Min-BDeu and Max-BDeu Scores for Learning Bayesian Networks

Mauro Scanagatta, Cassio P. de Campos, and Marco Zaffalon

Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA), Switzerland
{mauro,cassio,zaffalon}@idsia.ch

Abstract. This work presents two new score functions based on the Bayesian Dirichlet equivalent uniform (BDeu) score for learning Bayesian network structures. They consider the sensitivity of BDeu to varying parameters of the Dirichlet prior. The scores take on the most adversary and the most beneficial priors among those within a contamination set around the symmetric one. We build these scores in such way that they are decomposable and can be computed efficiently. Because of that, they can be integrated into any state-of-the-art structure learning method that explores the space of directed acyclic graphs and allows decomposable scores. Empirical results suggest that our scores outperform the standard BDeu score in terms of the likelihood of unseen data and in terms of edge discovery with respect to the true network, at least when the training sample size is small. We discuss the relation between these new scores and the accuracy of inferred models. Moreover, our new criteria can be used to identify the amount of data after which learning is saturated, that is, additional data are of little help to improve the resulting model.

Keywords: Bayesian networks, structure learning, Bayesian Dirichlet score

1 Introduction

A Bayesian network is a versatile and well-known probabilistic graphical model with applications in a variety of fields. It relies on a structured dependency among random variables to represent a joint probability distribution in a compact and efficient manner. These dependencies are encoded by an acyclic directed graph (DAG) where nodes are associated to random variables and conditional probability distributions are defined for variables given their parents in the graph. Learning the graph (or structure) of Bayesian networks from data is one of its most challenging problems.

The topic of Bayesian network learning has been extensively discussed in the literature and many different approaches are available. In general terms, the problem is to find the structure that maximizes a given score function that depends on the data [1]. The research on this topic is very active, with numerous methods and papers [2, 3, 4, 5, 6, 7, 8, 9, 10]. The main characteristic tying together all these methods is the score function. Arguably, the most commonly used score function is the Bayesian Dirichlet (likelihood) equivalent uniform (BDeu), which derives from BDe and BD [11, 12, 1] (other examples of score functions are the Bayesian Information Criterion [13], which is equivalent to *Minimum Description Length*, and the Akaike Information Criterion [14]). There are also more recent attempts to devise new score functions. For example, [15] presents a score that aims at having its maximization computationally facilitated as the amount of data increases.

The BDeu score aims at maximizing the posterior probability of the DAG given data, while assuming a uniform prior over possible DAGs. In this work we propose two new score functions, namely Min-BDeu and Max-BDeu. These scores are based on the BDeu score, but they consider all possible prior probability distributions inside an ε-contaminated set [16] of Dirichlet priors around the symmetric one (which is the one used by the original BDeu). Min-BDeu is the score obtained by choosing the most adversary prior distributions (that is, those minimizing the score) from the contaminated sets, while Max-BDeu is the score that uses the most beneficial priors to maximize the resulting value. We demonstrate that Min-BDeu and Max-BDeu can be efficiently calculated and are decomposable. Because of that, any structure learning solver can be used to find the best scoring DAG with them. We empirically show that Min-BDeu achieves better predictive accuracy (based on the likelihood of held-out data) than the original BDeu for small sample sizes, and performs similarly to BDeu when the amount of data is large. On the other hand, Max-BDeu achieves better edge accuracy (evaluated by the Hamming distance between the set of edges of true and learned moralized graphs).

A very important question regarding structure learning is whether the result is *accurate*, that is, whether it produces a network that will give accurate results on future unseen data. In this regard, we empirically show an interesting relation between accuracy obtained with a given training sample size and the gap between Max-BDeu and Min-BDeu. This relation might be used to identify the amount of data that is necessary to obtain an accurate network, as we will discuss later on.

The paper is divided as follows. Section 2 defines Bayesian networks, introduces our notation and the problem of structure learning. Section 3 presents our new score functions and demonstrates the existence of efficient algorithms to compute them. Section 4 describes our experimental setting and discusses two experiments regarding the accuracy of Min-BDeu and the use of Max-BDeu and Min-BDeu to help in predicting the amount of data needed to achieve a desired learning accuracy. Finally, Section 5 concludes the paper and discusses future work.

2 Learning Bayesian Networks

A Bayesian network represents a joint probability distribution over a collection of random variables, which we assume to be categorical. It can be defined as a triple $(\mathcal{G}, \mathcal{X}, \mathcal{P})$, where $\mathcal{G} \doteq (V_\mathcal{G}, E_\mathcal{G})$ is an acyclic directed graph (DAG) with $V_\mathcal{G}$ a collection of n nodes associated to random variables $\mathcal{X}$ (a node per variable, which might be used interchangeably to denote each other), and $E_\mathcal{G}$ a collection of arcs; $\mathcal{P}$ is a collection of conditional mass functions $p(X_i|\Pi_i)$ (one for each instantiation of Π_i), where Π_i denotes the parents of X_i in the graph (Π_i may be empty), respecting the relations of $E_\mathcal{G}$. In a Bayesian network every variable is conditionally independent of its non-descendant non-parent variables given its parent variables (Markov condition).

We use uppercase letters such as X_i, X_j to represent variables (or nodes of the graph), and x_i to represent a generic state of X_i, which has state space $\Omega_{X_i} \doteq \{x_{i1}, x_{i2}, \ldots, x_{ir_i}\}$, where $r_i \doteq |\Omega_{X_i}| \geq 2$ is the number of (finite) categories of X_i ($|\cdot|$ is the cardinality of a set or vector). Bold letters are used to emphasize sets or vectors. For example, $\mathbf{x} \in \Omega_\mathbf{X} \doteq \times_{X \in \mathbf{X}} \Omega_X$, for $\mathbf{X} \subseteq \mathcal{X}$, is an instantiation for all the variables in $\mathbf{X}$. $r_{\Pi_i} \doteq |\Omega_{\Pi_i}| = \prod_{X_t \in \Pi_i} r_t$ is the number of possible instantiations of the parent set Π_i of X_i, and $\boldsymbol{\theta} = (\theta_{ijk})_{\forall ijk}$ is the entire vector of parameters with elements $\theta_{ijk} = p(x_{ik}|\boldsymbol{\pi}_{ij})$, for $i \in \{1, \ldots, n\}$, $j \in \{1, ..., r_{\Pi_i}\}$, $k \in \{1, ..., r_i\}$, and $\boldsymbol{\pi}_{ij} \in \Omega_{\Pi_i}$. Because of the Markov condition, the Bayesian network represents a joint probability distribution by the expression $p(\mathbf{x}) = p(x_1, \ldots, x_n) = \prod_i p(x_i|\boldsymbol{\pi}_{ij})$, for every $\mathbf{x} \in \Omega_\mathcal{X}$, where every x_i and $\boldsymbol{\pi}_{ij}$ are consistent with $\mathbf{x}$.

Given a complete data set $D \doteq \{D_1, \ldots, D_N\}$ with N instances, where $D_u \doteq \mathbf{x}_u \in \Omega_\mathcal{X}$ is an instantiation of all the variables, the goal of structure learning is to find a DAG $\mathcal{G}$ that maximizes a given score function, that is, we look for $\mathcal{G}^* \doteq \operatorname{argmax}_{\mathcal{G} \in \boldsymbol{\mathcal{G}}} s_D(\mathcal{G})$, with $\boldsymbol{\mathcal{G}}$ the set of all DAGs with nodes $\mathcal{X}$, for a given score function s_D (the dependency on data is indicated by the subscript D).[1] In this paper, we consider the Bayesian Dirichlet equivalent uniform (BDeu) [11, 12, 1]. The BDeu score idea is to compute a function based on the posterior probability of the structure $p(\mathcal{G}|D)$. In this work we use $p(D|\mathcal{G})$, which is equivalent to the former (in the sense of yielding the same optimal graphs) if one assumes $p(\mathcal{G})$ to be uniform over DAGs:

$$s_D(\mathcal{G}) \doteq \log p(D|\mathcal{G}) = \log \int p(D|\mathcal{G}, \boldsymbol{\theta}) \cdot p(\boldsymbol{\theta}|\mathcal{G}) d\boldsymbol{\theta} \ ,$$

where the logarithm is used to simplify computations, $p(\boldsymbol{\theta}|\mathcal{G})$ is the prior of $\boldsymbol{\theta}$ for a given graph $\mathcal{G}$, assumed to be a Dirichlet with parameters $\boldsymbol{\alpha} \doteq (\boldsymbol{\alpha}_i)_{\forall i}$ with $\boldsymbol{\alpha}_i \doteq (\alpha_{ijk})_{\forall jk}$ (which are assumed to be strictly positive and whose dependence on $\mathcal{G}$, or more specifically on Π_i, is omitted unless necessary in the context):

$$p(\boldsymbol{\theta}|\mathcal{G}) = \prod_{i=1}^{n} \prod_{j=1}^{r_{\Pi_i}} \Gamma(\sum_k \alpha_{ijk}) \prod_{k=1}^{r_i} \frac{\theta_{ijk}^{\alpha_{ijk}-1}}{\Gamma(\alpha_{ijk})} \ .$$

[1] In case of many optimal DAGs, then we assume to have no preference and argmax returns one of them.

From now on, we denote the Dirichlet prior by its defining parameter $\boldsymbol{\alpha}$. Under these assumptions, it has been shown [12] that

$$s_D(\mathcal{G}) = \log \prod_{i=1}^{n} \prod_{j=1}^{r_{\Pi_i}} \frac{\Gamma(\sum_k \alpha_{ijk})}{\Gamma(\sum_k (\alpha_{ijk} + n_{ijk}))} \prod_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + n_{ijk})}{\Gamma(\alpha_{ijk})} \ , \qquad (1)$$

where n_{ijk} indicates how many elements of D contain both x_{ik} and $\boldsymbol{\pi}_{ij}$ (the dependence of n_{ijk} on Π_i is omitted too). The BDe score assumes the prior $\boldsymbol{\alpha}$ to be such that $\alpha_{ijk} \doteq \alpha^* \cdot p(\theta_{ijk}|\mathcal{G})$, where α^* is the parameter known as the Equivalent Sample Size (or the prior strength), and $p(\theta_{ijk}|\mathcal{G})$ is the prior probability for $(x_{ik} \wedge \boldsymbol{\pi}_{ij})$ given $\mathcal{G}$ (or simply given Π_i). The BDeu score assumes further that $p(\theta_{ijk}|\mathcal{G})$ is uniform and thus $\alpha_{ijk} \doteq \frac{\alpha^*}{r_{\Pi_i} r_i}$ and α^* becomes the only free parameter.

An important property of BDeu is that its function is decomposable and can be written in terms of the local nodes of the graph, that is, $s_D(\mathcal{G}) = \sum_{i=1}^{n} s_i(\Pi_i)$ (the subscript D is omitted from now on), such that

$$s_i(\Pi_i) = \sum_{j=1}^{r_{\Pi_i}} \left(\log \frac{\Gamma(\sum_k \alpha_{ijk})}{\Gamma(\sum_k (\alpha_{ijk} + n_{ijk}))} + \sum_{k=1}^{r_i} \log \frac{\Gamma(\alpha_{ijk} + n_{ijk})}{\Gamma(\alpha_{ijk})} \right) \ . \qquad (2)$$

3 Min-BDeu and Max-BDeu Scores

In order to study the sensitivity of the BDeu score to different choices of prior $\boldsymbol{\alpha}$, we define an ε-contaminated set of priors. Let β denote $\frac{\alpha^*}{r_{\Pi_i}}$, $\mathbf{1}$ denote the vector $[1, \ldots, 1]$ with length r_i, and $\mathcal{S}$ denote the set of the r_i distinct degenerate mass functions of dimension r_i. Then

$$\begin{aligned} \forall ij: \ A^{\varepsilon}_{ij} &\doteq \mathrm{CH} \left\{ \beta \left(\frac{(1-\varepsilon)}{r_i} \mathbf{1} + \varepsilon \mathbf{v} \right) \ | \ \mathbf{v} \in \mathcal{S} \right\} \\ &= \left\{ \forall k: \ \alpha_{ijk} \in \left[\beta \frac{(1-\varepsilon)}{r_i}, \beta \left(\varepsilon + \frac{(1-\varepsilon)}{r_i} \right) \right], \ \sum_{k=1}^{r_i} \alpha_{ijk} = \beta \right\} \ , \qquad (3) \end{aligned}$$

where CH means the convex hull operator. Equation (3) defines a set of priors for each i, j by allowing the Dirichlet parameters to vary "around" the symmetric Dirichlet with sum of parameters β. To accommodate possible different choices of priors, we rewrite the score function for each node to take into account the value of $\boldsymbol{\alpha}$:

$$\begin{aligned} s_i(\Pi_i, \boldsymbol{\alpha}_i) &= \sum_{j=1}^{r_{\Pi_i}} s_{i,j}(\Pi_i, \boldsymbol{\alpha}_{ij}), \\ s_{i,j}(\Pi_i, \boldsymbol{\alpha}_{ij}) &= \log \frac{\Gamma(\sum_k \alpha_{ijk})}{\Gamma(\sum_k (\alpha_{ijk} + n_{ijk}))} + s'_{i,j}(\Pi_i, \boldsymbol{\alpha}_{ij}), \text{ and} \\ s'_{i,j}(\Pi_i, \boldsymbol{\alpha}_{ij}) &= \sum_{k=1}^{r_i} \log \frac{\Gamma(\alpha_{ijk} + n_{ijk})}{\Gamma(\alpha_{ijk})} = \sum_{\substack{k \in \{1,\ldots,r_i\}: \\ n_{ijk} \neq 0}} \log \frac{\Gamma(\alpha_{ijk} + n_{ijk})}{\Gamma(\alpha_{ijk})} \ . \qquad (4) \end{aligned}$$

where $\boldsymbol{\alpha}_{ij} \in A^{\varepsilon}_{ij}$ for a given $0 < \varepsilon \leq 1$. Using this extended parametrization of the score, we can define our new score functions Min-BDeu $\underline{s}$ and Max-BDeu $\overline{s}$.

$$\underline{s}(\mathcal{G}) \doteq \min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} s_i(\mathit{\Pi}_i, \boldsymbol{\alpha}_i) \quad \text{and} \quad \overline{s}(\mathcal{G}) \doteq \max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} s_i(\mathit{\Pi}_i, \boldsymbol{\alpha}_i) \ ,$$

where maximization and minimization are taken with respect to the sets A^{ε}_{ij}, for every i, j. The names Max-BDeu and Min-BDeu represent the fact that a maximization (or minimization) of the Bayesian Dirichlet equivalent uniform score over the ε-contamination set is performed. However, we note that Min-BDeu and Max-BDeu scores as defined here do not necessarily respect the *likelihood equivalence* property of BDeu. These scores can be seen as a sensitivity analysis of the structure under different prior distributions [17]. It is possible to maintain likelihood equivalence by enforcing constraints among the Dirichlet parameters, but decomposability would be lost and the score computation would become very expensive [18]. Arguments about varying priors and likelihood equivalence are given in [19].

We devote the final part of this section to demonstrate that Min-BDeu and Max-BDeu can be efficiently computed. The first important thing to note is that the maximization can be performed independently for each i, j (the same holds for the minimization):

$$\overline{s}(\mathcal{G}) = \max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} s_i(\mathit{\Pi}_i, \boldsymbol{\alpha}_i) = \sum_{i=1}^{n} \max_{\boldsymbol{\alpha}_i} s_i(\mathit{\Pi}_i, \boldsymbol{\alpha}_i) = \sum_{i=1}^{n} \sum_{j=1}^{r_{\mathit{\Pi}_i}} \max_{\boldsymbol{\alpha}_{ij} \in A^{\varepsilon}_{ij}} s_{i,j}(\mathit{\Pi}_i, \boldsymbol{\alpha}_{ij}) \ ,$$

where

$$\max_{\boldsymbol{\alpha}_{ij} \in A^{\varepsilon}_{ij}} s_{i,j}(\mathit{\Pi}_i, \boldsymbol{\alpha}_{ij}) = \log \frac{\mathit{\Gamma}(\beta)}{\mathit{\Gamma}(\beta + \sum_k n_{ijk})} + \max_{\boldsymbol{\alpha}_{ij} \in A^{\varepsilon}_{ij}} s'_{i,j}(\mathit{\Pi}_i, \boldsymbol{\alpha}_{ij}) \ .$$

Expanding the Gamma functions of $s'_{i,j}(\mathit{\Pi}_i, \boldsymbol{\alpha}_{ij})$ by repeatedly using the equality $\mathit{\Gamma}(z + 1) = z\mathit{\Gamma}(z)$, we obtain the following convex optimization problem with linear constraints:

$$\max_{(\alpha_{ijk})_{\forall k}} \sum_{\substack{k \in \{1,\dots,r_i\}: \\ n_{ijk} \neq 0}} \sum_{w=0}^{n_{ijk}-1} \log(\alpha_{ijk} + w)$$

$$\text{subject to} \sum_{k=1}^{r_i} \alpha_{ijk} = \beta \text{ and } \forall k: \ \alpha_{ijk} \in \left[\beta \frac{(1-\varepsilon)}{r_i}, \beta\left(\varepsilon + \frac{(1-\varepsilon)}{r_i}\right)\right] \ .$$

Hence, the solution of $\max_{\boldsymbol{\alpha}_i} s_i(\mathit{\Pi}_i, \boldsymbol{\alpha}_i)$ to obtain the local score of parent set $\mathit{\Pi}_i$ of X_i can be done with $r_{\mathit{\Pi}_i}$ calls to a convex programming solver, each of which runs in worst-case time cubic in r_i [20].

The solution for the minimization $\min_{\boldsymbol{\alpha}_{ij} \in A^{\varepsilon}_{ij}} s'_{i,j}(\mathit{\Pi}_i, \boldsymbol{\alpha}_{ij})$ is even simpler: it is enough to find $k_* = \operatorname{argmin} n_{ijk}$ and take as optimal solution the prior with

$$\alpha_{ijk_*} = \beta\left(\varepsilon + \frac{(1-\varepsilon)}{r_i}\right) \quad \text{and} \quad \forall k \neq k_*: \ \alpha_{ijk} = \beta \frac{(1-\varepsilon)}{r_i} \ .$$

In order to compute the score of parent set Π_i, we simply repeat this procedure for every j and compute the associated scores. While much easier to solve, the proof of correctness is slightly more intricate. We do it in three steps. The first step considers the case when at least one $n_{ijk} = 0$. In this case, we can safely choose any k_* such that $n_{ijk_*} = 0$ and the solution value is trivially minimal, because the corresponding term does not appear in the objective function defined by (4) and each function

$$g(\eta, \alpha_{ijk}) = \sum_{w=0}^{\eta-1} \log(\alpha_{ijk} + w)$$

appearing in (4) is monotonically increasing with $\eta > 0$, hence we cannot do better than choosing the minimum possible value (that is, $\beta\frac{(1-\varepsilon)}{r_i}$) for each α_{ijk} associated to non-zero n_{ijk}.

Now we can assume that $n_{ijk} \geq 1$ for every k. The second step of the proof is by contradiction and its goal is to show that only one α_{ijk} will be different from $\beta\frac{(1-\varepsilon)}{r_i}$. So, suppose that the optimal solution is attained at a point $\boldsymbol{\alpha}_{ij}$ such that there are $k_1 \neq k_2$ with $\alpha_{ijk_1} > \frac{(1-\varepsilon)\beta}{r_i}$, $\alpha_{ijk_2} > \frac{(1-\varepsilon)\beta}{r_i}$, $n_{ijk_1} \geq 1$ and $n_{ijk_2} \geq 1$. Let $\mu = \alpha_{ijk_1} + \alpha_{ijk_2}$. Take the terms of the objective function in (4) that correspond to k_1 and k_2:

$$f(\mu, \alpha_{ijk_1}) = \sum_{w=0}^{n_{ijk_1}-1} \log(\alpha_{ijk_1} + w) + \sum_{w=0}^{n_{ijk_2}-1} \log(\mu - \alpha_{ijk_1} + w) \ .$$

While keeping μ constant, we can decrease α_{ijk_1} until $\beta\frac{(1-\varepsilon)}{r_i}$, or increase it until $\alpha_{ijk_2} = \beta\frac{(1-\varepsilon)}{r_i}$. The second derivative of $f(\mu, \alpha_{ijk_1})$ with respect to α_{ijk_1} is

$$-\sum_{w=0}^{n_{ijk_1}-1} \frac{1}{(\alpha_{ijk_1} + w)^2} - \sum_{w=0}^{n_{ijk_2}-1} \frac{1}{(\mu - \alpha_{ijk_1} + w)^2} < 0 \ ,$$

so the function is concave in α_{ijk_1}. Because of that, the minimum is attained at one of its extremes, that is, either at $\alpha_{ijk_1} = \beta\frac{(1-\varepsilon)}{r_i}$ or at $\alpha_{ijk_2} = \beta\frac{(1-\varepsilon)}{r_i}$. If we take such a new solution $\boldsymbol{\alpha}'_{ij}$, it achieves value strictly smaller than that of $\boldsymbol{\alpha}_{ij}$, which is a contradiction.

Hence we can assume that all $k \neq k_*$ must have $\alpha_{ijk} = \beta\frac{(1-\varepsilon)}{r_i}$ and we only need to choose the k_* whose α_{ijk_*} will be $\beta\left(\varepsilon + \frac{(1-\varepsilon)}{r_i}\right)$. The third step of the proof is straightforward: the best choice for k_* in order to minimize $s'_{i,j}(\Pi_i, \boldsymbol{\alpha}_{ij})$ is such that

$$k_* = \operatorname*{argmin}_k g\left(n_{ijk}, \beta\left(\varepsilon + \frac{(1-\varepsilon)}{r_i}\right)\right) \ ,$$

that is, the one of smallest n_{ijk}, simply because the function g is monotonically increasing with η, as described previously, and also with α_{ijk}.

4 Experimental Setup

We begin this section by describing data and settings that are used in the experiments. Our goals are to assess the accuracy of the new proposed scores and to understand the relation between them and the quality of inferred networks. The experimental procedure is performed in steps, as explained in the following.

In order to allow for a comparison against the true model, we generate data from pre-defined Bayesian networks. We experiment both with networks from the literature and with random-generated networks. In the former case, we use the well-known networks named *child* (20 nodes) [21], *insurance* (27 nodes) [22], *water* (32 nodes) [23], and *alarm* (37 nodes) [24]. In the latter case, we employ the *BNGenerator* package v0.3 of [25] to obtain random Bayesian networks. The options passed to the program are the desired number of nodes in the randomly generated network and the maximum degree (sum of number of parents and children) allowed for each node in the graph (to avoid excessively dense graphs, which would require a too large amount of data for learning because of their complexity). The maximum degree (sum of incoming and outgoing arcs) is fixed to six in all the experiments, while the number of nodes varies from 20 to 50 nodes. The number of states is 2 for every node. For each different number of nodes, ten networks are randomly generated.

From each one of these networks, data sets are sampled with ten different sample sizes $N = 10 \cdot 2^i, \forall i \in \{1, \ldots, 10\}$ using the R package *bnlearn* v3.5 [26]. These data sets are then used to compute the BDeu scores for each node, as well as Min-BDeu and Max-BDeu, with equivalent sample size α^* set to one, ε set to one half, and upper limit of three parents per node (because of decomposability, scores are always computed per node and stored in some data structure for later querying). We point out that we have not tuned these values, but instead we chose values that are common in the literature. In spite of that, the results presented in this work remain unaltered under small modifications of α^* around the chosen value (data not shown). We leave for future work a thorough analysis of different values for α^*. Limiting the number of parents per node is a common practice and has the purpose of avoiding a large computational cost of evaluating the scores of a great number of parent sets per node (this number increases exponentially with the limit). We discuss the implications of this decision later on. After the scores are computed and pruned [27], we call the structure learning solver *Gobnilp* v1.4.1 [2, 3] to infer the Bayesian network in an optimal way (that is, we wait until the solver finds the globally optimal structure for the given local scores). With that network, parameter estimation is performed using the same Dirichlet prior parameters and the same data as used for structure learning.

In order to check the accuracy of the inferred networks, from each true network we generate an additional data set with 10000 samples that is not available to the learning procedure. On these data we compute the log-likelihood function using both the true network and the inferred network, and to make the outputs comparable we take the percentage difference between them, which we call *likelihood accuracy*. Lower values of this measure indicate that the inferred network evaluates the log-likelihood of the held-out data in a more similar way to the evaluation done by the true network. The evaluation of the log-likelihood using the inferred network over unseen data sampled from the true network approximates the Kullback-Leibler (KL) divergence and converges to the latter when the amount of data goes to infinity. We refrained from a direct use of KL divergence or other measure of distance between the true distribution (represented by the true network) and the inferred one because of the computational cost of evaluating KL, given that true and inferred networks have (almost always) different structures.

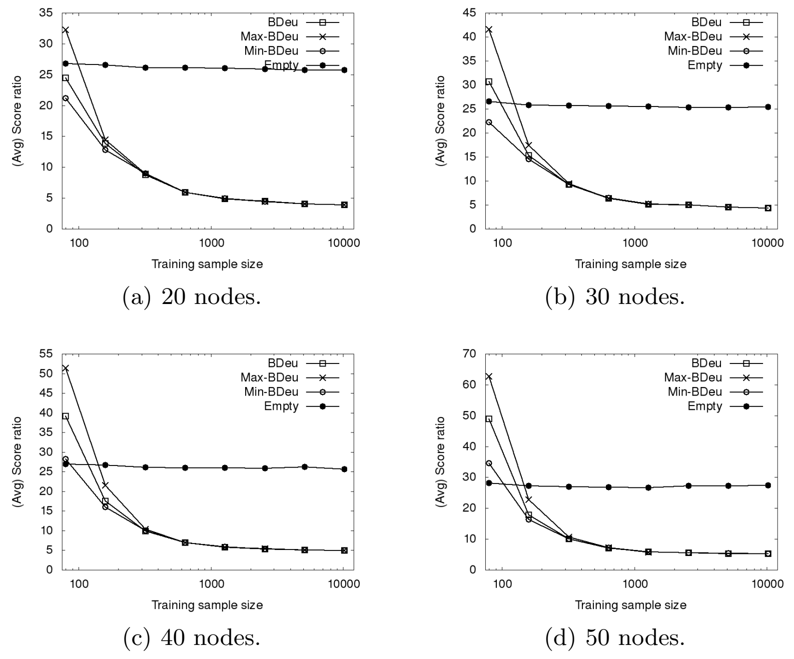
In summary, each execution (i) generates a network (or picks a known one) of a given number of nodes; (ii) generates a data set for accuracy evaluation; (iii) generates a data set for training of a given number of samples; (iv) computes the local scores for each variable using the training data set, for a given decomposable score function; (v) calls the learning procedure using the just calculated local scores; (vi) evaluates the likelihood accuracy with learned/true networks using the held-out data.

4.1 Comparison among Scores

In this experiment, we compare the likelihood accuracy obtained by our proposed scores and the original BDeu. Figure 1 show the average results of the experiment for random networks ranging from 20 to 50 nodes. Min-BDeu achieves better accuracy with respect to the original BDeu on held-out data for training sample size smaller than 100 to 200 samples, and performs similar to the others for larger sample sizes. On the other hand, Max-BDeu achieves worse accuracy than BDeu. Results are consistent across different true networks and network sizes. We conjecture that Min-BDeu has been able to produce better networks by reducing the amount of fitting to the training data when sample size was small, and by transparently increasing this fitting with the increase of the sample size. For the same reason, we further conjecture Max-BDeu produced worse results than BDeu when given few training data because of its increase in fit. The evaluation for well-known Bayesian networks is presented in Fig. 2. The same overall behavior as with random networks is observed.

We also compare the similarity between learned and true networks. We obtain a measure of dissimilarity by computing the moral graph of both true and learned networks, and by counting the total number of mismatches over all pairs of nodes. Figure 3 shows average results for random networks ranging from 20 to 50 nodes. Max-BDeu achieves a better similarity than others for training sample size smaller than 100 to 200 samples, and performs similarly for larger sample sizes. On the other hand, Min-BDeu achieves worse similarity than BDeu. Results are consistent across different network sizes. To analyze further these findings, we show the average total number of arcs produced by the different scores on the same experiments (Fig. 4). Notably, Max-BDeu yields denser inferred networks, while Min-BDeu prefers sparser networks.

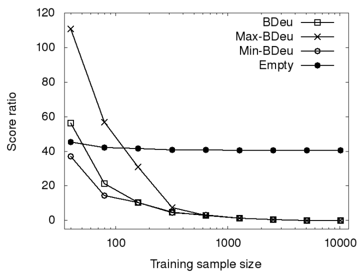
The results obtained so far suggest that Min-BDeu and Max-BDeu aim at different goals. Min-BDeu has better likelihood accuracy, which is achieved by using sparser graphs than original BDeu's. Max-BDeu has better edge accuracy,


(a) 20 nodes. (b) 30 nodes.

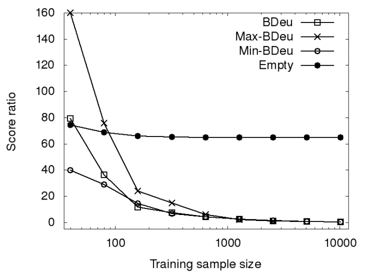
(c) 40 nodes. (d) 50 nodes.

Fig. 1. Average likelihood accuracy among networks learned from different score functions. The graphs feature BDeu (square points), Max-BDeu (cross points), Min-BDeu (circle points), Empty net (bullet points). Each curve of the graph corresponds to the average over random networks used for sampling the data. The points in those curves correspond to the accuracy for different training data sample size.

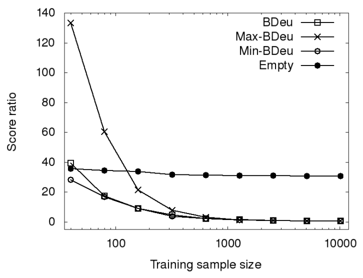
computed as the learned graph similarity to the true one, which is achieved by using denser graphs than BDeu's. These results suggest that with small amount of data, better likelihood accuracy is obtained with networks that are simpler than the true network, probably because the data are not enough to learn good parameters if a denser network were used. This might explain the reason why Max-BDeu has poorer performance in terms of log-likelihood of held-out data. In fact, we further analyzed this situation by learning the parameters of inferred graphs using a large data set of 5000 samples (the graphs themselves were learned with the appropriate varying sample sizes). Figure 5 shows average results for random networks of 20 nodes and different training sample sizes for structure learning, while using 5000 samples for parameter learning. In this scenario, Max-BDeu performed better than the others did even in terms of likelihood accuracy, suggesting that its poorer performance was related to poorer parameter estimation. When training sample size becomes large enough, the behavior of all different methods becomes similar.



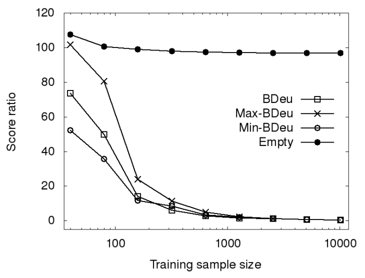
(a) Child network.



(b) Insurance network.



(c) Water network.



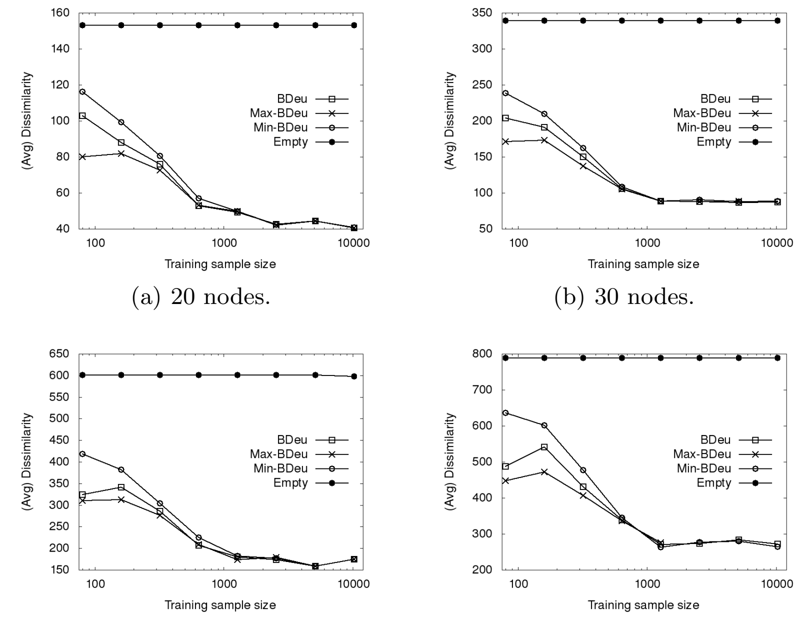
(d) Alarm network.

Fig. 2. Likelihood accuracy among networks learned from different score functions using data generated with well-known Bayesian networks. The graphs feature BDeu (square points), Max-BDeu (cross points), Min-BDeu (circle points), Empty net (bullet points). The points in the curves correspond to the accuracy for different training data sample size.

4.2 Relation between Scores and Learning Saturation

The proposed scores Min-BDeu and Max-BDeu have an interpretation as the pessimistic and optimistic scenarios with respect to unknown "ideal" prior Dirichlet distributions. The set of priors that we employ can be seen as a way to take into account the sensitivity of the score to variations of the local priors. Min-BDeu is aimed at finding the structure that maximizes the BDeu score under the most adversary prior, that is, learning with Min-BDeu is done by a maximin approach. Because of that, the fitting of Min-BDeu is less aggressive than that of BDeu. On the other hand, Max-BDeu is a maximax approach: it yields the structure that maximizes BDeu under the most beneficial prior, so it tends to fit more aggressively than BDeu.

With these considerations in mind, one may define a measure of "sensitivity" for an inferred graph $\mathcal{G}$ as the difference between $\overline{s}(\mathcal{G})$ and $\underline{s}(\mathcal{G})$ (note that these values are logarithms of probabilities, so imprecision here is defined as the ratio between upper and lower probabilities $\log(\overline{p}(D|\mathcal{G})/\underline{p}(D|\mathcal{G}))$). We have performed experiments using only the BDeu score for learning, but whose result is later evaluated in terms of this measure. That is, we execute experiments with only

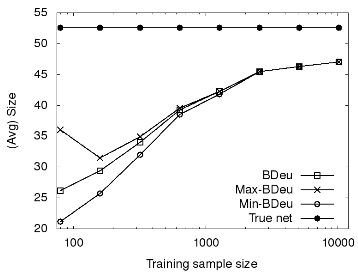


(a) 20 nodes. (b) 30 nodes.

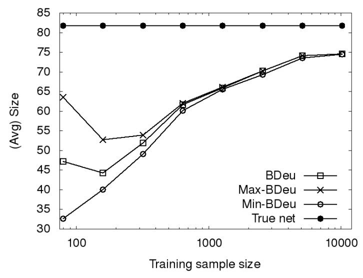
(c) 40 nodes. (d) 50 nodes.

Fig. 3. Comparison on average computed dissimilarity between true and learned moral graphs using different score functions. The graphs feature BDeu (square points), Max-BDeu (cross points), Min-BDeu (circle points), Empty net (bullet points). The points in the curves correspond to the total number of mismatches between the moral graph of true and learned networks.

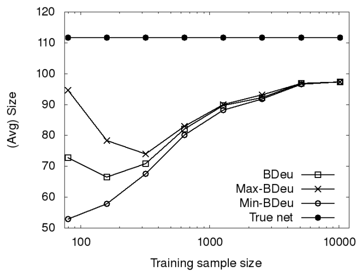
the BDeu score using randomly generated networks to sample training and held-out data. The curves in the upper graph of Fig. 6 show the average likelihood accuracy (over ten runs) obtained by the original BDeu for domains with different number of nodes, as done before. Such accuracy can only be computed because (for testing) we have available the true networks, thus we can generate samples from it to create the held-out test data. Such curves decrease with the amount of data used for learning, as expected. On the other hand, the same Fig. 6 (lower graph) displays the average sensitivity measure (score ratio) of the optimal networks that have been found by BDeu (because of scale differences, we divided them by the lower score so as to have them displayed in the same graph), which does not depend on knowing the true network and thus can be computed in the moment of learning. These curves of the lower graph of Fig. 6 also decrease as the amount of data increases, an expected phenomenon as the importance of the prior reduces with the increase in sample size. We can see in the figure a relation between likelihood accuracy and the sensitivity measure. This suggests that one can use the measure to determine (approximately) the amount of data after which structure learning will not benefit anymore (or will benefit very little) from additional data. In these tests, this saturation point happens around one



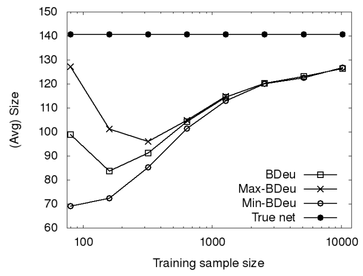
(a) 20 nodes.



(b) 30 nodes.



(c) 40 nodes.



(d) 50 nodes.

Fig. 4. Comparison on average total number of arcs in the learned networks from different score functions. The graphs feature BDeu (square points), Max-BDeu (cross points), Min-BDeu (circle points), true network (bullet points). The points in the curves correspond to the total number of arcs in the graphs of learned networks.

thousand samples. Previous to the saturation point, the likelihood accuracy of the inferred networks greatly increases from the provision of more training data. Past this saturation point, this increase becomes small and might not be cost-effective in many situations. We note that the score alone could not be used for this task, as it does not usually converge with the increase in the amount of data (recall that we use the logarithm of the probability of data given the structure).

One might have noticed that the accuracy curves in Fig. 6 do not converge to zero with the increase in the amount of data. This can be explained as an effect of the restricted maximum number of parents per node during learning (defined as three in our simulations). Figure 7 shows the likelihood accuracy of original BDeu for varying limits on the number of parents (from 1 to 6) for learned networks from data of randomly generated Bayesian networks with 20 nodes (true networks have no such limit in the number of parents, but have a total degree limit of six). It is also very interesting to note the shape of the accuracy curves. With small training sample size, it is seems better to force stronger limits, as the inferred networks with greater number of parents might be unreliable (recall that the number of training data and number of parents per node are very related in terms of learning accuracy). This indicates, at least in

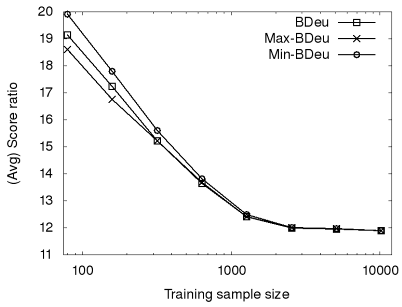


Fig. 5. Average likelihood accuracy among networks learned from different score functions. The graphs feature BDeu (square points), Max-BDeu (cross points), Min-BDeu (circle points) for networks with 20 nodes. The structure of the networks is learned with the given training data sample size, while the network parameters are learned with a training data set of 5000 samples.

these experiments, that the original BDeu score was not able to control well the complexity of the learned network with small sample sizes, given that the limit on the number of parents was able to improve learning accuracy. This empirical result corroborates with our previous results about Min-BDeu outperforming the original BDeu when sample size is small.

5 Conclusions

In this paper we presented two new score functions for learning the structure of Bayesian networks from data. They are based on allowing the Dirichlet prior parameters of the Bayesian Dirichlet equivalent uniform (BDeu) to vary inside a contamination set around the symmetric Dirichlet priors, while keeping its strength fixed. Over this set of priors we choose the most adversary and the most beneficial priors to construct the Min-BDeu and the Max-BDeu, respectively. Learning with Min-BDeu is equivalent to a maximin approach, as one must find the graph that maximizes the minimum score (over BDeu scores with priors in the contaminated set). Min-BDeu prefers sparser graphs than the original BDeu does when sample size is small, and converges to the original BDeu as the amount of data increases. Max-BDeu is analogous, but using the most beneficial prior, that is, a maximax approach.

We demonstrate that these new score functions can be efficiently computed and are decomposable just like the original BDeu, so they can be used within most of the current state-of-the-art structure learning solvers. In our experiments, Min-BDeu has led to networks with higher accuracy than that of the original BDeu score, in terms of fitting the true model. On the other hand, Max-BDeu has led to better edge accuracy, that is, has fit the graph structure better. We also employ a combination of Max-BDeu and Min-BDeu as a measure of sensitivity. This measure visually correlates with the accuracy of inferred networks

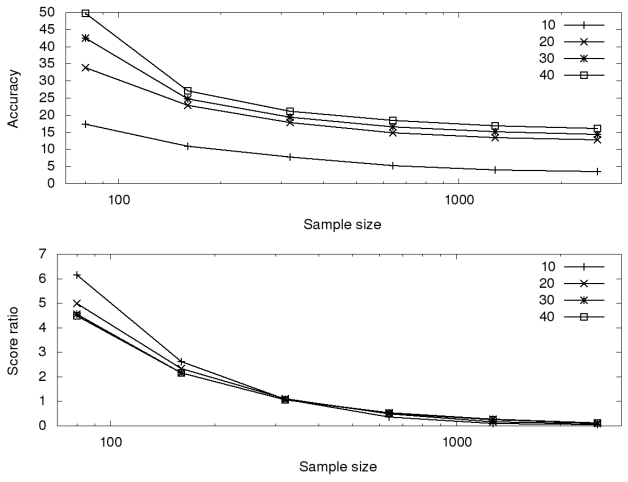


Fig. 6. Curves for likelihood accuracy (upper graph) and sensitivity measure as ratio of Max- and Min-BDeu (lower graph) for inferred networks learned from training data of different sizes using randomly generated Bayesian networks with 10 to 40 nodes.

and might be used to identify the amount of data that saturates the learning, that is, the amount of data after which the accuracy of the inferred network does not considerably improve anymore if additional data were made available. Finally, an analysis of the BDeu accuracy with respect to the restriction on the maximum number of parents per node helps to explain the better accuracy of Min-BDeu against the original BDeu score when the training sample size is small. Scenarios of small sample size are particularly important in applied fields such as biomedicine and bioinformatics.

As future work we intend to expand our experiments, including the study of the sensitivity of the parameters that define the size of the contamination and the strength of the prior, as well as a deeper analysis about the consequences of limiting the number of parents of each variable, in order to better understand the properties of Min-BDeu and Max-BDeu. We also want to further study the characteristics of the original BDeu and to investigate other score functions, for instance using entropy as criterion to select priors.

Acknowledgments

Work partially supported by the Swiss NSF grants Nos. 200021_146606 / 1 and 200020_137680 / 1, and by the Swiss CTI project with Hoosh Technology.

References

1. Heckerman, D., Geiger, D., Chickering, D.M.: Learning Bayesian networks: the combination of knowledge and statistical data. Machine Learning **20** (1995) 197–

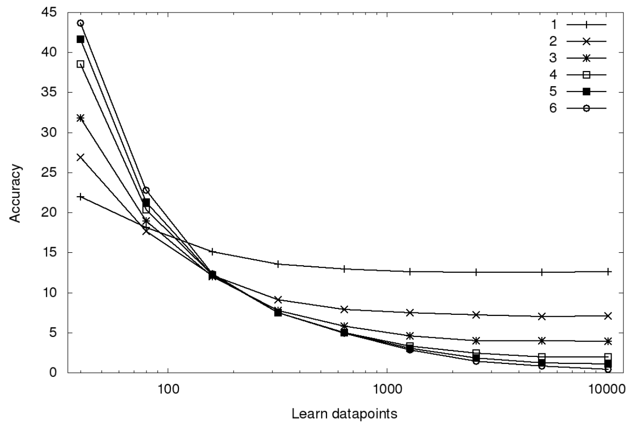


Fig. 7. Average likelihood accuracy curves of learned networks from data of randomly generated Bayesian networks with 20 nodes and maximum degree of six. Different curves represent the result of inferred networks under different parent set size limits during learning: maximum of one parent up to maximum of six parents.

2. Barlett, M., Cussens, J.: Advances in Bayesian network learning using integer programming. In: Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence. UAI'13 (2013) 182–191
3. Cussens, J.: Bayesian network learning with cutting planes. In: Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence. UAI'11, Barcelona, AUAI Press (2011) 153–160
4. de Campos, C.P., Ji, Q.: Efficient structure learning of Bayesian networks using constraints. Journal of Machine Learning Research **12** (2011) 663–689
5. de Campos, C.P., Zeng, Z., Ji, Q.: Structure learning of Bayesian networks using constraints. In: Proceedings of the 26th International Conference on Machine Learning. ICML'09, Montreal, Omnipress (2009) 113–120
6. Jaakkola, T., Sontag, D., Globerson, A., Meila, M.: Learning Bayesian Network Structure using LP Relaxations. In: Proceedings of the 13th International Conference on Artificial Intelligence and Statistics. AISTATS'10 (2010) 358–365
7. Niinimäki, T., Koivisto, M.: Annealed importance sampling for structure learning in Bayesian networks. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence. IJCAI'13, AAAI Press (2013) 1579–1585
8. Parviainen, P., Koivisto, M.: Exact structure discovery in Bayesian networks with less space. In: Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence. UAI'09, AUAI Press (2009) 436–443
9. Parviainen, P., Koivisto, M.: Finding optimal Bayesian networks using precedence constraints. Journal of Machine Learning Research **14** (2013) 1387–1415
10. Yuan, C., Malone, B.: Learning optimal Bayesian networks: A shortest path perspective. Journal of Artificial Intelligence Research **48** (2013) 23–65
11. Buntine, W.: Theory refinement on Bayesian networks. In: Proceedings of the 8th Conference on Uncertainty in Artificial Intelligence. UAI'92, San Francisco, CA, Morgan Kaufmann (1991) 52–60

12. Cooper, G.F., Herskovits, E.: A Bayesian method for the induction of probabilistic networks from data. Machine Learning **9** (1992) 309–347
13. Schwarz, G.: Estimating the dimension of a model. The Annals of Statistics **6**(2) (1978) 461–464
14. Akaike, H.: A new look at the statistical model identification. IEEE Transactions on Automatic Control **19**(6) (1974) 716–723
15. Brenner, E., Sontag, D.: Sparsityboost: A new scoring function for learning Bayesian network structure. In: Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence. UAI'13, Corvallis, Oregon, AUAI Press (2013) 112–121
16. Walley, P.: Statistical Reasoning with Imprecise Probabilities. Chapman and Hall, London (1991)
17. Silander, T., Kontkanen, P., Myllymäki, P.: On Sensitivity of the MAP Bayesian Network Structure to the Equivalent Sample Size Parameter. In: Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence. UAI'07 (2007) 360–367
18. Abellan, J., Moral, S.: New score for independence based on the imprecise Dirichlet model. In: International Symposium on Imprecise Probability: Theory and Applications. ISIPTA'05, SIPTA (2005) 1–10
19. Cano, A., Gómez-Olmedo, M., Masegosa, A.R., Moral, S.: Locally averaged Bayesian Dirichlet metrics for learning the structure and the parameters of Bayesian networks. International Journal of Approximate Reasoning **54**(4) (2013) 526–540
20. Ben-Tal, A., Nemirovski, A.: Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications. MPS/SIAM Series on Optimization. SIAM (2001)
21. Spiegelhalter, D.J., Cowell, R.G. In: Learning in probabilistic expert systems. Clarendon Press, Oxford (1992) 447–466
22. Binder, J., Koller, D., Russell, S., Kanazawa, K.: Adaptive probabilistic networks with hidden variables. Machine Learning **29** (1997)
23. Jensen, F.V., Kjærulff, U., Olesen, K.G., Pedersen, J.: Et forprojekt til et ekspertsystem for drift af spildevandsrensning (an expert system for control of waste water treatment — a pilot project). Technical report, Judex Datasystemer A/S, Aalborg, Denmark (1989) In Danish.
24. Beinlich, I.A., Suermondt, H.J., Chavez, R.M., Cooper, G.F.: The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks. In: Proceedings of the 2nd European Conference on Artificial Intelligence in Medicine. Volume 38 of Lecture Notes in Medical Informatics., Springer Berlin Heidelberg (1989) 247–256
25. Ide, J.S., Cozman, F.G.: Random generation of Bayesian networks. In: Brazilian Symposium on Artificial Intelligence. SBIA'02, Springer-Verlag (2002) 366–375
26. Nagarajan, R., Scutari, M., Lèbre, S.: Bayesian Networks in R with Applications in Systems Biology. Use R! series. Springer (2013)
27. de Campos, C.P., Ji, Q.: Properties of Bayesian Dirichlet scores to learn Bayesian network structures. In: AAAI Conference on Artificial Intelligence, AAAI Press (2010) 431–436