The Thirty-Third International
FLAIRS Conference (FLAIRS-33)

Learning NAT-Modeled Bayesian Networks from Data

Yang Xiang, Qian Wang

School of Computer Science
University of Guelph
Canada

Abstract

Bayesian networks (BNs) encode conditional independence to avoid combinatorial explosion on the number of variables, but are subject to exponential growth of space and inference time on the number of causes per effect variable. Among space-efficient local models, we focus on the Non-Impeding Noisy-AND Tree (NIN-AND Tree or NAT) models due to their multiple merits and on NAT-modeled BNs, where each multi-parent variable family may be encoded as a NAT-model. Although BN inference is generally exponential on treewidth, inference is tractable with NAT-modeled BNs of high treewidth and low density. In this work, we present the first study to learn NAT-modeled BNs from data. We apply the MDL principle to learning NAT-modeled BNs by developing a corresponding scoring function, and we couple it with heuristic structure search. We show that when data satisfy NAT causal independence, high treewidth, and low density structure, learning underlying NAT modeled BNs is feasible.

1 Introduction

Discrete BNs avoid combinatorial explosion on the number of variables by encoding conditional independence in directed acyclic graph (DAG) structures, but space and inference time grow exponentially in the number of causes per effect due to tabular conditional probability tables (CPTs). Space-efficient local models exist, such as noisy-OR, noisy-MAX (Henrion 1989), context-specific independence (CSI) (Boutilier et al. 1996), NAT (Xiang 2012), DeMorgan (Maaskant and Druzdzel 2008), tensor-decomposition (Vomlel and Tichavsky 2012), and cancellation (Woudenberg, van der Gaag, and Rademaker 2015).

We focus on NAT models due to merits of simple causal interactions (reinforcement/undermining), expressiveness (recursive mixture of causal interactions, multi-valued, ordinal or nominal (Xiang and Jiang 2018)), generality (generalizing noisy-OR, noisy-MAX, and DeMorgan), and orthogonality to CSI. Although BN inference is generally exponential on treewidth, inference is tractable with NAT-modeled BNs of high treewidth and low density.

Specifically, the space of a BN is $O(n\ s^{\kappa})$, where n is the number of variables, s bounds domain sizes of variables, and κ bounds numbers of causes (parents) per variable. In fully NAT-modeled BNs (see Section 2), variables quantify dependency on parents by NAT models instead of tabular CPTs. Their space is $O(n\ s\ \kappa)$. This efficiency extends to inference time with NAT modeled BNs of high treewidth (bounded by κ) and low density (measured by percentage of arcs beyond being singly connected) structures.

This work studies learning NAT-model BNs from data. A BN can be compressed into a fully NAT-modeled BN (Xiang and Jiang 2018). However, since the source BN must be either manually constructed or learned from data through other methods, the compression approach does not completely solve knowledge acquisition for NAT-modeled BNs.

The main contribution of this work is the first study on learning NAT-modeled BNs directly from data. We apply the MDL principle (Rissanen 1978) to learning NAT-modeled BNs to develop a NAT-enabled scoring function, and couple it with heuristic structure search. Our experiment shows that when data satisfy NAT causal independence, high treewidth, and low density structure, it is feasible to learn underlying NAT-modeled BNs that enable inference efficiency and accuracy.

In developing the MDL function, we resolve the following issues: We propose a decomposition of description length for NAT-modeled BNs. We show how to incorporate into description length persistent leaky causes (see Section 5) discovered during learning. We reveal the break-down of MDL decomposability, and propose remedy to maintain accuracy of MDL score and learning efficiency. We identify the role of NAT compression in learning NAT-modeled BNs.

The remainder is organized as follows: We review terminology on NAT-modeled BNs in Section 2. Section 3 motivates the task of learning NAT-modeled BNs. Decomposition of MDL scoring function for NAT-modeled BNs is presented in Section 4. How should each component sub-score be computed is presented in Sections 5 through 8. The structure search is described in Section 9 with complexity analysis. We report initial experimental results in Section 10.

Two general applications of NAT-modeled BNs can be identified. This work opens the door for possibility of a third, which is discussed in Section 11 along with future research.

2 NAT-modeled Bayesian Networks

We review terminology on NAT-modeled BNs.

Causal Variables and Causal Events

A NAT model is defined over an effect e and a set of $\kappa \geq 2$ *uncertain* causes $C = \{c_1, ..., c_\kappa\}$, where $e \in D_e = \{e^0, ..., e^\eta\}$ $(\eta \geq 1)$ and $c_i \in \{c_i^0, ..., c_i^{m_i}\}$ $(i = 1, ..., \kappa, m_i \geq 1)$. C and e form one family (a child variable plus its parents) in BNs. Values e^0 and c_i^0 are *inactive*. Other values (may be written as e^+ or c_i^+) are *active*. That causes in a NAT model are uncertain implies the following (they can cause effect to be active, but do not always do so):

$$0 < P(e^k | c_1^{j_1}, ..., c_\kappa^{j_\kappa}) < 1 \quad (k > 0, \exists_i j_i > 0). \tag{1}$$

That C is the set of all causes implies the following:

$$P(e^0 | c_1^0, ..., c_\kappa^0) = 1. \tag{2}$$

A causal event is a *success* or *failure* depending on if e is active up to a given value, is *single-* or *multi-causal* depending on the number of active causes, and is *simple* or *congregate* depending on the value range of e. For instance, $P(e^k \leftarrow c_i^j) = P(e^k | c_i^j, c_z^0 : \forall z \neq i)$ $(j > 0)$ is probability of a *simple single-causal success*. $P(e \geq e^k \leftarrow c_1^{j_1}, ..., c_q^{j_q})$

$$= P(e \geq e^k | c_1^{j_1}, ..., c_q^{j_q}, c_z^0 : c_z \in C \setminus X)$$

is probability of a *congregate multi-causal success*, where $j_1, ..., j_q > 0$, $X = \{c_1, ..., c_q\}$ $(q > 1)$, and it may be denoted as $P(e \geq e^k \leftarrow \underline{x}^+)$ where $\underline{x}^+$ corresponds to X.

NAT Models

Interactions among causes in a NAT model may be reinforcing or undermining: Let e^k be an active effect value, $R = \{W_1, ..., W_\phi\}$ $(\phi \geq 2)$ be a partition of a set $X \subseteq C$ of causes, $S \subset R$, and $Y = \cup_{W_i \in S} W_i$. Sets of causes in R *reinforce* each other relative to e^k, iff

$$\forall S \; P(e \geq e^k \leftarrow \underline{y}^+) \leq P(e \geq e^k \leftarrow \underline{x}^+), \tag{3}$$

where $\underline{y}^+$ corresponds to Y. They *undermine* each other iff

$$\forall S \; P(e \geq e^k \leftarrow \underline{y}^+) > P(e \geq e^k \leftarrow \underline{x}^+). \tag{4}$$

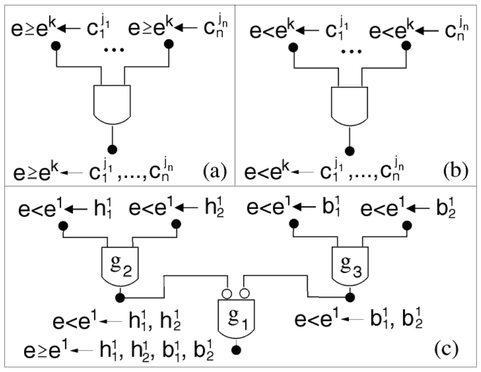
e≥e^k← c_1^{j_1}   e≥e^k← c_n^{j_n}
e≥e^k← c_1^{j_1},...,c_n^{j_n}
(a)
e<e^k← c_1^{j_1}   e<e^k← c_n^{j_n}
e<e^k← c_1^{j_1},...,c_n^{j_n}
(b)
e<e^1← h_1^1   e<e^1← h_2^1   e<e^1← b_1^1   e<e^1← b_2^1
g_2
g_3
g_1
e<e^1← h_1^1, h_2^1
e<e^1← b_1^1, b_2^1
e≥e^1← h_1^1, h_2^1, b_1^1, b_2^1
(c)

Figure 1: Direct (a) and dual (b) NIN-AND gates. (c) NAT.

A NAT consists of multiple NIN-AND gates. A *direct* gate involves disjoint sets of causes $W_1, ..., W_\phi$. Each input event is a success $e \geq e^k \leftarrow \underline{w}_i^+$ $(i = 1, ..., \phi)$, e.g., Fig. 1 (a) where each W_i is a singleton. The output event is $e \geq e^k \leftarrow \underline{w}_1^+, ..., \underline{w}_\phi^+$ with probability

$$P(e \geq e^k \leftarrow \underline{w}_1^+, ..., \underline{w}_\phi^+) = \prod_{i=1}^{\phi} P(e \geq e^k \leftarrow \underline{w}_i^+), \tag{5}$$

and hence direct gates encode undermining.

Each input event of a *dual* gate is a failure $e < e^k \leftarrow \underline{w}_i^+$, e.g., Fig. 1 (b). The output event is $e < e^k \leftarrow \underline{w}_1^+, ..., \underline{w}_\phi^+$ with probability

$$P(e < e^k \leftarrow \underline{w}_1^+, ..., \underline{w}_\phi^+) = \prod_{i=1}^{\phi} P(e < e^k \leftarrow \underline{w}_i^+), \tag{6}$$

and hence dual gates encode reinforcement.

Fig. 1 (c) shows a NAT, where causes h_1 and h_2 reinforce, so do b_1 and b_2, but the two groups undermine each other. A NAT is quantified by *single-causals* (probability parameters of root events), in the form $P(e^k \leftarrow c_i^j)$ $(j, k > 0)$. From the NAT and single-causals, $P(e \geq e^1 \leftarrow h_1^1, h_2^1, b_1^1, b_2^1)$, as well as other values of CPT $P(e | h_1, h_2, b_1, b_2)$, are uniquely specified.

NAT-modeled BNs

A BN, where CPTs of some variable families are NAT models, is a *NAT-modeled BN*, e.g., Fig. 2. Family of v_8 is a NAT model, whose NAT is in (b) (simplified notation). Gate g_3 is dual, and g_1 and g_2 are direct. A BN is *fully* NAT-modeled, if every multi-parent family is a NAT model. The above BN is so when families of v_5, v_8, v_9 are NAT-modeled.

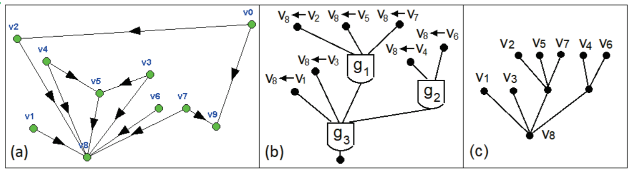

Figure 2: (a) DAG of a NAT-model BN. (b) NAT structure over the family of v_8. (c) Root labeled tree of NAT in (b).

Compression of NAT Models

An arbitrary tabular CPT (referred to as *source CPT*) can be approximated by a NAT model, termed *compression*, as follows: The CPT is analyzed to determine causal interaction (reinforcing or undermining) between every pair of causes if possible, with the result being a pair-wise causal interaction (PCI) pattern. From the PCI pattern, compatible candidate NATs are extracted, and are parameterized into NAT models through constrained gradient descent search. The output NAT is selected to minimize a distance measure between source CPT and the NAT CPT.

3 Learning NAT-modeled BNs

A NAT-modeled BN can be obtained by first learning a BN with tabular CPTs from data, and compressing it into a fully NAT-modeled BN. This approach has several limitations:

First, it relies on other methods to acquire a tabular BN, and does not completely resolve acquisition for NAT-modeled BNs. Second, the DAG structure of the resultant NAT-modeled BN is the same as the tabular BN. Since the structure was obtained independently of NAT-modeling, it may not be the most suitable structure for the NAT-modeled BN both efficiency-wise and accuracy-wise. Third, a tabular CPT may not be accurately approximated by a NAT model. The approach does not guide the compression explicitly by trade-off between representational efficiency and accuracy. To overcome these limitations, this work studies learning NAT-modeled BNs directly from data.

A common approach for learning BN structures from data is to combine heuristic search of structures with a scoring function, e.g., BD and MDL. The MDL principle (Rissanen 1978) views the best model of a dataset as one minimizing sum of encoding lengths of the model and the data. More accurate models reduce data encoding length, but increase model encoding length as they are more complex. Hence, the MDL principle offers a trade-off between accuracy and efficiency.

In this work, we extend structure learning of BNs to learning NAT-modeled BNs using a NAT-enabled MDL scoring function to trade off representational efficiency and accuracy. Our work also belongs to structure learning with local structures, e.g., (Friedman and Goldszmidt 1996; Chickering, Heckerman, and Meek 1997). The main difference is that their work focused on equality constraints such as CSI, with decision trees or decision graphs as local structures. Our current work focuses on inequality constraints such as Eqns. (3) and (4), with NAT models as local structures. Therefore, our work complements existing BN structure learning with local structures.

We develop a NAT-enabled MDL score below, couple with a heuristic structure search, and evaluate feasibility of learning NAT-modeled BNs from data experimentally.

4 NAT-Enabled MDL Scoring Function

Since a tabular CPT may not be sufficiently accurately approximated by a NAT model, we make the compression decision by a NAT-enabled MDL score. The MDL function for a tabular BN is additively decomposed into the model description length and the data description length. Both need to be extended to allow NAT modeling. We assume that the dataset D involves n variables $X_1, ..., X_n$ (additional variables may be introduced due to NAT modeling as shown below), and includes N data records. Since the MDL function is score equivalent (Chickering 1995) and decomposable (see below on its violation due to NAT-modeling and the proposed solution), we consider description length over a single family, made of a variable X_i and its parent set π_i.

To allow trade-off of representational efficiency and accuracy, dependency of X_i on π_i may be expressed by tabular CPT or NAT model. With tabular CPT, description length of the X_i family is decomposed into model description length and data description length:

$$DL_{TabCpt} = DL_{Model} + DL_{TabData},$$

where X_i in $DL_{TabCpt}(X_i)$ and other terms are omitted.

Model description length over the family can be further decomposed into DAG description length (relative to DAG structure over the family) and CPT description length (relative to tabular CPT):

$$DL_{TabCpt} = DL_{Dag} + DL_{Cpt} + DL_{TabData}.$$

A NAT-modeled BN has a global structure (the DAG) and local structures (NATs). If the X_i family forms a NAT model, its DAG description length is similar to that of a tabular CPT (except the difference in Section 7). Since the NAT CPT is defined by NAT structure and single-causals, CPT description length is decomposed into NAT description length and single-causal description length:

$$DL_{NatMod} = DL_{Dag} + DL_{Nat} + DL_{Sc} + DL_{NatData}.$$

As will be seen, data description length also differs depending on whether tabular CPT or NAT model is used (Section 8), and hence the naming of $DL_{TabData}$ and $DL_{NatData}$. The following sections present these components of description length separately in the order of

$$DL_{Sc}, DL_{Nat}, DL_{Dag}, DL_{NatData}.$$

5 Single-Causal Description Length

Let s_i denote the domain size of X_i, $\kappa_i = |\pi_i|$ denote the number of parents of X_i, and s_{ij} denote the domain size of the jth parent of X_i. If q_i denotes the number of configurations of π_i, we have $q_i = \prod_{j=1}^{\kappa_i} s_{ij}$. If X_i has a tabular CPT $P(X_i|\pi_i)$, the number of CPT parameters is $(s_i - 1)q_i$. Each parameter is typically encoded with $\frac{1}{2}log_2(N)$ bits (Friedman and Yakhini 1996). Hence, the CPT description length is

$$DL_{Cpt} = \frac{1}{2}log_2(N)(s_i - 1)\prod_{j=1}^{\kappa_i} s_{ij} \ (bits). \qquad (7)$$

If the X_i family is NAT modeled, where π_i is the set of all causes of X_i (see below for alternative), the number of single-causals needed to specify the NAT model is

$$(s_i - 1)\sum_{j=1}^{\kappa_i}(s_{ij} - 1).$$

Encoding each parameter with $\frac{1}{2}log_2(N)$ bits, single-causal description length of the X_i family is

$$DL_{Sc} = \frac{1}{2}log_2(N)(s_i - 1)\sum_{j=1}^{\kappa_i}(s_{ij} - 1) \ (bits). \qquad (8)$$

The above description length is applicable when X_i family has no persistent leaky cause (Henrion 1989; Xiang and Jiang 2018), but must be extended otherwise. The *leaky* cause for an effect X_i represents all causes of X_i that are not explicitly named. A leaky cause may be persistent. A *non-persistent* leaky cause can be modeled as other causes. In such cases, we denote all causes of effect e by $c_1, ..., c_\kappa$. If the leaky cause is non-persistent, we assume that one of $c_1, ..., c_\kappa$ is the leaky cause. When the X_i family forms a NAT model, we have $e = X_i$ and $\kappa = \kappa_i$. A source CPT with a non-persistent leaky cause has a fully specified $P(e|c_1, ..., c_n)$, where $P(e^0|c_1^0, ..., c_n^0) = 1$ and $P(e^k|c_1^0, ..., c_n^0) = 0$ for $k > 0$. Hence, Eqn. (2) holds.

A persistent leaky cause is always active. We integrate all persistent leaky causes of the same effect into a single cause, and denote the leaky cause by c_0. In such cases, we denote other causes of effect e by $c_1, ..., c_\kappa$. Since c_0 is persistent, we have $c_0 \in \{c_0^0, c_0^1\}$, and $c_0 = c_0^1$ always holds. Because conditions $(c_0^0, c_1, ..., c_n)$ never hold, and parameters $P(e|c_0^0, c_1, ..., c_n)$ are not empirically available, a source CPT has the form $Q(e|c_1, ..., c_n) = P(e|c_0^1, c_1, ..., c_n)$. Since c_0 is an uncertain cause, by Eqn. (1), we have

$$0 < P(e|c_0^1, c_1^0, ..., c_n^0) = Q(e|c_1, ..., c_n) < 1.$$

Hence, Eqn. (2) does not hold with the source CPT $Q(e|c_1, ..., c_n)$, which triggers identification during learning: If the X_i family forms a NAT model, it involves a persistent leaky cause. The source CPT $Q(e|c_1, ..., c_n)$ of n causes is compressed into $P_{Nat}(e|c_0, c_1, ..., c_n)$ of $n+1$ causes.

Due to the extra persistent leaky cause of the NAT model, and that it is binary, additional $s_i - 1$ single-causals are needed to specify the NAT model: $P(e^k \leftarrow c_0^1)$ $(k > 0)$. Hence, when the family of X_i forms a NAT model with the persistent leaky cause, Eqn. (8) no longer applies. Instead, the single-causal description length is

$$DL_{Sc} = \frac{1}{2} log_2(N)(s_i - 1)(1 + \sum_{j=1}^{\kappa_i}(s_{ij} - 1)). \quad (9)$$

6 NAT Description Length

We consider two options of NAT encoding. A NAT can be expressed as *root labeled tree* (RLT), where a node represents a root event, or the leaf event, or a gate, preserving the tree topology. The RLT of NAT in Fig. 2 (b) is shown in (c). When the family of X_i forms a NAT model, the NAT can be encoded by encoding RLT, e.g., encoding parent set of each node in the RLT. For RLT of m nodes, encoding index of each node takes $log_2(m)$ bits. For instance, if $\kappa_i = 3$, the RLT has as fewer as 4 nodes (3 roots and 1 leaf). Encoding index of each node with $log_2(4) = 2$ bits, the RLT can be encoded with $2*4 = 8$ bits. If $\kappa_i = 15$, the RLT has as fewer as 16 nodes, and can be encoded with $4 * 16 = 64$ bits.

Alternatively, the RLT can be encoded by encoding its unique PCI pattern. With $\kappa_i = |\pi_i|$ denoting the number of parents of X_i, encoding PCI pattern takes $\kappa_i\ (\kappa_i - 1)/2$ bits. When $\kappa_i = 3$, we need 3 bits. When $\kappa_i = 15$, we need $15 * 14/2 = 105$ bits.

The above shows that none of the options dominates the other: For simplicity, we use PCI pattern-based encoding. If the family of X_i forms a NAT model without persistent leaky cause, the NAT description length is

$$DL_{Nat} = \frac{1}{2}\kappa_i(\kappa_i - 1)\ (bits).$$

If the NAT model involves a persistent leaky cause, the number of causes increases to $\kappa_i + 1$. The description length is

$$DL_{Nat} = \frac{1}{2}\kappa_i(\kappa_i + 1)\ (bits).$$

7 DAG Description Length

DAG description length over the family of X_i encodes parent set π_i. Let n be the number of variables in the dataset D. It takes $log_2(n)$ bits to encode the index of each node. For tabular BNs, DAG description length of the family of X_i is

$$DL_{Dag} = log_2(n)\ \kappa_i.$$

For NAT-modeled BNs, the number of nodes in the DAG may be greater than n, invalidating the above. For each NAT family with a persistent leaky cause, an extra node is introduced. The extra node has both local and global impact to DAG description length:

Locally, since X_i has an extra parent, factor κ_i in DL_{Dag} becomes $1 + \kappa_i$. Globally, with the extra variable, $log_2(n)$ bits are insufficient to encode index of each node. Let β be the total number of persistent leaky cause variables (each over a distinct variable family) introduced at a given time during learning. Then DAG description length over the family of X_i is

$$DL_{Dag} = log_2(n + \beta)\ (1 + \kappa_i)\ bits. \quad (10)$$

Since β changes as learning proceeds, existence of β in DL_{Dag} introduces dependency between description lengths over different variable families, and breaks down decomposability of the MDL function.

To ensure accuracy of MDL score, we deployed the following: When a new NAT family with persistent leaky cause is learned, or such a family learned earlier is invalidated during learning, we apply a global updating of family scores to adjust β value in $log_2(n + \beta)$ coefficient. This allows decomposability of the MDL function to persist between NAT family updates, and enables efficient score computation.

8 Data Description Length

When CPT of X_i is tabular, data description length over the family is

$$DL_{TabData} = -\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} s_{ijk}\ log_2\Big(\frac{s_{ijk}}{M_{ij}}\Big),$$

where s_{ijk} counts family configurations $(X_i = k, \pi_i = \pi_{ij})$ in D, M_{ij} counts parent configurations $(\pi_i = \pi_{ij})$, and s_{ijk}/M_{ij} estimates $P(X_i = k|\pi_i = \pi_{ij})$.

When family of X_i is a NAT model Θ_i, the above data description length must replace $P(X_i = k|\pi_i = \pi_{ij})$ with $P_{\Theta_i}(X_i = k|\pi_i = \pi_{ij})$ defined by the NAT model CPT. Data description length over the family of X_i is

$$DL_{NatData} = -\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} s_{ijk}\ log_2\Big(P_{\Theta_i}(X_i = k|\pi_i = \pi_{ij})\Big).$$

The above requires fully specifying the NAT model Θ_i. To do so, we first estimate tabular CPT $P(X_i|\pi_i)$ from D, and then compress the CPT into Θ_i. Since computation of $DL_{NatData}$ requires compression, it is significantly more costly than $DL_{TabData}$ in learning tabular BNs.

9 Heuristic Search and Complexity

As learning BNs from data is NP-complete, a number of heuristics have been proposed for search of alternative structures. One may start with a complete graph, remove links by conditional independence, and orient the resultant graph. One may also start with an empty graph, add arcs until no further addition improves the score, and then remove arcs until no further removal improves the score. Alternatively, arcs may be added, deleted, or reversed until it is no longer possible to improve the score. The search may also be organized by orderings of variables, rather than by DAGs. A recent summary of search heuristics in BN structure learning can be found in (Lee and van Beek 2017). As the first study of learning NAT-modeled BNs, a heuristic similar to that of (Friedman and Goldszmidt 1996) is extended in our learning algorithm, referred to as $LearnNatBn$. As its pseudocode at a reasonable level of details cannot fit into the space limit, we describes its key components below:

$LearnNatBn$ takes as input a dataset D over a set V of variables. It learns a NAT-modeled BN $\mathcal{N} = (G, \Omega, \Theta)$. G is a DAG possibly over a superset of V (due to persistent leaky causes). Ω is a set of CPTs one for each variable family whose dependency is quantified by tabular CPT. Θ is a set of NAT models one for each variable family whose dependency is quantified by NAT model.

$LearnNatBn$ starts with an empty DAG. The MDL score of a DAG G is denoted as $DL(G)$. Search proceeds in multiple rounds. Each round tries to find a DAG G' that differs from G by one arc, whose score $DL(G')$ is better than $DL(G)$. Search terminates when no such DAG can be found to improve the score.

For each newly formed variable family in the current DAG, MDL sub-scores are computed for both tabular CPT and NAT model. For X_i with $\kappa_i = |\pi_i|$, the number of alternative NAT models over the X_i family is super-exponential in κ_i. Instead of computing a MDL sub-score for each NAT model, we compress the tabular CPT (estimated from data) into a NAT model. That is, the search through the NAT space is conducted by compression, and the best NAT model found is MDL-scored. Decision to model the family as tabular CPT or NAT model is made by comparing the two sub-scores. If the NAT model is selected that involves persistent leaky cause, a new variable will be included in the current DAG.

For complexity of $LearnNatBn$, denote $n = |V|$. $O(n^2)$ links are evaluated before one is added, removed, or reversed. At most $O(n^2)$ links can be added, yielding complexity of $O(n^4)$. The above does not count cost of NAT-model compression, including PCI pattern recognition, NAT extraction, and NAT parameterization. Complexity of PCI pattern recognition for a variable of κ parents is $O(\kappa^2)$ (Xiang and Jiang 2018). For PCI pattern with β missing bits, complexity of NAT extraction is $O(\kappa^2 \, 2^{\beta+\kappa})$. Let s bounds variable domain sizes, and τ bounds steps in gradient descent during NAT parameterization. Complexity of NAT parameterization is $O(\kappa \, s^2 \, \tau \, 2^{\beta})$.

Due to compression, learning NAT-modeled BNs is significantly more costly. To be as efficient as possible, $LearnNatBn$ also uses mutual information between pairs of variables to reduce links evaluated in each round.

Before $LearnNatBn$, D is pre-processed into a set F of frequencies of unique records over V. $|F|$ is significantly $< |D|$, and complexity of $LearnNatBn$ is linear on $|F|$.

10 Initial Experimental Results

To establish feasibility of learning NAT-modeled BNs from data, we generated 30 fully NAT-modeled BNs (Fig. 3), referred to as *source BNs*. Each source BN consists of 200 binary or ternary variables. The maximum number of parents per variable is 12. The density of the DAG is controlled by adding 5% extra arcs beyond being singly-connected. Each source BN is transformed to an equivalent *peer* tabular BN, from which a dataset of size $N = 5000$ is sampled as input to $LearnNatBn$.

Among the 200 variable families in each source BN, between 18 and 28% are NAT-models, and the rest have tabular CPTs. In NAT-modeled BNs, between 11 and 18% of variable families are NAT-models.

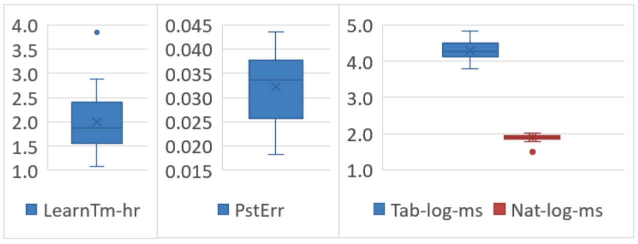

Figure 3: Summery of experimental results

Fig. 3 (left) reports learning time in hours. Learned NAT-modeled BNs are evaluated by accuracy of inference and efficiency gain, relative to the peer BN. Each peer BN is compiled into a junction tree for lazy propagation. Each learned BN is de-causalized (Xiang and Loker 2018) and compiled into junction tree. Ten runs of inference are performed on each peer BN and each learned BN, by observing 10% of randomly selected variables.

For inference accuracy, average differences on posterior marginals over all variables (10 runs per BN) are reported in Fig. 3 (middle). Learned NAT-modeled BNs yield sufficiently accurate posteriors with average errors between 0.018 and 0.044.

For efficiency gain in inference, average runtimes ($msec$; 10 runs per BN) in log10 for peer BNs (Tab-log-ms) and learned BNs (Nat-log-ms) are shown in Fig. 3 (right). Learned NAT-modeled BNs are between 110 and 990 times faster in inference.

The initial experiment suggests that when data satisfy NAT causal independence, and high treewidth, low density structure, it is feasible to learn underlying NAT-modeled BNs that enable inference efficiency and accuracy.

11 Conclusion and Future Work

The main contribution is the first investigation on learning NAT-modeled BNs. Although this study is not the first on learning BNs with local structures, previous work mainly focused on equality constraints such as CSI, with decision trees or decision graphs as local structures. This work focuses on inequality constraints with NAT models as local structures. Hence, this work complements existing literature on BN learning with local structures. Contributions also include development of the NAT-enabled MDL function, coupling it with a heuristic search, and initial empirical study on feasibility of learning NAT-modeled BNs from data.

Two general applications of NAT-modeled BNs are identified in the peer-reviewed literature: First, they offer a tractable subclass of BNs for knowledge representation and acquisition (in line with the recent trend about *tractable models* such as SPNs). Through recursive, reinforcing/undermining local modeling, they reduce space of BNs from $O(n\ s^{\kappa})$ to $O(n\ s\ \kappa)$. For high treewidth, low density BNs, they enable tractable inference through techniques such as de-causalization. Second, they offer a more efficient approximation of intractable BNs through compression, trading accuracy for efficiency.

The current work opens the door for a third possibility, where NAT-modeled BNs are used directly for modeling data. To realize this option, a number of research issues are yet to be addressed:

Feasibility of NAT-models as alternative for modeling data needs to be evaluated. They are most beneficial when underlying dependency structure has high treewidth and low density. Existing real world BNs often do not fit this profile. For instance, the 9 medium or large BNs in the BN Repository has the maximum number of parents per node of 7. We hypothesize the reason to be difficulty with tabular CPT elicitation (exponential human time) and learning (exponential data). NAT-modeled BNs promise to remove the difficulty. To test the hypothesis, learning NAT-model BNs from real world data needs to be conducted.

The initial experiment found that strength of dependency between individual causes and their effect in the same NAT model is far from uniformly distributed. Due to the uneven strength of dependency, a cause in the source NAT model may be excluded during learning. Implication of such exclusion should be evaluated. Deeper understanding of strength of dependency within NAT models is needed, e.g., how NAT topology and single-causal values determine relative strength of dependency for individual causes.

Investigation on learning of NAT-models requires simulation of source models as experimental testbeds. It is desirable that similarity between source and learned BNs positively validates learning. That is, source BNs should be faithful models. Fueled by deeper understanding of the dependency within NAT models, simulations that generate such source BNs should be developed.

The initial experimental study tested one heuristic search method. Many alternatives exist, e.g., NAT-enabled scoring is applied at only the last round of search. Further research to compare alternative heuristics relative to quality of output NAT-modeled BNs and efficiency of learning is needed.

Additional issues for future research include integration of alternative encodings relative to DL_{Nat}, improved NAT-modeling for small datasets, and comparison with learning algorithms utilizing other local models.

Acknowledgement

Financial support from the NSERC Discovery Grant to the first author is acknowledged. We thank anonymous reviewers for their time, effort, and constructive criticism.

References

Boutilier, C.; Friedman, N.; Goldszmidt, M.; and Koller, D. 1996. Context-specific independence in Bayesian networks. In *Proc. 12th Conf. on Uncertainty in Artificial Intelligence*, 115–123.

Chickering, D.; Heckerman, D.; and Meek, C. 1997. A Bayesian approach to learning Bayesian networks with local structure. In *Proc. of 13th Conf. on Uncertainty in Artificial Intelligence*, 80–89.

Chickering, D. 1995. A transformational characterization of equivalent Bayesian network structures. In *Proc. 11th Conf. on Uncertainty in Artificial Intelligence*, 87–98.

Friedman, N., and Goldszmidt, M. 1996. Learning Bayesian networks with local structure. In *Proc. 12th Conf. on Uncertainty in Artificial Intelligence*, 252–262. Morgan Kaufmann.

Friedman, N., and Yakhini, Z. 1996. On the sample complexity of learning Bayesian networks. In *Proc. 12th Conf. on Uncertainty in Artificial Intelligence*, 274–282. Morgan Kaufmann.

Henrion, M. 1989. Some practical issues in constructing belief networks. In Kanal, L.; Levitt, T.; and Lemmer, J., eds., *Uncertainty in Artificial Intelligence 3*. Elsevier Science Publishers. 161–173.

Lee, C., and van Beek, P. 2017. Metaheuristics for score-and-search Bayesian network structure learning. In *Proc 30th Canadian Conf. on Artificial Intelligence*, 129–141.

Maaskant, P., and Druzdzel, M. 2008. An independence of causal interactions model for opposing influences. In Jaeger, M., and Nielsen, T., eds., *Proc. 4th European Workshop on Probabilistic Graphical Models*, 185–192.

Rissanen, J. 1978. Modeling by shortest data description. *Automatica* 14:465–471.

Vomlel, J., and Tichavsky, P. 2012. An approximate tensor-based inference method applied to the game of Minesweeper. In *Proc. 7th European Workshop on Probabilistic Graphical Models, Springer LNAI 8745*, 535–550.

Woudenberg, S.; van der Gaag, L.; and Rademaker, C. 2015. An intercausal cancellation model for Bayesian-network engineering. *Inter. J. Approximate Reasoning* 63:32–47.

Xiang, Y., and Jiang, Q. 2018. NAT model based compression of Bayesian network CPTs over multi-valued variables. *Computational Intelligence* 34(1):219–240.

Xiang, Y., and Loker, D. 2018. De-causalizing NAT-modeled Bayesian networks for inference efficiency. In Bagheri, E., and Cheung, J., eds., *Canadian AI 2018, LNAI 10832*. Springer. 17–30.

Xiang, Y. 2012. Non-impeding noisy-AND tree causal models over multi-valued variables. *International J. Approximate Reasoning* 53(7):988–1002.