Cutset Bayesian Networks: A New Representation for Learning Rao-Blackwellised Graphical Models

Tahrima Rahman* , **Shasha Jin** and **Vibhav Gogate**
The University of Texas at Dallas
{tahrima.rahman, shasha.jin, vibhav.gogate}@utdallas.edu

Abstract

Recently there has been growing interest in learning probabilistic models that admit poly-time inference called tractable probabilistic models from data. Although they generalize poorly as compared to intractable models, they often yield more accurate estimates at prediction time. In this paper, we seek to further explore this trade-off between generalization performance and inference accuracy by proposing a novel, partially tractable representation called cutset Bayesian networks (CBNs). The main idea in CBNs is to partition the variables into two subsets $\boldsymbol{X}$ and $\boldsymbol{Y}$, learn a (intractable) Bayesian network that represents $P(\boldsymbol{X})$ and a tractable conditional model that represents $P(\boldsymbol{Y}|\boldsymbol{X})$. The hope is that the intractable model will help improve generalization while the tractable model, by leveraging Rao-Blackwellised sampling which combines exact inference and sampling, will help improve the prediction accuracy. To compactly model $P(\boldsymbol{Y}|\boldsymbol{X})$, we introduce a novel tractable representation called conditional cutset networks (CCNs) in which all conditional probability distributions are represented using calibrated classifiers—classifiers which typically yield higher quality probability estimates than conventional classifiers. We show via a rigorous experimental evaluation that CBNs and CCNs yield more accurate posterior estimates than their tractable as well as intractable counterparts.

1 Introduction

A major issue in using probabilistic graphical models such as Bayesian networks for solving real-world tasks is the intractability of probabilistic inference [Darwiche, 2009; Dechter, 2013]; the latter is NP-hard in general and computationally infeasible in practice. One approach to tackle the intractability of exact inference is to use poly-time approximate inference approaches such as likelihood weighting and belief propagation. However, approximate techniques are unreliable; specifically even if the model has excellent test set log-likelihood scores, its predictions can be way off because approximate inference schemes (which are used to make predictions) are often inaccurate. An alternative approach for tackling the intractability of inference is to learn tractable models—models that admit exact poly-time inference—from data. However, tractable models generalize poorly as compared to intractable models where generalization performance is measured using the average test set log-likelihood score. Despite poor generalization, tractable models are preferred over the approximate inference approach because numerous experimental studies have demonstrated that query answers derived from the former are often more accurate than the ones derived from the latter [Rooshenas and Lowd, 2014].

In this paper, we seek to learn probabilistic models that balance the trade-off between generalization performance and inference accuracy by leveraging Rao-Blackwellised (or cutset) sampling [Casella and Robert, 1996; Doucet *et al.*, 2000; Gogate and Dechter, 2005; Bidyuk and Dechter, 2007]. At a high level, Rao-Blackwellised sampling combines exact inference schemes with sampling techniques as follows. It partitions the set of variables into two subsets, say $\boldsymbol{X}$ and $\boldsymbol{Y}$, performs sampling-based inference on $\boldsymbol{X}$ and exact inference on $\boldsymbol{Y}$ for each sampled assignment $\boldsymbol{X} = \boldsymbol{x}$. Using the Rao-Blackwell theorem [Casella and Robert, 1996], it is easy to show that Rao-Blackwellised sampling will yield higher quality estimates as compared to pure sampling schemes (e.g., Gibbs sampling) that sample *all* variables in the model.

Thus, a straight-forward approach for learning models that admit efficient, poly-time Rao-Blackwellised sampling is to heuristically partition the variables into two subsets $\boldsymbol{X}$ and $\boldsymbol{Y}$, learn a (potentially intractable) Bayesian network that represents $P(\boldsymbol{X})$ and a tractable model that represents the conditional distribution $P(\boldsymbol{Y}|\boldsymbol{X})$. However, an issue with this approach is that $P(\boldsymbol{Y}|\boldsymbol{X})$ needs to be defined for each assignment $\boldsymbol{X} = \boldsymbol{x}$ and is thus exponential in the number of variables in $\boldsymbol{X}$. To address this issue, we propose a novel representation for $P(\boldsymbol{Y}|\boldsymbol{X})$ which is both compact and tractable called *conditional cutset networks* (CCNs).

CCNs are based on a popular class of tractable models called AND/OR cutset networks or CNs in short [Rahman *et al.*, 2014] that compactly model large joint probability distributions. These networks take advantage of fine-grained properties of probability distributions such as context-specific independence [Boutilier *et al.*, 1996], identical probabil-

*Contact Author

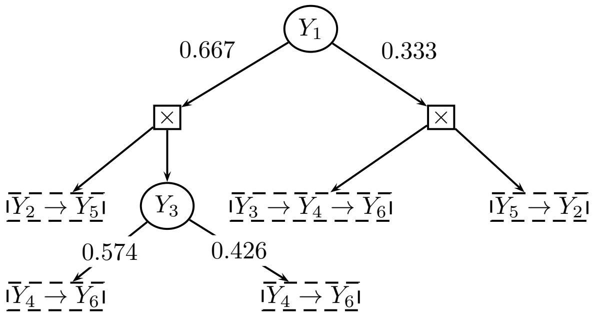

Figure 1: A cutset network defined over $\{Y_1, \ldots, Y_6\}$. OR, AND and leaf nodes (tree Bayesian networks) are denoted by ovals, squares and dotted rectangles respectively. Left and right arcs emanating from an OR node labeled by Y_i indicate conditioning over $Y_i = 1$ and $Y_i = 0$ respectively and are labeled with conditional probabilities.

ity values and determinism [Chavira and Darwiche, 2008; Gogate, 2009] as well as coarse properties such as conditional independence to yield compact, yet rich models that can faithfully model complex dependencies in many application domains (e.g., computer vision, natural language processing, etc.). At a high level, an AND/OR cutset network [Mateescu and Dechter, 2005] is a rooted, directed acyclic graph which consists of alternating levels of *OR/sum* and *AND/product* nodes with tree Bayesian networks as leaf nodes. OR nodes model conditioning while AND nodes model decomposition.

A CCN extends a CN to compactly represent $P(\boldsymbol{Y}|\boldsymbol{X})$ as follows. It attaches a calibrated classifier[1] [Niculescu-Mizil and Caruana, 2005] to each OR node and each conditional probability distribution in each tree Bayesian network. These classifiers take an assignment $\boldsymbol{X} = \boldsymbol{x}$ as input, treat $Y \in \boldsymbol{Y}$ as a class variable and output a well-calibrated probability distribution over Y. Compactness is achieved because the number of parameters used by the classifiers typically scales polynomially with $|\boldsymbol{X}|$. Tractability is achieved because given $\boldsymbol{X} = \boldsymbol{x}$, CCNs yield a (tractable) cutset network.

This paper makes three contributions. First, we propose a novel representation called cutset Bayesian networks (CBNs) that admits efficient Rao-Blackwellised sampling schemes. CBNs combine Bayesian networks and CCNs by partitioning the variables into two subsets $\boldsymbol{X}$ and $\boldsymbol{Y}$, using a Bayesian network to model $P(\boldsymbol{X})$ and a CCN to model $P(\boldsymbol{Y}|\boldsymbol{X})$. Second, we propose a novel structure learning algorithm for inducing CCNs (and thus CBNs) from data. Finally, we demonstrate via a thorough experimental evaluation that CBNs have superior generative as well as discriminative performance as compared to state-of-the-art CNs and Bayesian networks.

2 Background

Let $\mathbf{X} = \{X_1, \ldots, X_n\}$ and $\mathbf{Y} = \{Y_1, \ldots, Y_m\}$ denote sets of Boolean variables where each variable $X_i \in \mathbf{X}$ and $Y_j \in \mathbf{Y}$ takes values from the domain $\{0, 1\}$. Let $\boldsymbol{x}$ denote an assignment of values to all variables in $\boldsymbol{X}$. Given a subset $\boldsymbol{S}$ of $\boldsymbol{X}$, we denote the projection of $\boldsymbol{x}$ on $\boldsymbol{S}$ by $\boldsymbol{x}_{\boldsymbol{S}}$.

[1]Calibrated classifiers yield higher quality probability estimates over the class variables as compared with conventional classifiers.

AND/OR cutset networks [Rahman and Gogate, 2016b], which we call CNs in short, are probabilistic models for representing large, multidimensional discrete probability distributions. Graphically, a CN is a rooted directed AND/OR tree (or acyclic graph) with a tree Bayesian network attached to each leaf node of the AND/OR tree. A CN over a set of variables $\boldsymbol{Y}$ is defined recursively as follows:

- A tree Bayesian network over $\boldsymbol{Y}$ is a CN.
- An OR node labeled by a variable $Y_i \in \boldsymbol{Y}$ such that $|\boldsymbol{Y}| > 1$ with two child CNs, each defined over the set $\boldsymbol{Y} \setminus \{Y_i\}$ is a CN. The arcs from the OR node to its child nodes are labeled with probability values in $\mathbb{R}^+$ such that they sum to 1.
- Let $(\boldsymbol{Y}_1, \boldsymbol{Y}_2)$ be a partition of $\boldsymbol{Y}$ (i.e., $\boldsymbol{Y}_1 \cup \boldsymbol{Y}_2 = \boldsymbol{Y}$ and $\boldsymbol{Y}_1 \cap \boldsymbol{Y}_2 = \emptyset$) such that $|\boldsymbol{Y}| > 1$. Then, an AND node with two child CNs, one defined over $\boldsymbol{Y}_1$ and the second defined over $\boldsymbol{Y}_2$ is a CN.

In CNs, OR nodes represent conditioning over the labeled variable while AND nodes represent decomposition or conditional independence—the probability distribution associated with an AND node is the product of the probability distribution at its children. Each edge emanating from an OR node is labeled with the conditional probability of the variable taking the particular value given an assignment of values from the root to the parent node. Let $\mathcal{C}$ denote a CN defined over a set of variables $\boldsymbol{Y}$. The probability of $\boldsymbol{y}$ w.r.t. $\mathcal{C}$ is given by

$$P_{\mathcal{C}}(\boldsymbol{y}) = \prod_{T \in \mathcal{T}_{\mathcal{C}}(\boldsymbol{y})} T(\boldsymbol{y}_{V(T)}) \prod_{p_j \in p_{\mathcal{C}}(\boldsymbol{y})} p_j \tag{1}$$

where $\mathcal{T}_{\mathcal{C}}(\boldsymbol{y})$ is the set of tree Bayesian networks and $P_{\mathcal{C}}(\boldsymbol{y})$ is the set of conditional probability values respectively on the sub-tree corresponding to the assignment $\boldsymbol{y}$ in $\mathcal{C}$, and $V(T)$ denotes the variables of T. Fig. 1 shows an example cutset network defined over the set $\{Y_1, \ldots, Y_6\}$ of variables.

CNs are *tractable* in that they admit linear time algorithms (linear in the number of parameters) for two popular inference tasks: posterior marginal estimation (MAR) and (full) maximum-a-posteriori estimation (MAP). MAR is defined as the task of computing the *marginal probability distribution* over all non-evidence variables in a probabilistic model given evidence where evidence is defined as an assignment of values to a subset of variables. MAP is defined as the task of finding the *most likely assignment* to all non-evidence variables given evidence. Both tasks can be solved by performing two-passes over the CN [Dechter and Mateescu, 2007].

3 Conditional Cutset Networks

In this section, we propose a novel *template model* called conditional cutset networks (CCNs) for compactly representing conditional probability distributions $P(\boldsymbol{Y}|\boldsymbol{X})$ for all assignments $\boldsymbol{X} = \boldsymbol{x}$. We also describe a scalable structure learning algorithm for CCNs. We will use this algorithm and the CCN representation in the next section to define and learn cutset Bayesian networks, a novel representation for multidimensional joint probability distributions.

A naive approach for representing and learning $P(\boldsymbol{Y}|\boldsymbol{X})$ using cutset networks is to learn a cutset network over $\boldsymbol{Y}$ for each

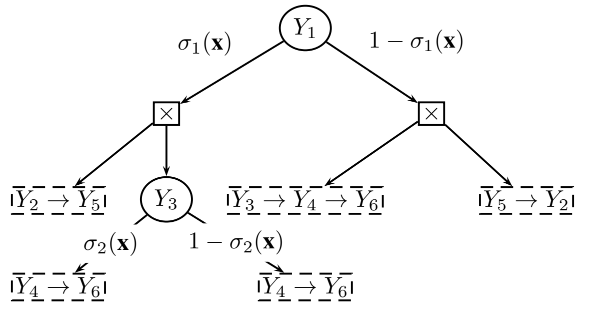

Figure 2: A conditional cutset network defined over 6 variables $\{Y_1,\ldots,Y_6\}$ given $\boldsymbol{X} = \{X_1, X_2, X_3\}$. Parameters of the sigmoid $\sigma_1(\boldsymbol{x})$ are (0.39,-0.5,0.3,0.305) and that of $\sigma_2(\boldsymbol{x})$ are (0.1,-0.7,-0.9,0.2). The first weight in each sigmoid function is the bias term. Sigmoids on the tree Bayesian network are not shown for brevity. The cutset network obtained by instantiating the CCN with the assignment $(X_1 = 0, X_2 = 0, X_3 = 1)$ is given in Fig. 1.

assignment $\boldsymbol{X} = \boldsymbol{x}$. Unfortunately, this approach is infeasible, especially when $|\boldsymbol{X}|$ is large. To address this issue, we propose to *template* the cutset networks as follows. We replace each conditional probability distribution in the cutset network—at each OR node and at each variable in each tree Bayesian network—with a function of $\boldsymbol{X}$ such that the function has polynomial (in $|\boldsymbol{X}|$) number of parameters. Specifically, we propose to represent each conditional distribution $P(Y|\boldsymbol{X}, \pi_o)$ at an OR node o using calibrated classifiers [Niculescu-Mizil and Caruana, 2005] where $Y \in \boldsymbol{Y}$, $\pi_o \in \boldsymbol{\Pi}_o$ and $\boldsymbol{\Pi}_o$ is the set of assignments on all paths from root of the CCN to o. Unlike conventional classifiers, calibrated classifiers yield accurate conditional probability estimates over the class variable Y. For instance, if we use a calibrated logistic regression classifier, we have $P(Y = 1|\boldsymbol{x}, \pi_o) = \sigma\left(w_0 + \sum_i w_i x_i\right)$ where σ is the sigmoid function and the weights w_i, $0 \leq i \leq |\boldsymbol{X}|$ are real numbers. We call conditional probability distributions (CPDs) represented using calibrated classifiers (CC) as CC-CPDs. More formally, a CCN is defined as follows:

Definition 1 (CCNs) *Let $\boldsymbol{X}, \boldsymbol{Y}$ be disjoint sets of random variables. A conditional cutset network is a cutset network over $\boldsymbol{Y}$ in which each conditional probability distribution is represented using CC-CPDs defined over $\boldsymbol{X}$.*

A CCN thus represents an exponential number of cutset networks. This is because given $\boldsymbol{X} = \boldsymbol{x}$, it yields a cutset network over $\boldsymbol{Y}$. Fig. 2 shows an example CCN that models the conditional distribution $P(Y_1,\ldots,Y_6|X_1,X_2,X_3)$. Instantiating the CCN with the assignment $(X_1 = 0, X_2 = 0, X_3 = 1)$ yields the cutset network given in Fig. 1. Thus, CCNs are conditionally tractable in that given $\boldsymbol{X} = \boldsymbol{x}$, they admit linear time MAP and MAR inference.

3.1 Structure Learning

Next, we present an algorithm for learning the structure of CCNs (see Algorithm 1). The learning algorithm follows the generic prescription given in prior work [Rahman *et al.*, 2014; Rahman and Gogate, 2016b] with the following major change: the sufficient statistics are computed using conditional or discriminative methods.

Algorithm 1: LEARNCCN (D, $\boldsymbol{Y}$, $\boldsymbol{X}$, E, ϵ)

```
Input: Training Instances $D$, Set of variables $\boldsymbol{Y}$, $\boldsymbol{X}$, an Integer
       $E > 0$ and a real number $\epsilon$.
Output: A CCN representing $P(\boldsymbol{Y}|\boldsymbol{X})$
1  begin
2      if $D$ has fewer than $E$ examples then
3          return CONDITIONAL-CHOW-LIU($D$,$\boldsymbol{Y}$, $\boldsymbol{X}$)
4      end
5      for all pairs $(Y_i, Y_j) \in \boldsymbol{Y}$, $i \neq j$ do
6          Compute $\overline{P}(Y_i, Y_j|\boldsymbol{X})$ using calibrated classifiers
7          Use $\overline{P}$ and $D$ to compute $Score(Y_i, Y_j)$ (see Eq. (2))
8      end
9      Construct a complete graph $G$ with variables in $\boldsymbol{Y}$ as nodes
         and edges weighted by $Score(Y_i, Y_j)$
10     Remove edges having weight $< \epsilon$ from $G$
11     if $G$ has $J > 1$ connected components then
12         Construct an AND node $a$ having $J$ child nodes $o_1,\ldots,o_J$
13         for $j = 1$ to $J$ do
14             $o_j$=LEARNCCN($D_j$,$\boldsymbol{Y}_j$, $\boldsymbol{X}$,$E$,$\epsilon$)
15         end
16         return $a$
17     end
18     Heuristically select $Y \in \mathbf{Y}$ (see Eq. (3)) and construct an OR
         node $o$ labeled with $Y$ and having two child nodes $l$ and $r$
19     Label the edge from $o$ to $l$ and from $o$ to $r$ with $\overline{P}(Y = 1|\boldsymbol{X})$
         and $1 - \overline{P}(Y = 1|\boldsymbol{X})$ respectively
20     $l$ =LEARNCCN($D|Y = 1$,$\boldsymbol{Y} \setminus \{Y\}$, $\boldsymbol{X}$, $E$, $\epsilon$)
21     $r$ =LEARNCCN($D|Y = 0$,$\boldsymbol{Y} \setminus \{Y\}$, $\boldsymbol{X}$, $E$, $\epsilon$)
22     return $o$
23 end
```

The input to the algorithm are two sets of variables: $\boldsymbol{X}$ and $\boldsymbol{Y}$, a dataset D over $\boldsymbol{X}$ and $\boldsymbol{Y}$ and two constants E and ϵ. The algorithm has three recursive steps which induce OR nodes, AND nodes and tree Bayesian networks respectively. In the base/termination step (lines 2-4), the algorithm induces a tree Bayesian network (leaf node) using the conditional Chow-Liu algorithm [Hong *et al.*, 2014] if the number of training examples is smaller than E. In the decomposition step, the algorithm partitions the variables in $\boldsymbol{Y}$ into multiple parts $\{\boldsymbol{Y}_1,\ldots,\boldsymbol{Y}_J\}$ if certain conditions are satisfied and recurses on each part, inducing an AND node (lines 5-17). In line 14, the notation D_j refers to the projection of training instances D on the variables in the set $\boldsymbol{X} \cup \boldsymbol{Y}_j$. If neither the base case nor the conditions for the decomposition step are satisfied, then the algorithm executes the splitting step (lines 18-22). In this step, the algorithm induces an OR node o labeled by a heuristically chosen variable $Y \in \boldsymbol{Y}$ and having two child nodes l and r (line 18). Then it recursively constructs CCNs on l and r (lines 20-21). In line 20 (21), the notation $D|Y = 1$ ($D|Y = 0$) denotes a dataset in which training instances having $Y = 0$ ($Y = 1$) are removed from D.

Next, we describe the details of the three main steps (decomposition, splitting, base case) of Algorithm 1.

In the *decomposition step*, we seek to partition $\boldsymbol{Y}$ into subsets $\{\boldsymbol{Y}_1,\ldots,\boldsymbol{Y}_J\}$ such that $P(\boldsymbol{Y}|\boldsymbol{X})$ equals $\prod_{j=1}^{J} P(\boldsymbol{Y}_j|\boldsymbol{X})$, namely each subset is conditionally independent of the other given $\boldsymbol{X}$. A naive approach will consider all possible partitions of $\boldsymbol{Y}$ and check whether the above condition is satisfied by estimating the strength of conditional independence using the conditional mutual information (CMI) score. However, this approach is impractical because it has exponential time complexity. To address this problem, we propose to approximate CMI using pairwise CMI and estimate the latter using calibrated classifiers. In particular, we propose to approximate the pairwise CMI between Y_i and Y_j using

$$Score(Y_i, Y_j) = \frac{1}{N}\sum_{k=1}^{N}\sum_{y_i}\sum_{y_j}\Big(\overline{P}(y_i, y_j|\boldsymbol{x}^{(k)}) \log\left[\frac{\overline{P}(y_i, y_j|\boldsymbol{x}^{(k)})}{\overline{P}(y_i|\boldsymbol{x}^{(k)})\overline{P}(y_j|\boldsymbol{x}^{(k)})}\right]\Big) \quad (2)$$

where $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}\}$ denotes the examples in D projected on $\boldsymbol{X}$ and $\overline{P}(y_i, y_j|\boldsymbol{x})$ is a calibrated classifier learned from the dataset D at each recursive step with $\{Y_i, Y_j\}$ as the class variable and $\boldsymbol{X}$ as the attributes.

$Score(Y_i, Y_j)$ measures the strength of dependence between Y_i and Y_j. We can use it to construct a partition of $\boldsymbol{Y}$ as follows. We first construct a complete graph G which has one node for each variable in $\boldsymbol{Y}$. Then we delete all edges from G such that $Score(Y_i, Y_j) < \epsilon$, where ϵ is a small pre-defined constant (the deleted edges signify weaker dependencies). Each connected component of G corresponds to a component of partition of $\boldsymbol{Y}$.

In the *splitting step*, we heuristically choose a variable to condition on. At a high level, we would like to have small depth CCNs (compact models) and to achieve this we should choose a variable that quickly yields a decomposition of variables or the base case. Literature on graphical models and cutset networks [Dechter, 2013; Rahman *et al.*, 2014; Di Mauro *et al.*, 2015b] suggests that the best heuristic in such cases is to choose a variable that has strong dependencies with other variables (highest degree in a graphical model). To measure the strength of the dependencies, we propose to use $Score(Y_i, Y_j)$, our approximation to pairwise CMI, and select the variable having the highest score (defined below) to condition on (line 18 in Alg. 1).

$$VarScore(Y_i) = \sum_{j:j\neq i} Score(Y_i, Y_j) \quad (3)$$

In the *base case*, we construct a (conditional) tree Bayesian network using the following approach [Hong *et al.*, 2014]. We construct a complete graph G in which each edge is labeled with $Score(Y_i, Y_j)$ (see Eq. (2)) and then choose a maximum spanning tree over G.

4 Cutset Bayesian Networks

CCNs can be used to model a joint distribution over a disjoint subsets of random variables $\boldsymbol{X}$ and $\boldsymbol{Y}$ using the chain rule, namely $P(\boldsymbol{X}, \boldsymbol{Y}) = P_{\mathcal{B}}(\boldsymbol{X})P_{\mathcal{C}}(\boldsymbol{Y}|\boldsymbol{X})$ where $P_{\mathcal{C}}(\boldsymbol{Y}|\boldsymbol{X})$ is represented using a CCN $\mathcal{C}$ and $P_{\mathcal{B}}(\boldsymbol{X})$ is represented using a Bayesian network $\mathcal{B}$. We call such models *cutset Bayesian networks* (CBNs). The main virtue of CBNs is that they help us explore the trade-off between generalization performance (which Bayesian networks are good at) and inference accuracy (which cutset networks are good at) by leveraging Rao-Blackwellised or cutset sampling approaches [Casella and Robert, 1996; Bidyuk and Dechter, 2007]. Next, we describe how to derive accurate MAR estimates using cutset sampling.

4.1 Cutset Importance Sampling

At a high level, in cutset (Rao-Blackwellised) sampling, we combine exact and sampling based approximate inference by only sampling a subset of variables and then *exactly* inferring over the remaining variables for each sampled assignment. We can apply this approach to CBNs by sampling all variables ($\boldsymbol{X}$) in the Bayesian network and exactly inferring over all variables ($\boldsymbol{Y}$) in the conditional cutset network given $\boldsymbol{X} = \boldsymbol{x}$.

Formally, let $\boldsymbol{A}$ and $\boldsymbol{E}$ denote the non-evidence and evidence variables of $\boldsymbol{X}$ respectively. Similarly, let $\boldsymbol{B}$ and $\boldsymbol{T}$ denote the non-evidence and evidence variables of $\boldsymbol{Y}$ respectively. Let $(\boldsymbol{E} = \boldsymbol{e}, \boldsymbol{T} = \boldsymbol{t})$ be the evidence. Then, we can compute the probability of evidence, as follows:

$$P(\boldsymbol{e}, \boldsymbol{t}) = \sum_{\boldsymbol{a},\boldsymbol{b}} P_{\mathcal{B}}(\boldsymbol{a}, \boldsymbol{e})P_{\mathcal{C}}(\boldsymbol{b}, \boldsymbol{t}|\boldsymbol{a}, \boldsymbol{e}) \quad (4)$$

$$= \sum_{\boldsymbol{a}} \frac{P_{\mathcal{B}}(\boldsymbol{a}, \boldsymbol{e})}{Q(\boldsymbol{a})} Q(\boldsymbol{a}) \sum_{\boldsymbol{b}} P_{\mathcal{C}}(\boldsymbol{b}, \boldsymbol{t}|\boldsymbol{a}, \boldsymbol{e}) \quad (5)$$

where $Q(\boldsymbol{a})$ is a proposal distribution defined over $\boldsymbol{A}$ such that whenever $P_{\mathcal{B}}(\boldsymbol{a}, \boldsymbol{e}) > 0$, $Q(\boldsymbol{a}) > 0$. The proposal distribution can be chosen using heuristic approaches [Gogate and Dechter, 2005] or it can equal the prior distribution such as in likelihood weighting [Bidyuk and Dechter, 2006]. Given samples $(\boldsymbol{a}^{(1)}, \ldots, \boldsymbol{a}^{(N)})$ generated uniformly at random from Q, we can estimate $P(\boldsymbol{e}, \boldsymbol{t})$ using the following unbiased estimator:

$$\widehat{P}(\boldsymbol{e}, \boldsymbol{t}) = \frac{1}{N}\sum_{i=1}^{N} \frac{P_{\mathcal{B}}(\boldsymbol{a}^{(i)}, \boldsymbol{e})}{Q(\boldsymbol{a}^{(i)})} \sum_{\boldsymbol{b}} P_{\mathcal{C}}(\boldsymbol{b}, \boldsymbol{t}|\boldsymbol{a}^{(i)}, \boldsymbol{e}) \quad (6)$$

Note that for each sample $\boldsymbol{a}^{(i)}$, the quantity $\sum_{\boldsymbol{b}} P_{\mathcal{C}}(\boldsymbol{b}, \boldsymbol{t}|\boldsymbol{a}^{(i)}, \boldsymbol{e})$ can be computed (exactly) in time that scales linearly with the size of the conditional cutset network and thus $\widehat{P}(\boldsymbol{e}, \boldsymbol{t})$ can be computed efficiently. It is known that the mean squared error of $\widehat{P}(\boldsymbol{e}, \boldsymbol{t})$ can be reduced by either increasing the number of samples N or by reducing the variance of weights. The latter in turn depends on how close the proposal distribution is to the posterior distribution given by $\alpha P_{\mathcal{B}}(\boldsymbol{a}^{(i)}, \boldsymbol{e}) \sum_{\boldsymbol{b}} P_{\mathcal{C}}(\boldsymbol{b}, \boldsymbol{t}|\boldsymbol{a}^{(i)}, \boldsymbol{e})$ where α is a normalization constant.

We can show that the Rao-Blackwell estimator $\widehat{P}(\boldsymbol{e}, \boldsymbol{t})$ is more accurate than the conventional importance sampling estimator (which samples all variables in $\boldsymbol{A} \cup \boldsymbol{B}$) given below:

$$\widetilde{P}(\boldsymbol{e}, \boldsymbol{t}) = \frac{1}{N}\sum_{i=1}^{N} \frac{P_{\mathcal{B}}(\boldsymbol{a}^{(i)}, \boldsymbol{e})P_{\mathcal{C}}(\boldsymbol{b}^{(i)}, \boldsymbol{t}|\boldsymbol{a}^{(i)}, \boldsymbol{e})}{Q(\boldsymbol{a}^{(i)}, \boldsymbol{b}^{(i)})} \quad (7)$$

where $Q(\boldsymbol{a}, \boldsymbol{b})$ is a proposal distribution defined over $\boldsymbol{A} \cup \boldsymbol{B}$, and $(\boldsymbol{a}^{(1)}, \boldsymbol{b}^{(1)}), \ldots, (\boldsymbol{a}^{(N)}, \boldsymbol{b}^{(N)})$ are the samples generated from Q. Specifically,

Proposition 1 *The variance of $\widehat{P}(\boldsymbol{e}, \boldsymbol{t})$ is smaller than or equal to $\widetilde{P}(\boldsymbol{e}, \boldsymbol{t})$ under the assumption $Q(\boldsymbol{a}) = \sum_{\boldsymbol{b}} Q(\boldsymbol{a}, \boldsymbol{b})$.*

Dataset	**Generative Performance (LL Scores)**				
	BN	CBN (%X)			CN
		20	50	80	
NLTCS	-6.00	-6.02	-6.01	-6.01	-6.00
MSNBC	-6.09	-6.04	-6.04	-6.06	-6.25
KDD	-2.13	-2.18	-2.15	-2.13	-2.15
Plants	**-12.17**	-12.82	-12.38	-12.19	-12.46
Audio	-39.19	-40.32	-39.28	**-39.12**	-40.22
Jester	-51.96	-53.00	-52.02	**-51.88**	-52.91
Netflix	-55.44	-56.52	-55.33	**-55.26**	-56.67
Accident	-26.25	-27.71	-26.62	**-26.11**	-28.78
Retail	-10.81	-10.87	-10.82	**-10.80**	-10.91
Pumsb*	**-21.46**	-22.93	-21.92	-21.51	-24.73
DNA	**-76.49**	-78.94	-77.16	-76.60	-82.49
Kosarek	**-10.56**	-10.78	-10.60	-10.57	-10.84
MSWEB	-9.76	-10.16	-9.80	-9.76	-9.82
Book	**-33.62**	-34.36	-33.84	-33.72	-35.38
Movie	-48.61	-50.02	-49.11	**-48.52**	-53.81
WebKB	-144.02	-147.75	-145.33	**-143.94**	-156.03
Reuters	**-80.78**	-84.43	-81.54	-81.12	-86.00
20NG	-147.98	-150.3	-147.89	**-147.86**	-156.09
BBC	-230.70	-238.95	-231.82	**-230.30**	-236.25
Ad	-14.39	-14.55	**-14.01**	-14.01	-15.16
Average	-51.42	-52.93	-51.68	**-51.38**	-54.14

(a)

Dataset	**Discriminative Performance (CLL Scores)**								
	20% Evidence			50% Evidence			80% Evidence		
	DACL	CCN	CN	DACL	CCN	CN	DACL	CCN	CN
NLTCS	-4.31	-4.33	**-4.30**	**-2.56**	-2.58	-2.57	-0.98	-0.99	-0.98
MSNBC	-4.59	-4.59	-4.81	-2.18	-2.18	-2.37	**-0.86**	-0.87	-0.98
KDD	**-1.60**	-1.63	-1.61	-1.19	-1.19	-1.20	-0.36	-0.36	-0.37
Plants	**-8.73**	-8.89	-8.96	**-4.51**	-4.53	-4.79	-1.64	**-1.54**	-1.70
Audio	**-31.38**	-31.83	-31.61	**-18.64**	-18.67	-19.31	-7.66	**-7.58**	-7.99
Jester	**-41.48**	-41.88	-41.71	-25.03	**-24.96**	-25.61	-9.89	**-9.75**	-10.25
Netflix	**-44.07**	-44.30	-44.27	-26.12	**-26.03**	-26.84	-10.31	**-10.22**	-10.72
Accident	-19.33	**-18.50**	-21.27	**-10.01**	-10.24	-12.83	**-3.55**	-3.61	-4.95
Retail	**-8.42**	-8.45	-8.45	-4.89	**-4.88**	-4.93	-1.67	**-1.66**	-1.70
Pumsb*	-15.16	**-14.72**	-16.30	**-6.87**	-6.98	-9.18	**-1.98**	-2.02	-2.67
DNA	-61.30	**-60.12**	-67.75	-34.90	**-32.98**	-46.87	-13.06	**-12.29**	-14.82
Kosarek	**-8.77**	-8.81	-8.80	-4.79	**-4.76**	-4.92	-1.16	**-1.14**	-1.22
MSWEB	**-8.65**	-9.00	-8.69	**-4.19**	-4.25	-4.31	**-1.67**	-1.69	-1.75
Book	-27.46	**-26.85**	-28.26	-16.39	**-15.90**	-17.54	-6.73	**-6.48**	-7.74
Movie	-39.77	**-36.55**	-41.94	-26.73	**-24.85**	-31.86	-9.18	**-8.40**	-13.55
WebKB	-118.21	**-114.40**	-122.82	-72.31	**-69.34**	-77.32	-27.04	**-25.76**	-29.69
Reuters	-68.86	**-65.19**	-66.55	-39.20	**-36.56**	-40.00	-16.97	**-15.67**	-18.13
20NG	-122.76	**-119.00**	-125.25	-75.01	**-71.69**	-78.66	-29.29	**-27.72**	-31.75
BBC	-198.26	**-188.23**	-200.87	-124.36	**-114.52**	-129.49	-47.30	**-43.27**	-50.13
Ad	-8.94	**-8.52**	-9.79	-4.09	**-3.41**	-4.65	-1.57	**-1.18**	-1.93
Average	-42.10	**-40.79**	-43.20	-25.20	**-24.03**	-27.26	-9.64	**-9.11**	-10.65

(b)

Table 1: (a) Generative and (b) discriminative performance measured using the test set log-likelihood (LL) and conditional log-likelihood (CLL) score respectively of CBNs, CCNs and competing algorithms. Bold values indicate significantly higher scores.

Remarks: (1) $\widehat{P}(\boldsymbol{e}, \boldsymbol{t})$ can be easily modified to yield posterior marginal estimates using the normalized importance sampling approach (cf. [Liu, 2008]). **(2)** CBNs help us trade inference accuracy with generalization performance because as the number of variables in $\boldsymbol{X}$ is increased we expect that the generalization performance will improve while inference accuracy will drop because the variance of $\widehat{P}(\boldsymbol{e}, \boldsymbol{t})$ will increase. CBNs include BNs and CNs as special cases. All we have to do is set $\boldsymbol{Y} = \emptyset$ and $\boldsymbol{X} = \emptyset$ respectively.

4.2 Learning CBNs

CBNs can be learned from data using the following approach: (1) Heuristically partition the variables into two subsets $\boldsymbol{X}$ and $\boldsymbol{Y}$; (2) Learn a BN over $\boldsymbol{X}$; and (3) Learn a CCN over $\boldsymbol{Y}$ given $\boldsymbol{X}$ using Algorithm 1. We leave partitioning heuristics for future work and use a random partition in our experiments.

5 Experiments

We evaluate the performance of CBNs on 20 benchmark datasets (see Table 2 for their characteristics) which have been used extensively in previous studies [Rooshenas and Lowd, 2014]. We evaluate the following algorithms: (1) bags of cutset networks (CNs) which serves as a strong state-of-the-art baseline; (2) CBNs: we randomly select (exactly) $T\%$ of the variables to include in the set $\boldsymbol{X}$, with the rest as elements of $\boldsymbol{Y}$. We experiment with the following 4 values for T: 20%, 50%, 80% and 100%. In the latter case, the CBN yields a BN. Thus, our study involves 5 competing algorithms: CNs, BNs and CBNs with 20%, 50% and 80% of the variables designated as the set $\boldsymbol{X}$. Each algorithm was given a time bound of 48 hours. We learn a CCN over $\boldsymbol{Y}$ using Algorithm 1. We set E to 10 in Algorithm 1 and for pruning low mutual information edges, we use an adaptive threshold of $\epsilon = N^{-\beta}$ where N is the number of training examples and $\beta \in \{0.5, 0.7, 1.0\}$. We learn a complete (intractable) Bayesian network (BN) over $\boldsymbol{X}$ as follows: given a random order $(X_1, \ldots, X_n)$ over all the variables in $\boldsymbol{X}$, we represent each conditional probability distribution $P(X_i|X_1, \ldots, X_{i-1})$ using a calibrated classifier. We learn bags of CNs following [Rahman and Gogate, 2016a; Di Mauro *et al.*, 2017] with the number of bags fixed to 50 and the maximum depth fixed to 5.

We experimented with the following mixture of calibrated classifiers: $\alpha \times (LR) + (1-\alpha) \times (RF)$ where LR is a logistic regression classifier with ℓ_2 penalty $\lambda \in \{10^{-i}\}_{i=-1}^{i=3}$, RF is a random forest classifier having 40 random trees with depth chosen from $\{3, 6, 9\}$ and $\alpha \in [0, 1]$. We tuned the various hyper-parameters (e.g., α, λ, etc.) using the validation set.

5.1 Generative Performance

Table 1(a) shows the average test set log-likelihood (LL) scores achieved by fully connected Bayesian networks (BNs) (100% of variables in $\boldsymbol{X}$), CBNs and CNs. The scores show a clear trade-off between generalization performance and tractability. As we increase the size of $\boldsymbol{X}$ from 20% to 100% (BNs), namely as the exact inference complexity increases, the generalization performance improves. We observe that in many cases, CNs have worse test set LL scores than CBNs.

Table 2 shows the test set conditional marginal log-likelihood scores achieved by various models. Here, we randomly select 50% and 80% of the variables as query variables and the rest as evidence. Then we perform probabilistic inference to compute the posterior marginal distribution of each query variable given evidence. On CNs, we use exact

Datasets	**Dataset Characteristics**				**Conditional Marginal Log-Likelihood Scores**											
					50% Query						80% Query					
					BN	CN	CBN (% $\boldsymbol{X}$)				BN	CN	CBN (% $\boldsymbol{X}$)			
	#Var	#Train	#Valid	#Test			20	50	80	Best			20	50	80	Best
NLTCS	16	16181	2157	3236	-0.430	-0.348	-0.287	-0.249	-0.246	**-0.246**	**-0.140**	-0.171	-0.235	-0.187	-0.183	-0.183
MSNBC	17	291326	38843	58265	-0.512	-0.364	-0.349	-0.323	-0.324	**-0.323**	-0.432	-0.400	-0.387	-0.384	-0.439	**-0.384**
KDD	64	180092	19907	34955	**-0.028**	-0.036	-0.036	-0.035	-0.035	-0.035	-0.042	-0.045	-0.045	-0.042	-0.048	-0.042
Plants	69	17412	2321	3482	-0.206	-0.168	-0.072	-0.076	-0.084	**-0.072**	**-0.090**	-0.148	-0.112	-0.121	-0.111	-0.111
Audio	100	15000	2000	3000	**-0.420**	-0.431	-0.450	-0.438	-0.443	-0.438	**-0.436**	-0.483	-0.488	-0.467	-0.462	-0.462
Jester	100	9000	1000	4116	**-0.558**	-0.626	-0.606	-0.591	-0.564	-0.564	-0.589	-0.602	-0.624	-0.601	-0.588	**-0.588**
Netflix	100	15000	2000	3000	-0.501	-0.559	-0.546	-0.515	-0.497	**-0.497**	**-0.497**	-0.565	-0.564	-0.537	-0.514	-0.514
Accident	111	12758	1700	2551	-0.245	-0.249	-0.272	-0.222	-0.202	**-0.202**	**-0.144**	-0.192	-0.258	-0.208	-0.180	-0.180
Retail	135	22041	2938	4408	**-0.083**	-0.091	-0.086	-0.084	-0.085	-0.084	-0.113	-0.115	-0.113	-0.117	-0.113	-0.113
Pumsb*	163	12262	1635	2452	-0.190	-0.333	-0.209	-0.180	-0.196	**-0.180**	**-0.057**	-0.197	-0.146	-0.111	-0.069	-0.069
DNA	180	1600	400	1186	-0.540	-0.524	-0.512	-0.514	-0.526	**-0.512**	-0.446	-0.458	-0.469	-0.457	-0.442	**-0.442**
Kosarek	190	33375	4450	6675	-0.043	-0.071	-0.047	-0.045	-0.042	**-0.042**	-0.034	-0.038	-0.035	-0.034	-0.034	-0.034
MSWEB	294	29441	3270	5000	**-0.047**	-0.103	-0.066	-0.065	-0.052	-0.052	**-0.035**	-0.047	-0.039	-0.036	-0.038	-0.036
Book	500	8700	1159	1739	**-0.053**	-0.079	-0.063	-0.060	-0.056	-0.056	-0.058	-0.057	-0.054	-0.065	-0.059	**-0.054**
Movie	500	4524	1002	591	-0.169	-0.210	-0.148	-0.152	-0.166	**-0.148**	-0.259	-0.200	-0.161	-0.179	-0.194	**-0.161**
WebKB	839	2803	558	838	-0.316	-0.222	-0.202	-0.251	-0.301	**-0.202**	-0.265	-0.212	-0.212	-0.233	-0.266	**-0.212**
Reuters	889	6532	1028	1540	-0.106	-0.149	-0.092	-0.104	-0.103	**-0.092**	-0.116	-0.103	-0.098	-0.102	-0.125	**-0.098**
20NG	910	11293	3764	3764	-0.125	-0.135	-0.110	-0.111	-0.139	**-0.110**	-0.146	-0.112	-0.107	-0.104	-0.121	**-0.104**
BBC	1058	1670	225	330	-0.330	-0.249	-0.213	-0.305	-0.295	**-0.213**	-0.222	-0.215	-0.179	-0.191	-0.200	**-0.179**
Ad	1556	2461	327	491	**-0.005**	-0.212	-0.057	-0.021	-0.009	-0.009	**-0.004**	-0.107	-0.035	-0.012	-0.006	-0.006
Average					-0.222	-0.258	-0.221	-0.217	-0.218	**-0.204**	-0.206	-0.223	-0.218	-0.209	-0.210	**-0.199**

Table 2: Average test set conditional marginal log-likelihood (normalized by the number of query variables) of BNs, CNs and CBNs when we randomly select 50% and 80% of the variables as the query variables and the remaining as evidence variables. Column Best shows the score for CBNs having the smallest sample variance. Bold values indicate significantly higher score.

inference, on CBNs (Rao-Blackwellised) cutset likelihood weighting (see section 4.1) and on BNs likelihood weighting to compute the posterior marginal estimates. Table 2 includes a column titled "Best" which gives the scores for the CBN having the smallest sample variance (of weights) on each dataset. It is well known that the smaller the sample variance, the more accurate the algorithm. For the sampling algorithms, for each test example, we generate 1000 samples and repeat the experiment 10 times. From Table 2, we observe that CBNs outperform CNs on a majority of the datasets for both 50% and 80% query variables cases. Moreover, the CBN having the smallest variance (column "Best") almost always outperforms both CNs and BNs. Thus, our study suggests that we can select the best CBN for a particular test example by running inference over all CBNs in parallel and choosing the one having the smallest sample variance.

5.2 Discriminative Performance

We compare the discriminative performance of CCNs, namely when all variables in $\boldsymbol{X}$ are observed to disciminatively trained arithmetic circuits (DACL) [Rooshenas and Lowd, 2016] and bags of cutset networks (CNs) [Rahman and Gogate, 2016a] (the latter is a generative model). We use the Libra toolkit [Lowd and Rooshenas, 2015] to learn ACs following the methodology described in [Rooshenas and Lowd, 2016]. Table 1(b) shows the average test-set conditional log-likelihood (CLL) scores obtained by each of these models with (randomly selected) 20%, 50% and 80% evidence variables ($\boldsymbol{X}$). CCNs outperform DACL on 10, 12 and 14 out of the 20 datasets in the presence of 20%, 50% and 80% evidence variables respectively (there are several ties: MSNBC for 20% evidence, MSNBC and KDD for 50% evidence and KDD for the 80% case). Both CCNs and DACLs outperform the well-trained generative CNs in a majority of the cases. CCNs are significantly better than DACLs on datasets having more than 500 variables. This implies that CCNs are capable of learning more accurate models in high dimensions.

6 Conclusion

In this paper, we introduced a novel conditionally tractable model called conditional cutset networks (CCNs) and its generative variant called cutset Bayesian networks (CBNs) that combines CCNs with Bayesian networks. The main idea in the latter is to partition the variables into two subsets $\boldsymbol{X}$ and $\boldsymbol{Y}$, learn a possibly intractable model over $\boldsymbol{X}$ but learn a compact, conditionally tractable model for $P(\boldsymbol{Y}|\boldsymbol{X})$. This allows us to explore the trade off between tractability and generalization performance (measured using test-set likelihood score) in a systematic manner. We presented a novel structure learning algorithm for CCNs and showed that it runs in polynomial time by leveraging a novel approximation to conditional mutual information that uses calibrated classifiers and data. Experimentally, we showed that our new representation and learning algorithms yield superior posterior estimates when Rao-Blackwellised cutset sampling is used for inference.

Acknowledgements

This work was supported in part by the DARPA Explainable Artificial Intelligence (XAI) Program under contract number N66001-17-2-4032, and by the National Science Foundation grants IIS-1652835 and IIS-1528037.

References

[Bidyuk and Dechter, 2006] B. Bidyuk and R. Dechter. Cutset sampling with likelihood weighting. In *Proceedings of the Twenty-Second Conference Annual Conference on Uncertainty in Artificial Intelligence*, pages 39–46, 2006.

[Bidyuk and Dechter, 2007] B. Bidyuk and R. Dechter. Cutset Sampling for Bayesian Networks. *Journal of Artificial Intelligence Research*, 28:1–48, 2007.

[Boutilier *et al.*, 1996] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-Specific Independence in Bayesian Networks. In *Proceedings of the Twelfth Conference Annual Conference on Uncertainty in Artificial Intelligence*, pages 115–123, 1996.

[Casella and Robert, 1996] G. Casella and C. P. Robert. Rao-blackwellisation of sampling schemes. *Biometrika*, 83(1):81–94, 1996.

[Chavira and Darwiche, 2008] M. Chavira and A. Darwiche. On probabilistic inference by weighted model counting. *Artificial Intelligence*, 172(6-7):772–799, 2008.

[Darwiche, 2009] A. Darwiche. *Modeling and reasoning with Bayesian networks*. Cambridge university press, 2009.

[Dechter and Mateescu, 2007] R. Dechter and R. Mateescu. AND/OR search spaces for graphical models. *Artificial Intelligence*, 171:73–106, 2007.

[Dechter, 2013] R. Dechter. *Reasoning with Probabilistic and Deterministic Graphical Models: Exact Algorithms*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2013.

[Di Mauro *et al.*, 2015a] N. Di Mauro, A. Vergari, and T. Basile. Learning bayesian random cutset forests. In *Foundations of Intelligent Systems*, pages 122–132. Springer International Publishing, 2015.

[Di Mauro *et al.*, 2015b] N. Di Mauro, A. Vergari, and F. Esposito. Learning accurate cutset networks by exploiting decomposability. In *Proceedings of the Fourteenth International Conference of the Italian Association for Artificial Intelligence*, pages 221–232. Springer, 2015.

[Di Mauro *et al.*, 2017] N. Di Mauro, A. Vergari, T. Basile, and G. Esposito. Fast and accurate density estimation with extremely randomized cutset networks. In *In Proceedings of Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 203–219, 2017.

[Doucet *et al.*, 2000] A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-blackwellised particle filtering for dynamic bayesian networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 176–183, 2000.

[Friedman and Van den Broeck, 2018] T. Friedman and G. Van den Broeck. Approximate knowledge compilation by online collapsed importance sampling. In *Proceedings of the Thirty-Second International Conference on Neural Information Processing Systems*, pages 8035–8045, 2018.

[Gogate and Dechter, 2005] V. Gogate and R. Dechter. Approximate inference algorithms for hybrid bayesian networks with discrete constraints. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, pages 209–216, 2005.

[Gogate, 2009] V. Gogate. *Sampling algorithms for probabilistic graphical models with determinism*. PhD thesis, University of California, Irvine, 2009.

[Hong *et al.*, 2014] C. Hong, I. Batal, and M. Hauskrecht. A mixtures-of-trees framework for multi-label classification. In *Proceedings of the Twenty-Third ACM International Conference on Conference on Information and Knowledge Management*, pages 211–220, 2014.

[Liang *et al.*, 2017] Y. Liang, J. Bekker, and Guy Van den Broeck. Learning the structure of probabilistic sentential decision diagrams. In *Uncertainty in Artificial Intelligence*, 2017.

[Liu, 2008] J. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer Publishing Company, Incorporated, 2008.

[Lowd and Rooshenas, 2015] D. Lowd and A. Rooshenas. The libra toolkit for probabilistic models. *Journal of Machine Learning Research*, 16(1):2459–2463, 2015.

[Mateescu and Dechter, 2005] R. Mateescu and R. Dechter. AND/OR cutset conditioning. In *Proceedings of the Nineteenth international joint conference on Artificial intelligence*, pages 230–235, 2005.

[Niculescu-Mizil and Caruana, 2005] A. Niculescu-Mizil and R. Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the Twenty-Second International Conference on Machine Learning*, pages 625–632, 2005.

[Rahman and Gogate, 2016a] T. Rahman and V. Gogate. Learning ensembles of cutset networks. In *AAAI conference on Artificial Intelligence*, pages 3301–3307, 2016.

[Rahman and Gogate, 2016b] T. Rahman and V. Gogate. Merging strategies for sum-product networks: From trees to graphs. In *Proceedings of the Thirty-Second Conference Conference on Uncertainty in Artificial Intelligence*, pages 617–626, 2016.

[Rahman *et al.*, 2014] T. Rahman, P. Kothalkar, and V. Gogate. Cutset networks: A simple, tractable, and scalable approach for improving the accuracy of chow-liu trees. In *Proceedings of the Twenty-Fifth European Conference on Machine Learning*, pages 630–645, 2014.

[Rooshenas and Lowd, 2014] A. Rooshenas and D. Lowd. Learning sum-product networks with direct and indirect variable interactions. In *Proceedings of the Thirty-First International Conference on Machine Learning*, pages 710–718, 2014.

[Rooshenas and Lowd, 2016] A. Rooshenas and D. Lowd. Discriminative structure learning of arithmetic circuits. In *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Statistics*, pages 1506–1514, 2016.