Impact of Learning Strategies on the Quality of Bayesian Networks: An Empirical Evaluation

Brandon Malone
Max Planck Institute
for the Biology of Ageing

Matti Järvisalo and **Petri Myllymäki**
HIIT, Department of Computer Science,
University of Helsinki

Abstract

We present results from an empirical evaluation of the impact of Bayesian network structure learning strategies on the learned structures. In particular, we investigate how learning algorithms with different optimality guarantees compare in terms of structural aspects and generalisability of the produced network structures. For example, in terms of generalization to unseen testing data, we show that local search algorithms often benefit from a tight constraint on the number of parents of variables in the networks, while exact approaches tend to benefit from looser parent restrictions. Overall, we find that learning strategies with weak optimality guarantees show good performance on synthetic datasets, but, compared to exact approaches, perform poorly on the more "real-world" datasets. The exact approaches, which guarantee to find globally optimal solutions, consistently generalize well to unseen testing data, motivating further work on increasing the robustness and scalability of such algorithmic approaches to Bayesian network structure learning.

1 INTRODUCTION

This work focuses on the well-known problem of learning the structure of a Bayesian network (BN) from data (Heckerman et al., 1995). The BN structure learning problem (BNSL) is to find a BN structure which optimizes a decomposable scoring function—typically, either a Bayesian posterior probability such as the Bayesian Dirichlet (BD) score or a penalized likelihood function. BNSL is NP-hard (Chickering, 1996), which poses challenges for developing algorithmic solutions for this important problem.

Methods proposed for solving BNSL on real-world datasets divide into approximate, local search methods, e.g., (Heckerman et al., 1995; Chickering, 2002; Teyssier and Koller, 2005; Tsamardinos et al., 2006), which do not offer quality guarantees (in terms of how well a given decomposable scoring function is optimized), and exact algorithms (Ott and Miyano, 2003; Koivisto and Sood, 2004; Silander and Myllymäki, 2006; Parviainen and Koivisto, 2009; de Campos and Ji, 2011; Yuan and Malone, 2013; Bartlett and Cussens, 2013) which produce guaranteed optimal solutions with respect to the scoring function. Often, the local search methods are computationally efficient, while the exact algorithms can require an exponential amount of time (and in cases, even memory).

The score of a BN structure is ideally a reflection of how well it models a training dataset. The general assumption has been that networks which model the training data well should also accurately reflect new data. However, it is well-known that a model can describe a training set very well, yet generalize poorly to new data (Mitchell, 1997). Thus, there is no guarantee that a network which optimizes a score for a training set will generalize well to new data.

There is currently no clear empirical evidence (for or against) on whether the increased computational efforts required by exact approaches to BNSL are justifiable in terms of performance of learned BNs on unseen testing data. All in all, the relationship between the chosen learning algorithm and the quality of the learned BN structures in terms of generalisability is not well understood. Furthermore, the choice of the learning algorithm used can affect structural properties of the learned networks in other ways. For example, for many of the exact algorithmic solutions to be applicable, in practice structural restrictions must be placed on the classes of BN structures considered during search. A commonly-applied restriction is a hard constraint on the number of parents allowed for each vertex in the network structures (Friedman et al., 1999). Since most scores incorporate a complexity penalty as a "soft constraint" favoring sparser networks, this is a practically-motivated restriction, as computing the scoring function for an arbitrary number of parents is in general infeasible (though pruning rules (de Campos and Ji, 2011) may in cases permit computing all necessary scores for datasets with few records).

However, the influence of such choices combined with the choice of the learning algorithm used—we refer to the combination of the two as a *learning strategy*—has not received much attention.

The aim of this work is to establish a more in-depth understanding of the aforementioned, currently not well-understood aspects of BNSL via an extensive empirical evaluation. Our aim is to shed light on the relationship of different learning strategies, based on four popular score-based structure learning algorithms, and the unknown discrepancy between training set scores and generalization. In particular, we address the following research questions for different fixed learning algorithms and training sets.

Q1 How similar are structures found using different learning strategies?

Q2 How do hard constraints on the number of parents in learned structures affect their generalization?

Q3 How does the amount of training data affect the generalization of learned structures?

Q4 Which learning strategies result in networks with the best generalization?

Our main contributions, based on a rigorous experimental setup, are the following. For Q1, we show that the different learning strategies tend to learn dissimilar network structures. With respect to Q2, we show that for small datasets, hard constraints limiting the maximum number of parents to 2 improves generalization on a few datasets for local search algorithms; however, optimal algorithms usually benefit from a higher limit. We answer Q3 by using increasingly large subsets of available training data. Regardless of the algorithms' guarantees, more training data results in more accurate predictions on testing data. Finally, we address Q4 by considering all of the data collected during the evaluation. For some datasets, simple strategies such as the tractable Chow-Liu algorithm can provide good generalization. However, the simple strategies fail to generalize well on other datasets. Predictive likelihood results show that the optimal algorithms consistently generalize well.

While some of the observations made in this work, based on concrete empirical data, may appear to the reader as common knowledge, we note that to our best knowledge this is the first work which studies all of the question listed above thoroughly via empirical means. A previous related study is (Acid et al., 2004), which focuses on a case study of the choice of BNSL learning strategies for data from an emergency medical service. That study focused on Q1 for that particular domain and, orthogonally to the present work, the impact of the choice of scoring functions on the learning results. Furthermore, focusing on causal Bayesian networks, an evaluation of structural distance measures was presented in (de Jongh and Druzdzel, 2009). (Ueno, 2010; Ueno, 2011) studied the effect of the equivalent sample size of BD on learned structures but did not consider predictive likelihood.

2 BACKGROUND

A Bayesian network (BN) (Pearl, 1988) is a compact representation of a joint probability distribution over random variables $\mathbf{X} = \{X_1, \ldots, X_n\}$. It consists of a directed acyclic graph (DAG) G, in which each vertex in the graph corresponds to one of the random variables, and conditional probability distributions $P(X_i|PA_i)$, where PA_i is the set of parents of X_i in G. The (log) joint probability distribution over all of the variables is

$$\log P(X_1, \ldots, X_n|G) = \sum_i^n \log P(X_i|PA_i).$$

Given a BN $\mathcal{N}$ and a dataset $d = \{d_1, \ldots, d_N\}$, where each d_r (*record*) is independent of the others, and a complete instantiation of $\mathbf{X}$, the likelihood of d given $\mathcal{N}$ is

$$\log P(d|\mathcal{N}) = \sum_r^N \log P(d_r|\mathcal{N}) = \sum_r^N \sum_i^n \log P(X_i^r|PA_i^r), \tag{1}$$

where X_i^r and PA_i^r are the instantiations of X_i and its parents in record d_r, respectively. In total, Equation 1 consists of $N \cdot n$ terms, each of which corresponds to the likelihood of one variable in one record given $\mathcal{N}$. We will refer to this as the *predictive likelihood* of $\mathcal{N}$ on d and use the notation ℓ^d when $\mathcal{N}$ is clear from context.

Given a training dataset d_t and a scoring function s, the *Bayesian network structure learning problem* (BNSL) is to find a BN $\mathcal{N}^* \in \arg\max_{\mathcal{N}} s(d_t, \mathcal{N})$, i.e., a Bayesian network structure with the best score in terms of the scoring function s. The scoring function s is usually a Bayesian posterior probability or penalized likelihood function which measures how well $\mathcal{N}$ "fits" d_t. In practice, s is *decomposable* (Heckerman et al., 1995), i.e., $s(d_t, \mathcal{N}) = \sum_i s(d_t, X_i, PA_i)$. The $s(d_t, X_i, PA_i)$ terms are often called *local scores*.

3 LEARNING ALGORITHMS

Our primary goal is to compare the impact of the guarantees of structure learning algorithms to the generalization of learned networks to unseen testing data. For this, we use four popular score-based learning algorithms with a range of optimality guarantees. In the following discussion, optimality guarantees refer to behavior with respect to the scoring function and a training dataset. In particular, optimality *does not* refer to behavior on unseen testing data.

Hill climbing with a tabu list and random restarts (*tabu*). Hill climbing is a widely-used local search technique in discrete optimization (Russell and Norvig, 2003)

that typically finds local optima for an objective function f by maintaining a *current* solution and applying *search operators*. At each step, all search operators are tentatively applied to the current solution to find its *neighborhood*. The member of the neighborhood which results in the biggest improvement to f is selected as the new current solution. This process is repeated until a local optimum is found, that is, when the current solution is better than everything in its neighborhood. *Random restarting* is a strategy to escape from a local optimum by randomly changing a locally optimal solution and restarting the search from the new random solution. The *tabu list* strategy (Glover, 1990) augments random restarts by keeping track of recently visited solutions; solutions in the tabu list are ignored when considering new neighborhoods. Even with random restarts and the tabu list, the algorithm is unable to provide guarantees on how close the found local optima are to the globally optimal solutions in terms of their scores.

In the context of Bayesian networks, each solution corresponds to a network; the search operators considered here are edge addition, deletion and reversal (as long as the resulting structure is a DAG). The objective function f is exactly the scoring function s.

Max-min hill climbing (*mmhc*). Max-min hill climbing (Tsamardinos et al., 2006) is a two-phase hybrid learning algorithm. During the first phase, it uses a set of statistical independence tests to identify arcs that are forbidden from appearing in the learned network. The second phase uses *tabu* to find local optima within this restricted space. Here we use a mutual information statistical test during the first phase. The first phase of *mmhc* is similar to constraint-based methods such as *pc* (Spirtes et al., 2000). Empirically, *mmhc* has been shown to outperform several other state-of-the-art algorithms, including PC, sparse candidate, three phase dependency analysis, optimal reinsertion and greedy equivalence search (Tsamardinos et al., 2006). While *mmhc* does guarantee to recover BN structures when the data are faithful to a DAG in the large sample limit (Tsamardinos et al., 2006), it does not offer any non-trivial guarantees about the generalization quality of the learned network for unfaithful, finite datasets.

Chow-Liu (*cl*). The Chow-Liu algorithm (Chow and Liu, 1968) is an exact, polynomial-time algorithm for finding an optimal tree-structured BN. The algorithm calculates the mutual information between all pairs of variables to form a weighted graph. The maximum spanning tree through the graph corresponds to the optimal tree-structured BN.

Provably optimal (*opt*). Several algorithms have been developed which are guaranteed to find a network which optimizes s (Ott and Miyano, 2003; Koivisto and Sood, 2004; Silander and Myllymäki, 2006; de Campos and Ji, 2011; Parviainen and Koivisto, 2009; Cussens, 2011; Yuan and Malone, 2013). In practice, these algorithms take as input a set of local scores for each variable and find an optimal network with respect to these scores. In this work, we use two of these algorithms which have previously (Malone et al., 2014) been shown to perform well on a variety of datasets. The first (Yuan and Malone, 2013) is based on casting BNSL as a shortest-path finding problem; it then uses A* to solve the shortest path problem, which gives the optimal network for the given local scores. The second algorithm (Bartlett and Cussens, 2013) creates an integer linear program (ILP) based on the local scores. The solution to the ILP corresponds to the optimal network for the local scores. Both A* and ILP are guaranteed to find (equivalent) optimal network structures. In this work, our goal is to understand the impact of the optimality guarantee on generalization. Since we are not interested in the relative performance in terms of running time or memory consumption among the algorithms, we make no distinction between the different optimal algorithms.

4 QUALITY MEASURES

In order to address our research question Q1, we propose a normalized version of the structural Hamming distance to quantify structural similarity. Research questions Q2–Q4 concern generalization; we propose measures based on the predictive likelihood of Equation 1 to evaluate these results.

4.1 Structural Similarity

We evaluate the structural similarity of two networks with structural Hamming distance (SHD) (Tsamardinos et al., 2006). The SHD between two networks is calculated by transforming the two networks into the partially directed acyclic graphs (PDAGs) representing their equivalence classes. The number of edge additions, deletions, reversals, and orientation changes (converting an undirected edge into a directed edge and vice versa) to transform one PDAG into the other is the SHD. The motivation behind SHD lies in equivalence classes of BNs; using different conditional probability distributions, different DAG structures can describe exactly the same set of joint probability distributions (Chickering, 1995); these DAGs belong to the same *equivalence class*. The SHD is 0 between DAGs in the same equivalence class. Thus, in a sense, SHD serves as an imperfect proxy for measuring the distance between the distributions represented by two DAGs. In order to facilitate comparison across datasets with differing numbers of variables, we use a normalized form $\widehat{\text{SHD}}$ of SHD:

$$\widehat{\text{SHD}}_d = \frac{SHD_d}{\binom{n_d}{2}/2}, \tag{2}$$

where SHD_d is the structural Hamming distance between two networks for dataset d, n_d is the number of variables in dataset d and $\binom{n}{2}$ is the binomial coefficient. The normalization constant $\binom{n_d}{2}/2$ is approximately the maximum

number of edges present in a network structure, and hence also in the order of the SHD between two networks learned for dataset d.

4.2 Predictive Likelihood

We use the predictive likelihood to evaluate the generalization capability of the learned networks. In particular, for a dataset d and learning strategy l we calculate the per-prediction-likelihood, $\ell_{pp}^{d,l}$, which is the likelihood of each prediction on the test set:

$$\ell_{pp}^{d,l} = -\frac{\sum_i^{10} \ell_i^{d,l}}{N_d \cdot n_d}, \tag{3}$$

summing over the folds $i = 1..10$, where $\ell_i^{d,l}$ is the predictive likehood according to Equation 1 on the test set for fold i using learning strategy l (see Section 5.1 for cross-validation discussion), N_d is the number of records in the test set, and n_d is the number of variables in the dataset.

The numerator of Equation 3 is the sum over all of the test set predictive likelihoods for learning strategy l and dataset d. As discussed in Section 2, each $\ell_i^{d,l}$ term comprises $\frac{N_d}{10} \cdot n_d$ terms. In total, the sum in the numerator includes $N_d \cdot n_d$ terms, each of which corresponds to the log probability of one variable of one record from the test set. Consequently, the denominator serves as a normalizing constant, and $\ell_{pp}^{d,l}$ is the average log probability of each prediction.

In order to compare learning strategies, we normalize the $\ell_{pp}^{d,l}$ values for each dataset between 0 and 1 to obtain

$$\hat{\ell}_{pp}^{d,l} = 1 - \frac{\ell_{pp}^{d,l} - \min_{l'}\{\ell_{pp}^{d,l'}\}}{\max_{l'}\{\ell_{pp}^{d,l'}\} - \min_{l'}\{\ell_{pp}^{d,l'}\}} \tag{4}$$

where l' ranges over all learning strategies. Note that, after normalization, the learning strategy with the best $\ell_{pp}^{d,l}$ has $\hat{\ell}_{pp}^{d,l} = 0$ while the worst learning strategy has $\hat{\ell}_{pp}^{d,l} = 1$.

It is important to note that $\ell_{pp}^{d,l}$ and $\hat{\ell}_{pp}^{d,l}$ consider all variables equally. In particular, they do not consider a special "class" variable.

5 EXPERIMENTAL SETUP

We continue by describing the experiment setup.

5.1 Datasets

We used datasets from a previous wide-scale empirical study that focused on predicting the efficiency of BNSL algorithms (Malone et al., 2014)[1]. In total, we obtained 29 datasets from two categories used in that study: 16 are "real" datasets from the UCI Machine Learning Repository (*uci*), and further 13 are generated using logic sampling from well-known benchmark networks (*sam*).

Table 1: Basic dataset characteristics

Dataset	Type	n	N
agaricus	*uci*	22	8 123
alarm$_{10\ 000}$	*sam*	37	10 000
alarm$_{1\ 000}$	*sam*	37	1 000
alarm$_{100}$	*sam*	37	100
carpo$_{10\ 000}$	*sam*	60	10 000
carpo$_{1\ 000}$	*sam*	58	1 000
carpo$_{100}$	*sam*	56	100
connect$_{6\ 000}$	*sam*	39	6 000
anneal	*uci*	32	897
credit	*uci*	18	1 000
lymph	*uci*	19	147
tumor	*uci*	18	338
dermatology	*uci*	34	365
flag	*uci*	27	193
hailfinder$_{1\ 000}$	*sam*	56	1 000
hailfinder$_{100}$	*sam*	56	100
votes	*uci*	17	434
hypothyroid	*uci*	22	3 771
insurance$_{10\ 000}$	*sam*	27	10 000
kredit	*uci*	18	1 000
kr-vs-kp	*uci*	37	3 195
letter	*uci*	17	20 000
lung	*uci*	57	31
mildew$_{1\ 000}$	*sam*	35	1 000
mildew$_{100}$	*sam*	35	100
soybean	*uci*	36	306
spect	*uci*	23	186
water$_{100}$	*sam*	26	100
zoo	*uci*	17	100

As Table 1 shows, the number of variables in the datasets ranges from 17 to 60, and the number of records ranges from about 30 to 20 000. We used standard 10-fold cross-validation in order to evaluate the learning strategies. Datasets were randomly split into 10 folds. Unless otherwise noted, all results are averages over all 10 folds. For a few datasets, not all learning strategies completed on all folds. In these cases, the averages were adjusted to properly account for the number of completed folds.

5.2 Learning and Inference

Most of the learning algorithm implementations are publicly available. In all cases, default parameter values were used (excluding number of parents).

Exact algorithms. The previous study (Malone et al., 2014) found that the ILP algorithm usually ran faster than A* for datasets with up to 10 000 local scores. In order to limit the computational requirements of this study, we used this simple decision rule to select either ILP or A* for each dataset. We used a time limit of 2 hours. If the selected algorithm failed, the other one was used.

[1]The datasets are available at `http://bnportfolio.cs.helsinki.fi/`. Please see the original study for data preprocessing.

Parent limit. For all algorithms except cl, we used hard limits of 2 and 8 on the number of parents. This constraint serves two purposes. First, a hard limit on the complexity of the learned network allows evaluation of the learning algorithms in the different search spaces. In particular, the space defined by a parent limit of 2 is a subset of the space defined by a parent limit of 8. Thus, the optimal solution for the at-most-8-parents space is always at least as good as that of the at-most-2-parents space. Second, calculating all local scores for large parent limits is impractical.

Table 2 outlines the seven combinations of learning algorithms and parent limits used in the evaluation.

Scoring function. We selected the commonly-used Bayesian Dirichlet with score equivalence and uniform structure prior (BDeu) scoring function (Heckerman et al., 1995) with an equivalent sample size (ESS) of 1 as the scoring function. Note that, while several previous studies (Silander et al., 2008; Liu et al., 2012; Korucuoglu et al., 2014) have demonstrated that BDeu is sensitive to the ESS, BDeu with an ESS of 1 is a commonly-used score (Acid et al., 2004), which motivates this choice.

Inference. For all learned structures, parameter values were set using a symmetric Dirichlet prior with a concentration parameter of 1 (which is equivalent to Laplacian smoothing). All testing likelihood calculations were performed by multiplying relevant family factors.

6 RESULTS

In this section, we address each of the research questions Q1–Q4. In Section 6.1, we answer Q1 by showing that $\widehat{\text{SHD}}$ is typically high among networks learned using different algorithms, particularly for *uci* datasets. In Section 6.2, we show that, surprisingly, the answer to Q2 depends upon learning algorithm and dataset. We address Q3 in Section 6.3, where we find that increasing training data both improves prediction accuracy and reduces variance among cross-validation folds. In Section 6.4 we demonstrate that somewhat unexpectdly, for *sam* datasets simple learning strategies like *cl* perform well. The *opt* strategy consistently generalizes better for *uci* datasets. These observations suggest fundamental differences in the *uci* and *sam* datasets.

6.1 Structural Similarity

We address Q1 by evaluating the similarity of learned structures using $\widehat{\text{SHD}}$ in two different settings. We first consider the similarity of structures learned using the same algorithm on different cross-validation folds. We then investigate the similarity of structures learned using different algorithms on the same cross-validation fold. As a trivial baseline learning "result," the empty network with no edges, *empty*, is also included. We stress that the goal of Q1 is to evaluate the structural similarity of learned networks to each other; we do not consider e.g. similarity to "gold standard" networks which do not exist for *uci* datasets.

Variation within a learning algorithm. Due to the cross-fold validation strategy, each algorithm results in multiple networks on each dataset. We compared the $\widehat{\text{SHD}}$ among all pairs of networks learned using a single learning strategy. Figure 1(top) shows that, for *sam* datasets, the difference among learned networks was usually small. In contrast, Figure 1(bottom) shows that *uci* datasets result in more varied networks. The variation for *cl* in Figure 1(bottom, right) is much lower than those of either opt_8 (left) or $tabu_2$ (center). This highlights how the reduced search space decreases variability among optimal structures.

Interestingly, for *uci* datasets, opt_8 networks tend to have a slightly higher $\widehat{\text{SHD}}$ than those of $tabu_2$. However, the variance for opt_8 is smaller than that of $tabu_2$. One interpretation is that opt_8 networks are, in terms of $\widehat{\text{SHD}}$, "equally spaced." On the other hand, some pairs of the $tabu_2$ networks are quite similar, while others have more pronounced differences. An explanation for this phenomenon is the greedy search strategy of $tabu_2$. In the beginning stages of the search, $tabu_2$ is likely to select the same dominant edges, regardless of the nuances of the dataset at hand. In contrast, opt_8 selects accurate substructures and optimally combines them, so it may disregard single strong edges in favor of more informative structures. Another explanation is that the search space for $tabu_2$ is more constrained than that of opt_8. As further evidence for the more constrained space leading to more similar networks, both *cl* (Figure 1(bottom, right)) and opt_2 on *uci* datasets (not shown) typically result in smaller $\widehat{\text{SHD}}$ than $tabu_2$.

Variation between learning algorithm. We additionally compared the average $\widehat{\text{SHD}}$ among networks learned using different strategies on the same training set. Figure 2 shows that some of the strategies learn quite similar networks, while for other strategies the networks differ quite a bit. Perhaps unsurprisingly, the same learning algorithm with different parent limits often result in similar networks. The *mmhc* networks are quite similar to *empty*, which suggests that they are very sparse. Many of the datasets have fewer than 1 000 records. So the statistical tests employed at the beginning of *mmhc* are often unable to detect dependencies, and thus many possible edges are discarded before beginning the score-based search.

In general, all of the learning strategies tend to learn similar networks for the *sam* datasets, while the *uci* datasets result in more diverse structures. Despite this, some patterns among the pairs of learning strategies are consistent among both types of datasets. For example, opt_8 and *cl* exhibit the highest average $\widehat{\text{SHD}}$. On the other hand, $tabu_2$ and opt_2 are more similar than most other pairs for the *uci* datasets, but not for *sam* datasets. This again suggests that the more

Table 2: Learning algorithms used in the study

Algorithm	Parent limits	Abbreviation	Availability
tabu	2, 8	$tabu_2$, $tabu_8$	`http://www.bnlearn.com/`
mmhc	2, 8	$mmhc_2$, $mmhc_8$	`http://www.bnlearn.com/`
cl	-	*cl*	custom
opt	2, 8	opt_2, opt_8	`http://urlearning.org` `http://www.cs.york.ac.uk/aig/sw/gobnilp/`

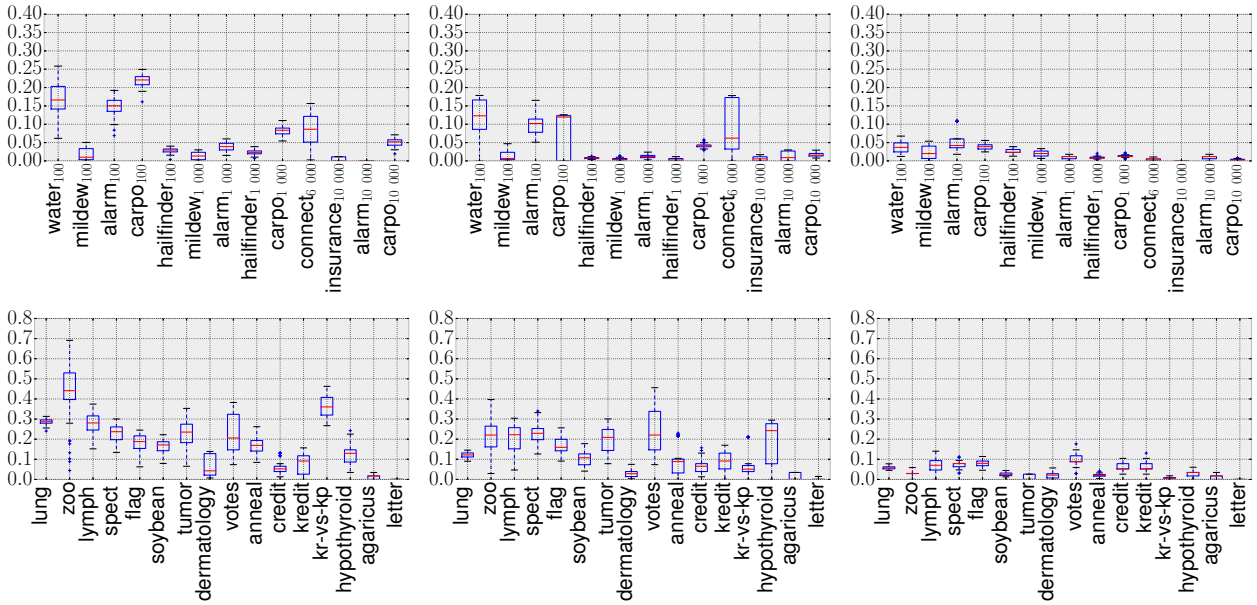

Figure 1: The $\widehat{\text{SHD}}$ values for opt_8 (left), $tabu_2$ (center) and *cl* (right). The top row contains *sam* datasets, and the bottom row contains *uci* datasets. The datasets are sorted in ascending number of records. Note the different scale for *sam* and *uci*.

constrained search space leads to more similar structures.

In summary for Q1, the similarity of learned network structures depends upon the nature of the training dataset. Typically, though, both within and between learning algorithms, *sam* datasets result in similar learned structures, while structures learned from *uci* datasets are more diverse.

6.2 Impact of Restricting Parent Set Size

We study question Q2 by comparing the $\hat{\ell}_{pp}^{d,l}$ among datasets when using $k = 2$ and $k = 8$ as the maximum number of parents for each learning algorithm. The BDeu score implicitly restricts the maximum number of selected parents as a soft constraint by integrating over all parameterizations of parent instantiations. Other scores, such as MDL, explicitly incorporate a complexity penalty to discourage large parent sets. In both cases, though, this restriction is a soft constraint. Here consider the maximum number of parents as a *hard constraint*.

Optimal. Figure 3 (left) shows the performance (in terms of $\hat{\ell}_{pp}^{d,l}$) of generalization using opt_k for parent limits $k = 2, 8$. The (left, top) and (left, bottom) plots show distinctly different patterns. Figure 3 (left, top) clearly shows that opt_2 results in better generalization for *sam* datasets with 100 records. However, as the number of records increases, opt_8 yields better performance. In contrast, for *uci* datasets, opt_8 is almost always better.

Tabu. Contrasting the results for *opt*, Figure 3 (center, bottom) shows that $tabu_2$ generalizes better than $tabu_8$ for *uci* datasets. One possible explanation for this difference is that the greedy strategy of $tabu_8$ favors structures which improve the likelihood while increasing the complexity of the learned structures. Thus, the learned structure overfits the training data and does not generalize well to testing data. In contrast, as *opt* is guaranteed to find the best-scoring structure, it finds structures which better balance training set likelihood and complexity. The hard constraints on the number of parents for $tabu_2$ forbid it from selecting the complex structures. Both $tabu_2$ and $tabu_8$ typically generalize well on *sam* datasets.

MMHC. Figure 3 (right) shows that the hard parent limit has little effect on $\hat{\ell}_{pp}^{d,l}$ for *mmhc*. The first phase of *mmhc* uses a set of statistical independence tests to restrict the learned network structures. For many of the datasets, the

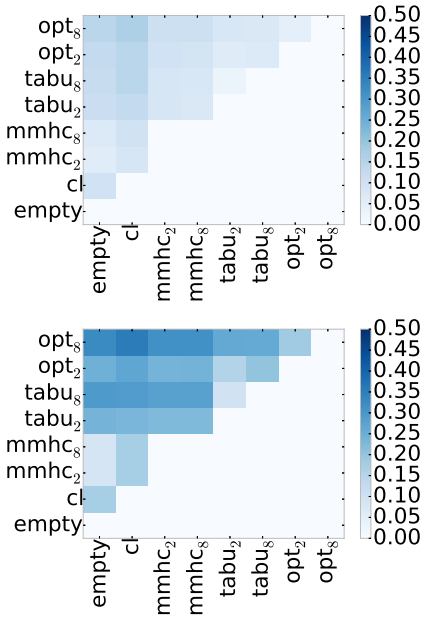

Figure 2: The average $\widehat{\text{SHD}}$ values between networks learned using the same training set among all algorithms for *sam* (top) and *uci* (bottom) datasets. Lighter colors indicate more similar structures. The lower triangle is left empty due to $\widehat{\text{SHD}}$ being symmetric.

relatively small number of records restricts the power of these tests and leads to a very small search space in the second phase, despite initially allowing many more structures for the 8-parent space.

Based on these observations, in the rest of this study we will focus on *opt*$_8$, *tabu*$_2$ and *mmhc*$_8$.

In summary, the answer to Q2 clearly depends both on the training datasets and learning algorithm; the global guarantees of *opt* allow it to fully take advantage of the larger $k = 8$ search space, but the local search strategy of *tabu* performs better in the more restricted $k = 2$ space.

More data is required to accurately estimate the conditional probability distributions for complex structures (with more parameters). This may explain why *opt*$_2$ generalizes better than *opt*$_8$ for datasets with a small number of records.

6.3 Impact of Amount of Training Data

To investigate the impact of the amount of available training data, to answer Q3 we compared how $\ell_{pp}^{d,l}$ of *opt*$_8$, *tabu*$_2$ and *cl* behave as the number of records available for training increases. Figure 4 shows that for all algorithms on both *sam* and *uci* datasets, more records lead to better $\ell_{pp}^{d,l}$. Furthermore, the plots also show that with more records, the variance of $\ell_{pp}^{d,l}$ decreases. Interestingly, the plot also shows that *cl* performs better than *opt*$_8$ and *tabu*$_2$ on *carpo*, a *sam* dataset, when only 100 records are available. This again highlights that restricted model classes can generalize better than those with more parameters, especially when little data is available to estimate the parameter values. Despite the differences in guarantees, *opt*$_8$, *tabu*$_2$ and *cl* perform similarly for *carpo*$_{1\ 000}$ and *carpo*$_{10\ 000}$.

As with *carpo*, for the *uci agaricus* dataset, the likelihood improves and variance decreases as the number of records increases. However, *opt*$_8$ improves from $\ell_{pp}^{d,l} \approx 0.7$ for 81 records to $\ell_{pp}^{d,l} \approx 0.48$ with 812 records. In contrast, *tabu*$_2$ only improves from $\ell_{pp}^{d,l} \approx 0.7$ to about $\ell_{pp}^{d,l} \approx 0.55$, and *cl* exhibits even less improvement. For *agaricus*, *opt*$_8$ using only 812 records results in better generalization than *tabu*$_2$ or *cl* with all 8 123 records.

We observed similar behavior on other *sam* and *uci* datasets as the amount of training data was varied. Unlike in the case of Q1 and Q2, the same general trends hold for all algorithms and datasets with respect to Q3. Namely, the predictive likelihood improves and variance decreases as the size of the training set increases.

6.4 Comparison Across Learning Strategies

Finally, based on the previous results, we studied Q4 by choosing the best learning strategies and comparing their $\hat{\ell}_{pp}^{d,l}$ across all of the datasets. In essence, we fix the training set while varying the learning strategy. Additionally, *empty* (with no edges) was included as a baseline. The results in Figure 5 show several expected trends and a few surprises. As expected, *empty* is the worst on almost all of the datasets. Due to its structural similarity to *empty*, *mmhc*$_8$ was typically worse than the other strategies. These trends are consistent for both *sam* and *uci* datasets. For *sam* datasets, *tabu*$_2$ and *opt*$_8$ have very similar $\hat{\ell}_{pp}^{d,l}$ for most datasets; the $\hat{\ell}_{pp}^{d,l}$ of *cl* is also surprisingly similar to that of the two more "sophisticated" strategies.

For *uci* datasets, *opt*$_8$ continues to consistently have good $\hat{\ell}_{pp}^{d,l}$. On the other hand, *cl* and *tabu*$_2$ exhibit much more inconsistency in their generalization relative to *opt*$_8$. For some datasets, such as *dermatology* and *kredit*, they match *opt*$_8$; on others, such as *credit* and *tumor*,*cl* and *tabu*$_2$ do not generalize well. Surprisingly, *cl* exhibits the best $\hat{\ell}_{pp}^{d,l}$ for *letter*, the *uci* dataset with the most records.

For Q4, *opt* guarantees consistently translate into networks with good generalization. Algorithms with weaker guarantees produce networks with inconsistent generalization.

Comments on Datasets. Besides the behavior of the learning algorithms, these results also suggest differences in the datasets themselves. In particular, it seems that *sam* datasets are "easier," in the sense that many learning strategies find networks which generalize well. On the other hand, only the strategy with strong guarantees consistently generalizes well on *uci* datasets. In some sense, this result is not surprising. The *sam* data is by construction accurately

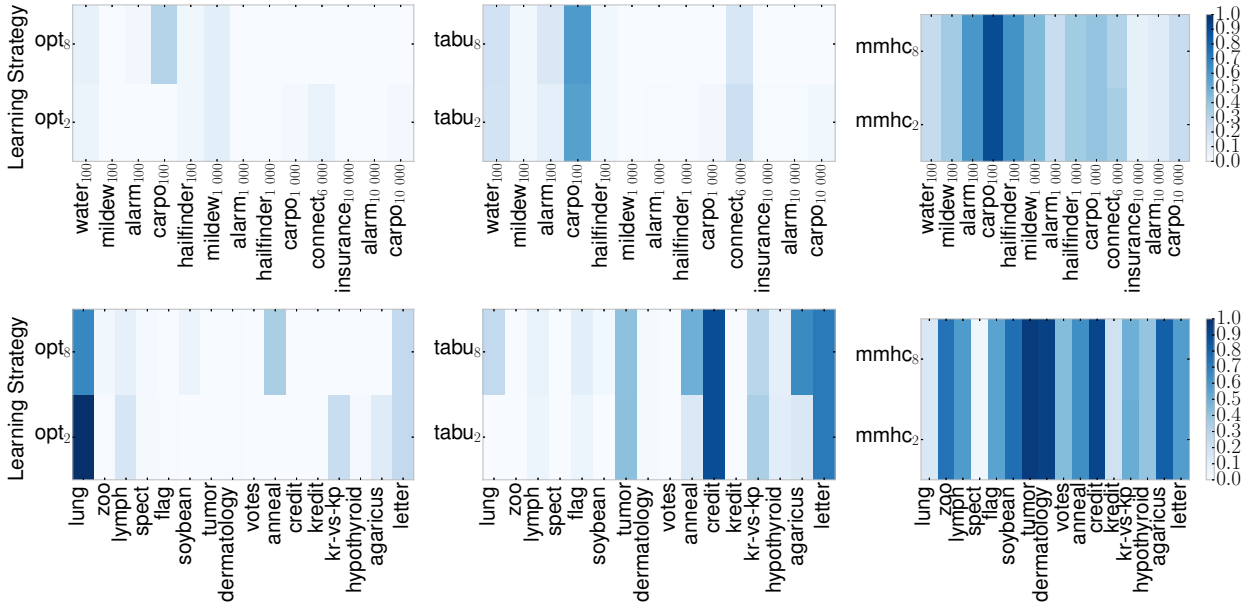

Figure 3: The $\hat{\ell}_{pp}^{d,l}$ values for *opt* (left), *tabu* (center) and *mmhc* (right) with a hard limit of $k = 2$ and $k = 8$ for *sam* (top) and *uci* (bottom) datasets. The datasets are sorted in ascending number of records. Lighter colors indicate better performance. Close inspection of the *mmhc* strategies show some slight difference; however, they are difficult to discern in the scaled image.

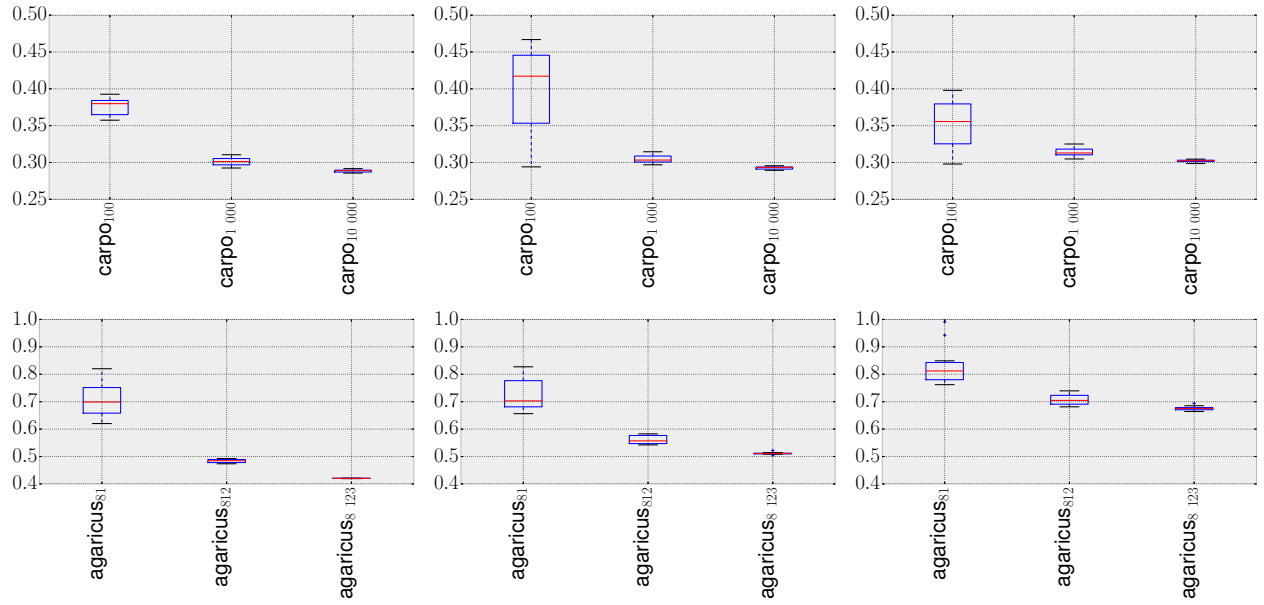

Figure 4: The $\ell_{pp}^{d,l}$ values for using the opt_8 (left), $tabu_2$ (center), and *cl* (right) learning strategies as the number of records increases. The top row is for the *carpo* dataset (*sam*); the bottom row is for the *agaricus* dataset (*uci*). Note the different y-axes for the plots. Lower values and smaller boxes are better.

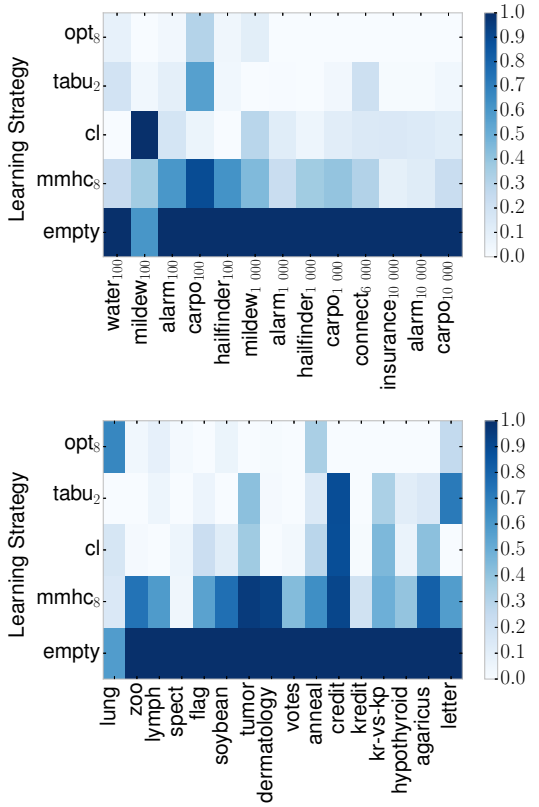

Figure 5: The $\hat{\ell}_{pp}^{d,l}$ values for the best learning strategies. The empty network is included as a baseline. The *sam* datasets are shown in the top heatmap, and *uci* datasets are in the bottom. The datasets are sorted in ascending number of records. Lighter colors indicate better performance.

modeled by a BN, while it is very unlikely that *uci* datasets are faithful to any BN. These caveats are also important for future evaluations.

7 CONCLUSIONS

In this work, we systematically evaluated the impact of learning strategy, including choice of learning algorithm and hard constraints on the number of parents for variables in the network, on the properties of the learned network. Our experiments address the four research questions introduced in Section 1. In particular, we answered Q1 by showing that different learning strategies result in dissimilar structures, particularly for *uci* datasets. For Q2, we showed that for small *sam* datasets, *opt* generalizes better when constrained to 2 parents; however, for all other datasets considered in this study, opt_8 performed better. In contrast, $tabu_2$ almost always outperformed $tabu_8$. Q3 had the clearest answer of the four questions; increasing the size of the training set consistently both improved the predictive likelihood and decreased its variance across cross-validation folds. Finally, with respect to Q4, for the datasets in this evaluation, *opt* consistently results in networks with good generalization. Nevertheless, for some of the *sam* datasets, simpler strategies such as the polynomial-time Chow-Liu algorithm yielded nearly as good generalization. In our view, these results justify the research into learning optimal BN structures in large, complex spaces.

The aim of the study was to better understand how the combination of learning strategy and dataset affect generalization. Consequently, we deliberately disregarded the runtime and memory usage of the studied learning strategies. These are important constraints, especially for *opt*, i.e., exact algorithms which provide provably optimal network structures with respect to the score-based objective function. However, the results clearly show that, whenever possible (as long as the computational resources allow it), it is worthwhile to use *opt*. Similarly, we did not evaluate the SHD between learned network structures and a "gold standard" network because this does not directly reflect generalization. Also, trustworthy "gold standard" networks do not generally exist for real-world datasets.

This empirical study clarifies some common assumptions about relationship between structure learning algorithms and learned Bayesian network structures, including empirically observed conditions for strong theoretical guarantees translating into improved empirical results. The results suggest many interesting questions for further study. For example, the finding that quite simple networks can generalize well suggests that a scoring function with a high complexity penalty, such as the Bayesian Information Criterion (BIC), might yield networks with good predictive capabilities. Another interesting question concerns the impact of more sophisticated restrictions on learned networks structures. For example, recently several algorithms (Korhonen and Parviainen, 2013; Berg et al., 2014; Parviainen et al., 2014) have been proposed which find provably optimal BNs with bounded treewidth, resulting in networks for which exact inference is provably tractable. As bounded treewidth represents another well-principled approach to constraining complexity of the learned networks, an interesting question is whether the quality of the learned (optimal) networks is affected by such a stringent constraint. Other extensions of this work would be to involve yet more structure learning algorithms (such as the Greedy Equivalence Search (Chickering, 2002)), and extending from using single structure for predictive inference to a more Bayesian approach by collecting several high-scoring networks and averaging their predictions. Additionally, the predictive likelihood analysis is not restricted to Bayesian networks; extensions to other generative models, like Markov random fields, would also be of interest.

Acknowledgements This research was supported in part by the Academy of Finland (grants 251170/COIN, 276412, and 284591).

References

Acid, S., de Campos, L. M., Fernandez-Luna, J. M., Rodriguez, S., Rodriguez, J. M., and Salcedo, J. L. (2004). A comparison of learning algorithms for Bayesian networks: a case study based on data from an emergency medical service. *Artificial Intelligence in Medicine*, 30(3):215–232.

Bartlett, M. and Cussens, J. (2013). Advances in Bayesian network learning using integer programming. In *Proc. UAI*, pages 182–191. AUAI Press.

Berg, J., Järvisalo, M., and Malone, B. (2014). Learning optimal bounded treewidth Bayesian networks via maximum satisfiability. In *Proc. AISTATS*, pages 86–95. JMLR.org.

Chickering, D. M. (1995). A transformational characterization of equivalent Bayesian network structures. In *Proc. UAI*, pages 87–98. Morgan Kaufmann.

Chickering, D. M. (1996). Learning Bayesian networks is NP-complete. In *Learning from Data: Artificial Intelligence and Statistics V*, pages 121–130. Springer-Verlag.

Chickering, D. M. (2002). Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research*, 2:445–498.

Chow, C. and Liu, C. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467.

Cussens, J. (2011). Bayesian network learning with cutting planes. In *Proc. UAI*, pages 153–160. AUAI Press.

de Campos, C. P. and Ji, Q. (2011). Efficient learning of Bayesian networks using constraints. *Journal of Machine Learning Research*, 12:663–689.

de Jongh, M. and Druzdzel, M. J. (2009). A comparison of structural distance measures for causal Bayesian network models. In *Recent Advances in Intelligent Information Systems, Challenging Problems of Science, Computer Science series*, pages 443–456. Academic Publishing House EXIT.

Friedman, N., Nachman, I., and Peer, D. (1999). Learning Bayesian network structure from massive datasets: The "sparse candidate" algorithm. In *Proc.UAI*, pages 206–215. Morgan Kaufmann.

Glover, F. (1990). Tabu search: A tutorial. *Interfaces*, 20(4):74–94.

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

Koivisto, M. and Sood, K. (2004). Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5:549–573.

Korhonen, J. H. and Parviainen, P. (2013). Exact learning of bounded tree-width Bayesian networks. In *Proc. AISTATS*, pages 370–378. JMLR.org.

Korucuoglu, M., Isci, S., Ozgur, A., and Otu, H. H. (2014). Bayesian pathway analysis of cancer microarray data. *PLoS ONE*, 9(7):e102803.

Liu, Z., Malone, B., and Yuan, C. (2012). Empirical evaluation of scoring functions for Bayesian network model selection. *BMC Bioinformatics*, 13(Suppl 15):S14.

Malone, B., Kangas, K., Järvisalo, M., Koivisto, M., and Myllymäki, P. (2014). Predicting the hardness of learning Bayesian networks. In *Proc. AAAI*, pages 2460–2466. AAAI Press.

Mitchell, T. (1997). *Machine Learning*. McGraw-Hill.

Ott, S. and Miyano, S. (2003). Finding optimal gene networks using biological constraints. *Genome Informatics*, 14:124 – 133.

Parviainen, P., Farahani, H. S., and Lagergren, J. (2014). Learning bounded tree-width Bayesian networks using integer linear programming. In *Proc. AISTATS*, pages 751–759. JMLR.org.

Parviainen, P. and Koivisto, M. (2009). Exact structure discovery in Bayesian networks with less space. In *Proc. UAI*, pages 436–443. AUAI Press.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc.

Russell, S. J. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach*. Pearson Education.

Silander, T. and Myllymäki, P. (2006). A simple approach for finding the globally optimal Bayesian network structure. In *Proc. UAI*, pages 445–452. AUAI Press.

Silander, T., Roos, T., Kontkanen, P., and Myllymäki, P. (2008). Factorized normalized maximum likelihood criterion for learning Bayesian network structures. In *Proc. PGM*, pages 257–272.

Spirtes, P., Glymour, C., and Schemes, R. (2000). *Causation, Prediction, and Search*. The MIT Press, 2 edition.

Teyssier, M. and Koller, D. (2005). Ordering-based search: A simple and effective algorithm for learning Bayesian networks. In *Proc. UAI*, pages 548–549. AUAI Press.

Tsamardinos, I., Brown, L., and Aliferis, C. (2006). The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65:31–78.

Ueno, M. (2010). Learning networks determined by the ratio of prior and data. In *Proc. UAI*, pages 598–605. AUAI Press.

Ueno, M. (2011). Robust learning Bayesian networks for prior belief. In *Proc. UAI*, pages 698–707. AUAI Press.

Yuan, C. and Malone, B. (2013). Learning optimal Bayesian networks: A shortest path perspective. *Journal of Artificial Intelligence Research*, 48:23–65.