial variable (or, an empty Bayesian network with no arcs), the value of the NML criterion can be computed in linear time [7], and for the Naive Bayes structure in quadratic time [8]. In this paper we consider more general forest-shaped network structures, and introduce an algorithm for computing the NML score in polynomial time – where the order of the polynomial depends on the number of possible values of the network variables. Although the problem of computing the NML for general Bayesian network structures remains unsolved, this work represents another step towards that goal.

The paper is structured as follows. In Section 2 we briefly review some basic properties of the NML distribution. Section 3 introduces the Bayesian Forest model family and some inevitable notation. The algorithm that calculates the NML distribution for Bayesian forests is developed in Section 4 and summarized in Section 5. We close with the concluding remarks of Section 6.

2 PROPERTIES OF THE NML DISTRIBUTION

The NML distribution, founding on the *Minimum Description Length* (MDL) principle, has several desirable properties. Firstly, it automatically protects against overfitting in the model class selection process. Secondly, there is no need to assume that there exists some underlying "true" model, while most other statistical methods do: in NML the model class is only used as a technical device to describe the data, not as a hypothesis. Consequently, the model classes amongst which to choose are allowed to be of utterly different types; any collection of model classes may be considered as long as the corresponding NML distributions can be computed. For this reason we find it important to push the boundaries of NML computability and develop algorithms that extend to more and more complex model families.

NML is closely related to Bayesian inference. There are, however, some fundamental differences dividing the two, the most important being that NML is not dependent on any prior distribution, it only uses the data at hand. For more discussion on the theoretical motivations behind NML and the MDL principle see, e.g., [6, 9, 10, 11, 12, 13].

In the following, we give the definition of the NML distribution and discuss some of its theoretical properties.

2.1 Definition of a Model Class and Family

Let $\mathbf{x}^n = (x_1, \ldots, x_n)$ be a data sample of n outcomes, where each outcome x_j is an element of some space of observations $\mathcal{X}$. The n-fold Cartesian product $\mathcal{X} \times \cdots \times \mathcal{X}$ is denoted by $\mathcal{X}^n$, so that $\mathbf{x}^n \in \mathcal{X}^n$. Consider a set $\Theta \subseteq \mathbb{R}^d$, where d is a positive integer. A class of parametric distributions indexed by the elements of Θ is called a *model class*. That is, a model class $\mathcal{M}$ is defined as

$$\mathcal{M} = \{P(\cdot \mid \boldsymbol{\theta}) : \boldsymbol{\theta} \in \Theta\}, \tag{1}$$

and the set Θ is called a *parameter space*.

Consider a set $\Phi \subseteq \mathbb{R}^e$, where e is a positive integer. Define a set $\mathcal{F}$ by

$$\mathcal{F} = \{\mathcal{M}(\boldsymbol{\phi}) : \boldsymbol{\phi} \in \Phi\}. \tag{2}$$

The set $\mathcal{F}$ is called a *model family*, and each of the elements $\mathcal{M}(\boldsymbol{\phi})$ is a model class. The associated parameter space is denoted by $\Theta_{\boldsymbol{\phi}}$. The model class selection problem can now be defined as a process of finding the parameter vector $\boldsymbol{\phi}$, which is optimal according to some pre-determined criteria.

2.2 The NML Distribution

One of the most theoretically and intuitively appealing model class selection criteria is the *Normalized Maximum Likelihood*. Denote the parameter vector that maximizes the likelihood of data $\mathbf{x}^n$ for a given model class $\mathcal{M}(\boldsymbol{\phi})$ by $\hat{\boldsymbol{\theta}}(\mathbf{x}^n, \mathcal{M}(\boldsymbol{\phi}))$:

$$\hat{\boldsymbol{\theta}}(\mathbf{x}^n, \mathcal{M}(\boldsymbol{\phi})) = \arg\max_{\boldsymbol{\theta} \in \Theta_{\boldsymbol{\phi}}} \{P(\mathbf{x}^n \mid \boldsymbol{\theta})\}. \tag{3}$$

The *normalized maximum likelihood* (NML) distribution [5] is now defined as

$$P_{\mathrm{NML}}(\mathbf{x}^n \mid \mathcal{M}(\boldsymbol{\phi})) = \frac{P(\mathbf{x}^n \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n, \mathcal{M}(\boldsymbol{\phi})))}{\mathcal{C}(\mathcal{M}(\boldsymbol{\phi}), n)}, \tag{4}$$

where the normalizing term $\mathcal{C}(\mathcal{M}(\boldsymbol{\phi}), n)$ in the case of discrete data is given by

$$\mathcal{C}(\mathcal{M}(\boldsymbol{\phi}), n) = \sum_{\mathbf{y}^n \in \mathcal{X}^n} P(\mathbf{y}^n \mid \hat{\boldsymbol{\theta}}(\mathbf{y}^n, \mathcal{M}(\boldsymbol{\phi}))), \tag{5}$$

and the sum goes over the space of data samples of size n. If the data is continuous, the sum is replaced by the corresponding integral. From this definition, it is immediately evident that NML is invariant with respect to any kind of parameter transformation, since such transformation does not affect the maximum likelihood $P(\mathbf{x}^n \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n, \mathcal{M}(\boldsymbol{\phi})))$.

In the MDL literature – which views the model class selection problem as a task of minimizing the resulting code length – the minus logarithm of (4) is referred to as the *stochastic complexity* of the data $\mathbf{x}^n$ given model class $\mathcal{M}(\boldsymbol{\phi})$ and the logarithm of the normalizing sum $\log \mathcal{C}(\mathcal{M}(\boldsymbol{\phi}), n)$ is referred to as the *parametric complexity* or *(minimax) regret* of $\mathcal{M}(\boldsymbol{\phi})$.

The NML distribution (4) has several important theoretical optimality properties. The first one is that NML provides a unique solution to the minimax problem posed in [5],

$$\min_{\hat{P}} \max_{\mathbf{x}^n} \log \frac{P(\mathbf{x}^n \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n, \mathcal{M}(\boldsymbol{\phi})))}{\hat{P}(\mathbf{x}^n \mid \mathcal{M}(\boldsymbol{\phi}))} \tag{6}$$

i.e., the minimizing $\hat{P}$ is the NML distribution, and it assigns a probability to any data that differs from the highest achievable probability within the model class – the *maximum likelihood*

$P(\mathbf{x}^n \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n, \mathcal{M}(\boldsymbol{\phi})))$ – by the constant factor $\mathcal{C}(\mathcal{M}(\boldsymbol{\phi}), n)$. In this sense, the NML distribution can be seen as a truly uniform prior, with respect to the data itself, not its representation by a model class $\mathcal{M}(\boldsymbol{\phi})$. In other words, the NML distribution is the *minimax optimal universal model*. The term universal model in this context means that the NML distribution represents (or mimics) the behaviour of all the distributions in the model class $\mathcal{M}(\boldsymbol{\phi})$. Note that the NML distribution itself does not have to belong to the model class, and typically it does not.

A related property of NML was proven in [11]. It states that NML also minimizes

$$\min_{\hat{P}} \max_{g} E_g \log \frac{P(\mathbf{x}^n \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n, \mathcal{M}(\boldsymbol{\phi})))}{\hat{P}(\mathbf{x}^n \mid \mathcal{M}(\boldsymbol{\phi}))} \tag{7}$$

where the expectation is taken over $\mathbf{x}^n$ and g is the worst-case data generating distribution.

3 THE BAYESIAN FOREST MODEL FAMILY

We assume m variables $X_1, \ldots, X_m$ with given value cardinalities $K_1, \ldots, K_m$. We further assume a data matrix $\mathbf{x}^n = (x_{ji}) \in \mathcal{X}^n$, $1 \leq j \leq n$ and $1 \leq i \leq m$, given.

A Bayesian network structure $\mathcal{G}$ encodes independence assumptions so that if each variable X_i is represented as a node in the network, then the joint probability distribution factorizes into a product of local probability distributions, one for each node, conditioned on its parent set. We define a *Bayesian forest* (BF) to be a Bayesian network structure $\mathcal{G}$ on the node set $X_1, \ldots, X_m$ which assigns at most one parent $X_{pa(i)}$ to any node X_i. Consequently, a *Bayesian tree* is a connected Bayesian forest and a Bayesian forest breaks down into component trees, i.e. connected subgraphs. The root of each such component tree lacks a parent, in which case we write $pa(i) = \emptyset$.

The parent set of a node X_i thus reduces to a single value $pa(i) \in \{1, \ldots, i-1, i+1, \ldots, m, \emptyset\}$. Let further $ch(i)$ denote the set of children of node X_i in $\mathcal{G}$ and $ch(\emptyset)$ denote the "children of none", i.e. the roots of the component trees of $\mathcal{G}$.

The corresponding model family $\mathcal{F}_{\mathrm{BF}}$ can be indexed by the network structure $\mathcal{G} \in \Phi_{BF} \subset \mathbb{N} \subset \mathbb{R}$ according to some enumeration of all Bayesian forests on $(X_1, \ldots, X_m)$:

$$\mathcal{F}_{BF} = \{\mathcal{M}(\mathcal{G}) : \mathcal{G} \text{ is a forest}\}. \tag{8}$$

Given a forest model class $\mathcal{M}(\mathcal{G})$, we index each model by a parameter vector $\boldsymbol{\theta}$ in the corresponding parameter space $\Theta_{\mathcal{G}}$.

$$\Theta_{\mathcal{G}} = \{\boldsymbol{\theta} = (\theta_{ikl}) : \ \theta_{ikl} \geq 0, \ \sum_{l} \theta_{ikl} = 1, \ i = 1, \ldots, m, \ k = 1, \ldots, K_{pa(i)}, \ l = 1, \ldots, K_i\}, \tag{9}$$

where we define $K_\emptyset := 1$ in order to unify notation for root and non-root nodes. Each such θ_{ikl} defines a probabilty

$$\theta_{ikl} = P(X_i = l \mid X_{pa(i)} = k, \ \mathcal{M}(\mathcal{G}), \ \boldsymbol{\theta}) \tag{10}$$

where we interpret $X_\emptyset = 1$ as a null condition.

The joint probability distribution that such a model $M = (\mathcal{G}, \ \boldsymbol{\theta})$ assigns to a data vector $\mathbf{x} = (x_1, \ldots, x_m)$ becomes

$$P(\mathbf{x} \mid \mathcal{M}(\mathcal{G}), \boldsymbol{\theta}) = \prod_{i=1}^{m} P(X_i = x_i \mid X_{pa(i)} = x_{pa(i)}, \mathcal{M}(\mathcal{G}), \boldsymbol{\theta}) = \prod_{i=1}^{m} \theta_{i, x_{pa(i)}, x_i}. \tag{11}$$

For a sample $\mathbf{x}^n = (x_{ji})$ of n vectors $\mathbf{x}_j$ we define the corresponding frequencies

$$f_{ikl} := |\{j : x_{ji} = l \ \wedge \ x_{j,pa(i)} = k\}| \quad \text{and} \quad f_{il} := |\{j : x_{ji} = l\}| = \sum_{k=1}^{K_{pa(i)}} f_{ikl}. \tag{12}$$

By definition, for any component tree root X_i we have $f_{il} = f_{i1l}$. The probability assigned to an i.i.d. sample $\mathbf{x}^n$ can then be written as

$$P(\mathbf{x}^n \mid \mathcal{M}(\mathcal{G}), \boldsymbol{\theta}) = \prod_{i=1}^{m} \prod_{k=1}^{K_{pa(i)}} \prod_{l=1}^{K_i} \theta_{ikl}^{f_{ikl}}, \tag{13}$$

which is maximized at

$$\hat{\theta}_{ikl}(\mathbf{x}^n, \mathcal{M}(\mathcal{G})) = \frac{f_{ikl}}{f_{pa(i),k}}, \tag{14}$$

where we define $f_{\emptyset,1} := n$. The maximum data likelihood thereby is

$$P(\mathbf{x}^n \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n, \mathcal{M}(\mathcal{G}))) = \prod_{i=1}^{m} \prod_{k=1}^{K_{pa(i)}} \prod_{l=1}^{K_i} \left(\frac{f_{ikl}}{f_{pa(i),k}} \right)^{f_{ikl}}. \tag{15}$$

4 CALCULATING THE NML DISTRIBUTION

The goal is to calculate the NML distribution $P_{\mathrm{NML}}(\mathbf{x}^n \mid \mathcal{M}(\mathcal{G}))$ defined in (4). This consists of calculating the maximum data likelihood (15) and the normalizing term $\mathcal{C}(\mathcal{M}(\mathcal{G}), n)$ given in (5). The former involves frequency counting – one sweep through the data – and multiplication of the appropriate values. This can be done in time $\mathcal{O}(n + \sum_i K_i K_{pa(i)})$. The latter involves a sum exponential in n, which clearly makes it the computational bottleneck of the algorithm.

Our approach is to break up the normalizing sum in (5) into terms corresponding to subtrees with given frequencies in either their root or its parent. We then calculate the complete sum by sweeping through the graph once, bottom-up. The exact ordering will be irrelevant, as long as we deal with each node before its parent. Let us now introduce the needed notation.

Let $\mathcal{G}$ be a given Bayesian forest. In order to somewhat shorten our notation, from now on we do not write out the model class $\mathcal{M}(\mathcal{G})$ anymore, as it may be assumed fixed. We thus write e.g. $P(\mathbf{x}^n \mid \boldsymbol{\theta})$, meaning $P(\mathbf{x}^n \mid \boldsymbol{\theta}, \mathcal{M}(\mathcal{G}))$. When in the following we restrict to subsets of the attribute space, we implicitly restrict the model class accordingly, e.g. in (16) below, we write $P(\mathbf{x}^n_{sub(i)} \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n_{sub(i)}))$ as a short notation for $P(\mathbf{x}^n_{sub(i)} \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n_{sub(i)}), \mathcal{M}(\mathcal{G}_{sub(i)}))$.

For any node X_i denote the subtree rooting in X_i by $\mathcal{G}_{sub(i)}$ and the forest built up by all descendants of X_i by $\mathcal{G}_{dsc(i)}$. The corresponding data domains are $\mathcal{X}_{sub(i)}$ and $\mathcal{X}_{dsc(i)}$, respectively. Denote the partial normalizing sum over all n-instantiations of a subtree by

$$\mathcal{C}_i(n) := \sum_{\mathbf{x}^n_{sub(i)} \in \mathcal{X}^n_{sub(i)}} P(\mathbf{x}^n_{sub(i)} \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n_{sub(i)})) \tag{16}$$

and for any vector $\mathbf{x}_i^n \in X_i^n$ with frequencies $\mathbf{f}_i = (f_{i1}, \ldots, f_{iK_i})$ we define

$$\mathcal{C}_i(n \mid \mathbf{f}_i) := \sum_{\mathbf{x}^n_{dsc(i)} \in \mathcal{X}^n_{dsc(i)}} P(\mathbf{x}^n_{dsc(i)}, \mathbf{x}^n_i \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n_{dsc(i)}, \mathbf{x}^n_i)) \tag{17}$$

to be the corresponding sum with fixed root instantiation, summing only over the attribute space spanned by the descendants on X_i. Note, that we condition on $\mathbf{f}_i$ on the left-hand side, and on $\mathbf{x}_i^n$ on the right-hand side of the definition. This needs to be justified. Interestingly, while the terms in the sum depend on the ordering of $\mathbf{x}_i^n$, the sum itself depends on $\mathbf{x}_i^n$ only through its frequencies $\mathbf{f}_i$. To see this pick any two representatives $\mathbf{x}_i^n$ and $\bar{\mathbf{x}}_i^n$ of $\mathbf{f}_i$ and find, e.g. after lexicographical ordering of the elements, that

$$\{(\mathbf{x}_i^n, \mathbf{x}^n_{dsc(i)}) : \mathbf{x}^n_{dsc(i)} \in \mathcal{X}^n_{dsc(i)}\} = \{(\bar{\mathbf{x}}_i^n, \mathbf{x}^n_{dsc(i)}) : \mathbf{x}^n_{dsc(i)} \in \mathcal{X}^n_{dsc(i)}\} \tag{18}$$

Next, we need to define corresponding sums over $\mathcal{X}_{sub(i)}$ with the frequencies at the subtree root parent $X_{pa(i)}$ given. For any $\mathbf{f}_{pa(i)} \sim \mathbf{x}^n_{pa(i)} \in X^n_{pa(i)}$ define

$$\mathcal{L}_i(n \mid \mathbf{f}_{pa(i)}) := \sum_{\mathbf{x}^n_{sub(i)} \in \mathcal{X}^n_{sub(i)}} P(\mathbf{x}^n_{sub(i)} \mid \mathbf{x}^n_{pa(i)}, \hat{\boldsymbol{\theta}}(\mathbf{x}^n_{sub(i)}, \mathbf{x}^n_{pa(i)})) \tag{19}$$

Again, this is well-defined since any other representative $\bar{\mathbf{x}}^n_{pa(i)}$ of $\mathbf{f}_{pa(i)}$ yields summing the same terms in different order.

After having introduced this notation, we now briefly outline the algorithm and – in the following subsections – give a more detailed description of the steps involved. As stated before, we go through $\mathcal{G}$ bottom-up. At each inner node X_i, we receive $\mathcal{L}_j(n \mid \mathbf{f}_i)$ from each child X_j, $j \in ch(i)$. Correspondingly, we are required to send $\mathcal{L}_i(n \mid \mathbf{f}_{pa(i)})$ up to the parent $X_{pa(i)}$. At each component tree root X_i we then calculate the sum $\mathcal{C}_i(n)$ for the whole connectivity component and then combine these sums to get the normalizing sum $\mathcal{C}(n)$ for the complete forest $\mathcal{G}$.

4.1 Leaves

It turns out, that for a leave node X_i we can calculate the terms $\mathcal{L}_i(n \mid \mathbf{f}_{pa(i)})$ without listing the frequencies $\mathbf{f}_i$ at X_i itself. The parent frequencies $\mathbf{f}_{pa(i)}$ split the n data vectors into $K_{pa(i)}$ subsets of sizes $f_{pa(i),1}, \ldots, f_{pa(i),K_{pa(i)}}$ and each of them can be modelled independently as a multinomial. We have

$$\mathcal{L}_i(n \mid \mathbf{f}_{pa(i)}) = \prod_{k=1}^{K_{pa(i)}} \mathcal{C}_{\mathrm{MN}}(K_i, f_{pa(i),k}). \tag{20}$$

where

$$\mathcal{C}_{\mathrm{MN}}(K_i, n') = \sum_{\mathbf{x}_i \in X_i} P(\mathbf{x}_i^{n'} \mid \hat{\boldsymbol{\theta}}(x_i^{n'}), \mathcal{M}_{\mathrm{MN}}(K_i)) = \sum_{\mathbf{x}_i \in X_i} \prod_{l=1}^{K_i} \left(\frac{f_{il}}{n'}\right)^{f_{il}} \tag{21}$$

is the normalizing sum (5) for the multinomial model class $\mathcal{M}_{\mathrm{MN}}(K_i)$ for a single discrete variable with K_i values, see e.g. [8, 14, 7] for details. [7] derives a simple recurrence for these terms, namely

$$\mathcal{C}_{\mathrm{MN}}(K+2, n') = \mathcal{C}_{\mathrm{MN}}(K+1, n') + \frac{n'}{K}\mathcal{C}_{\mathrm{MN}}(K, n'), \tag{22}$$

which we can use to precalculate all $\mathcal{C}_{\mathrm{MN}}(K_i, n')$ (for $n' = 0, \ldots, n$) in linear time each, i.e. in quadratic time altogether, for details see [7].

4.2 Inner Nodes

For inner nodes X_i we divide the task into two steps. First collect the messages $\mathcal{L}_j(n \mid \mathbf{f}_i)$ sent by each child $X_j \in ch(i)$ into partial sums $\mathcal{C}_i(n \mid \mathbf{f}_i)$ over $\mathcal{X}_{dsc(i)}$, then "lift" these to sums $\mathcal{L}_i(n \mid \mathbf{f}_{pa(i)})$ over $\mathcal{X}_{sub(i)}$ which are the messages to the parent.

The first step is simple. Given an instantiation $\mathbf{x}_i^n$ at X_i or, equivalently, the corresponding frequencies $\mathbf{f}_i$, the subtrees rooting in the children $ch(i)$ of X_i become independent of each other.

Thus we have

$$\mathcal{C}_i(n \mid \mathbf{f}_i) = \sum_{\mathbf{x}^n_{dsc(i)} \in \mathcal{X}^n_{dsc(i)}} P(\mathbf{x}^n_{dsc(i)}, \mathbf{x}^n_i \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n_{dsc(i)}, \mathbf{x}^n_i)) \tag{23}$$

$$= P(\mathbf{x}^n_i \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n_{dsc(i)}, \mathbf{x}^n_i)) \left(\sum_{\mathbf{x}^n_{dsc(i)} \in \mathcal{X}^n_{dsc(i)}} \prod_{j \in ch(i)} P(\mathbf{x}^n_{dsc(i)|sub(j)} \mid \mathbf{x}^n_i, \hat{\boldsymbol{\theta}}(\mathbf{x}^n_{dsc(i)}, \mathbf{x}^n_i)) \right) \tag{24}$$

$$= P(\mathbf{x}^n_i \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n_{dsc(i)}, \mathbf{x}^n_i)) \prod_{j \in ch(i)} \left(\sum_{\mathbf{x}^n_{sub(j)} \in \mathcal{X}^n_{sub(j)}} P(\mathbf{x}^n_{sub(j)} \mid \mathbf{x}^n_i, \hat{\boldsymbol{\theta}}(\mathbf{x}^n_{dsc(i)}, \mathbf{x}^n_i)) \right) \tag{25}$$

$$= \prod_{l=1}^{K_i} \left(\frac{f_{il}}{n} \right)^{f_{il}} \prod_{j \in ch(i)} \mathcal{L}_j(n \mid \mathbf{f}_i) \tag{26}$$

where $\mathbf{x}^n_{dsc(i)|sub(j)}$ is the restriction of $\mathbf{x}_{dsc(i)}$ to columns corresponding to nodes in $\mathcal{G}_{sub(j)}$. We have used (17) for (23), (11) for (24) and (25) and finally (15) and (19) for (26).

Now we calculate the outgoing messages $\mathcal{L}_i(n \mid \mathbf{f}_{pa(i)})$ from the incoming messages we have just combined into $\mathcal{C}_i(n \mid \mathbf{f}_i)$. This is the most demanding part of the algorithm, as we need to list all possible conditional frequencies, of which there are $\mathcal{O}(n^{K_i K_{pa(i)} - 1})$ many, the -1 being due to the sum-to-n constraint. For fixed i, we arrange the conditional frequencies f_{ikl} into a matrix $\mathbf{F} = (f_{ikl})$ and define its marginals

$$\boldsymbol{\rho}(\mathbf{F}) := \left(\sum_k f_{ik1}, \ldots, \sum_k f_{ikK_i} \right) \quad \text{and} \quad \boldsymbol{\gamma}(\mathbf{F}) := \left(\sum_l f_{i1l}, \ldots, \sum_l f_{iK_{pa(i)}l} \right) \tag{27}$$

to be the vectors obtained by summing the rows of $\mathbf{F}$ and the columns of $\mathbf{F}$, respectively. Each such matrix then corresponds to a term $\mathcal{C}_i(n \mid \boldsymbol{\rho}(\mathbf{F}))$ and a term $\mathcal{L}_i(n \mid \boldsymbol{\gamma}(\mathbf{F}))$. Formally we have

$$\mathcal{L}_i(n \mid \mathbf{f}_{pa(i)}) = \sum_{\mathbf{F}: \boldsymbol{\gamma}(\mathbf{F}) = \mathbf{f}_{pa(i)}} \mathcal{C}_i(n \mid \boldsymbol{\rho}(\mathbf{F})). \tag{28}$$

4.3 Component Tree Roots

For a component tree root $X_i \in ch(\emptyset)$ we do not need to pass any message upward. All we need is the complete sum over the component tree

$$\mathcal{C}_i(n) = \sum_{\mathbf{f}_i} \frac{n!}{f_{i1}! \ldots f_{iK_i}!} \mathcal{C}_i(n \mid \mathbf{f}_i) \tag{29}$$

where the $\mathcal{C}_i(n \mid \mathbf{f}_i)$ are calculated using (26). The summation goes over all non-negative integer vectors $\mathbf{f}_i$ summing to n. The above is trivially true since we sum over all instantiations $\mathbf{x}^n_i$ of X^n_i and group like terms – corresponding to the same frequency vector $\mathbf{f}_i$ – keeping track of their respective count, namely $n!/(f_{i1}! \ldots f_{iK_i}!)$.

5 THE ALGORITHM

For the complete forest $\mathcal{G}$ we simply multiply the sums over its tree components. Since these are independent of each other, in analogy to (23)-(26) we have

$$\mathcal{C}(n) = \prod_{i \in ch(\emptyset)} \mathcal{C}_i(n). \tag{30}$$

Algorithm 1 collects all the above into pseudo-code.

Algorithm 1
Computing $P_{\mathrm{NML}}(\mathbf{x}^n)$ for a Bayesian Forest $\mathcal{G}$.

1: Count all frequencies f_{ikl} and f_{il} from the data $\mathbf{x}^n$
2: Compute $P(\mathbf{x}^n \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n))= \prod_{i=1}^{m} \prod_{k=1}^{K_{pa(i)}} \prod_{l=1}^{K_i} \left(\frac{f_{ikl}}{f_{pa(i),k}}\right)^{f_{ikl}}$
3: **for** $K' = 1, \ldots, K_{max} := \max_{i:X_i \text{ is a leaf}} \{K_i\}$ and $n' = 0, \ldots, n$ **do**
4: Compute $\mathcal{C}_{\mathrm{MN}}(K', n')$ using recurrence (22)
5: **end for**
6: **for** each node X_i in some bottom-up order **do**
7: **if** X_i is a leaf **then**
8: **for** each frequency vector $\mathbf{f}_{pa(i)}$ of $X_{pa(i)}$ **do**
9: Compute $\mathcal{L}_i(n \mid \mathbf{f}_{pa(i)}) = \prod_{k=1}^{K_{pa(i)}} \mathcal{C}_{\mathrm{MN}}(K_i, \mathbf{f}_{pa(i)k})$
10: **end for**
11: **else if** X_i is an inner node **then**
12: **for** each frequency vector $\mathbf{f}_i$ of X_i **do**
13: Compute $\mathcal{C}_i(n \mid \mathbf{f}_i) = \prod_{l=1}^{K_i} \left(\frac{f_{il}}{n}\right)^{f_{il}} \prod_{j \in ch(i)} \mathcal{L}_j(n \mid \mathbf{f}_i)$
14: **end for**
15: initialize $\mathcal{L}_i \equiv 0$
16: **for** each non-negative $K_i \times K_{pa(i)}$ integer matrix $\mathbf{F}$ with entries summing to n **do**
17: $\mathcal{L}_i(n \mid \boldsymbol{\gamma}(\mathbf{F})) \mathrel{+}= \mathcal{C}_i(n \mid \boldsymbol{\rho}(\mathbf{F}))$
18: **end for**
19: **else if** X_i is a component tree root **then**
20: Compute $\mathcal{C}_i(n) = \sum_{\mathbf{f}_i} \prod_{l=1}^{K_i} \left(\frac{f_{il}}{n}\right)^{f_{il}} \prod_{j \in ch(i)} \mathcal{L}_j(n \mid \mathbf{f}_i)$
21: **end if**
22: **end for**
23: Compute $\mathcal{C}(n) = \prod_{i \in ch(\emptyset)} \mathcal{C}_i(n)$
24: Output $P_{\mathrm{NML}}(\mathbf{x}^n) = \frac{P(\mathbf{x}^n \mid \hat{\boldsymbol{\theta}}(\mathbf{x}^n))}{\mathcal{C}(n)}$

The time complexity of this algorithm is $\mathcal{O}(n^{K_i K_{pa(i)} - 1})$ for each inner node, $\mathcal{O}(n(n + K_i))$ for each leaf and $\mathcal{O}(n^{K_i - 1})$ for a component tree root of $\mathcal{G}$. When all $m' < m$ inner nodes are binary it runs in $\mathcal{O}(m' n^3)$, independent of the number of values of the leaf nodes. This is polynomial wrt. the sample size n, while applying (5) directly for computing $\mathcal{C}(n)$ requires exponential time. The order of the polynomial depends on the attribute cardinalities: the algorithm is exponential wrt. the number of values a non-leaf variable can take.

Finally, note that we can speed up the algorithm when $\mathcal{G}$ contains multiple copies of some subtree. Also we have $\mathcal{C}_i/\mathcal{L}_i(n \mid \mathbf{f}_i) = \mathcal{C}_i/\mathcal{L}_i(n \mid \pi(\mathbf{f}_i))$ for any permutation π of the entries of $\mathbf{f}_i$. However, this does not lead to considerable gain, at least in $\mathcal{O}$-notation. Also, we can see that in line 16 of the algorithm we enumerate all frequency matrices $\mathbf{F}$, while in line 17 we sum the same terms whenever the marginals of $\mathbf{F}$ are the same. Unfortunately, computing the number of non-negative integer matrices with given marginals is a #P-hard problem already when one of the matrix dimensions is fixed to 2, as proven in [15]. This suggests that for this task there may not exist an algorithm that is polynomial in all input quantities. The algorithm presented here is polynomial in both the sample size n and the graph size m. For attributes with relatively few values, the polynomial is of tolerable degree.

6 CONCLUSION

The information-theoretic normalized maximum likelihood (NML) criterion offers an interesting, non-informative approach to Bayesian network structure learning. It has some links to the Bayesian marginal likelihood approach — NML converges asymptotically to the marginal likelihood with the

Jeffreys prior — but it avoids the technical problems related to parameter priors as no explicitly defined prior distributions are required. Unfortunately a straightforward implementation of the criterion requires exponential time. In this paper we presented a computationally feasible algorithm for computing the NML criterion for tree-structured Bayesian networks: Bayesian trees and forests (collections of trees).

The time complexity of the algorithm presented here is polynomial with respect to the sample size and the number of domain variables, but the order of the polynomial depends on the number of values of the inner nodes in the tree to be evaluated, which makes the algorithm impractical for some domains. However, we consider this result as an important extension of the earlier results which were able to handle only Naive Bayes structures, i.e., Bayesian trees of depth one with no inner nodes. In the future we plan to test the validity of the suggested NML approach in practical problem domains, and we also wish to extend this approach to more complex Bayesian network structures.

7 ACKNOWLEDGEMENTS

This work was supported in part by the Finnish Funding Agency for Technology and Innovation under projects PMMA, KUKOT and SIB, by the Academy of Finland under project CIVI, and by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

References

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

[2] D. Heckerman, D. Geiger, and D.M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, September 1995.

[3] W. Buntine. Theory refinement on Bayesian networks. In B. D'Ambrosio, P. Smets, and P. Bonissone, editors, *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, pages 52–60. Morgan Kaufmann Publishers, 1991.

[4] P. Kontkanen, P. Myllymäki, T. Silander, H. Tirri, and P. Grünwald. On predictive distributions and Bayesian networks. *Statistics and Computing*, 10:39–54, 2000.

[5] Yu M. Shtarkov. Universal sequential coding of single messages. *Problems of Information Transmission*, 23:3–17, 1987.

[6] J. Rissanen. Fisher information and stochastic complexity. *IEEE Transactions on Information Theory*, 42(1):40–47, January 1996.

[7] P. Kontkanen and P. Myllymäki. A linear-time algorithm for computing the multinomial stochastic complexity. *Submitted to Information Processing Letters*, 2007.

[8] P. Kontkanen, P. Myllymäki, W. Buntine, J. Rissanen, and H. Tirri. An MDL framework for data clustering. In P. Grünwald, I.J. Myung, and M. Pitt, editors, *Advances in Minimum Description Length: Theory and Applications*. The MIT Press, 2006.

[9] A. Barron, J. Rissanen, and B. Yu. The minimum description principle in coding and modeling. *IEEE Transactions on Information Theory*, 44(6):2743–2760, October 1998.

[10] Q. Xie and A.R. Barron. Asymptotic minimax regret for data compression, gambling, and prediction. *IEEE Transactions on Information Theory*, 46(2):431–445, March 2000.

[11] J. Rissanen. Strong optimality of the normalized ML models as universal codes and information in data. *IEEE Transactions on Information Theory*, 47(5):1712–1717, July 2001.

[12] P. Grünwald. Minimum description length tutorial. In P. Grünwald, I.J. Myung, and M. Pitt, editors, *Advances in Minimum Description Length: Theory and Applications*, pages 23–79. The MIT Press, 2006.

[13] J. Rissanen. Lectures on statistical modeling theory, August 2005. Available online at www.mdl-research.org.

[14] P. Kontkanen and P. Myllymäki. MDL histogram density estimation. In *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (to appear)*, San Juan, Puerto Rico, March 2007.

[15] M.E. Dyer, R. Kannan, and J. Mount. Sampling contingency tables. *Random Structures and Algorithms*, 10(4):487–506, 1997.