Dynamic Cutset Networks

Chiradeep Roy, Tahrima Rahman, Hailiang Dong, Nicholas Ruozzi, Vibhav Gogate
The University of Texas at Dallas, Richardson, TX, 75080, USA
{*chiradeep.roy, tahrima.rahman, hailiang.dong, nicholas.ruozzi, vibhav.gogate*}*@utdallas.edu*

Abstract

Tractable probabilistic models (TPMs) are appealing because they admit polynomial-time inference for a wide variety of queries. In this work, we extend the cutset network (CN) framework, a powerful sub-class of TPMs that often outperforms probabilistic graphical models in terms of prediction accuracy, to the temporal domain. This extension, dubbed dynamic cutset networks (DCNs), uses a CN to model the prior distribution and a conditional CN to model the transition distribution. We show that although exact inference is intractable when arbitrary conditional CNs are used, particle filtering is efficient. To ensure tractability of exact inference, we introduce a novel conditional model called AND/OR conditional cutset networks and show that under certain restrictions exact inference is linear in the size of the corresponding constrained DCN. Experiments on several sequential datasets demonstrate the efficacy of our framework.

1 Introduction

Dynamic Bayesian networks (DBNs) (Dean and Kanazawa, 1989) and their (typically) discriminatively trained deep learning counterparts, recurrent neural networks, are widely used in practice to solve reasoning and prediction tasks in temporal domains—domains in which random variables evolve over time. However, DBNs have a well-known shortcoming: exact probabilistic inference over them is intractable. Moreover, approximate inference algorithms such as particle filtering (Carpenter et al., 1999; Doucet et al., 2001) often yield inaccurate estimates in practice because high quality proposal distributions are not readily accessible.

In this paper, we address the aforementioned shortcomings of DBNs by introducing a new probabilistic representation called dynamic cutset networks (DCNs) that extends the cutset network (CN) framework (Rahman et al., 2014; Rahman and Gogate, 2016a; Rahman et al., 2019) to temporal domains. CNs are tractable (static or non-temporal) probabilistic models that represent large multi-dimensional discrete probability distributions by leveraging AND/OR cutset conditioning (Mateescu and Dechter, 2005) as well as fine-grained features such as identical probability values, dynamic variable orders, and context-specific independence (Chavira and Darwiche, 2008; Boutilier et al., 1996). A CN consists of a rooted AND/OR graph (AND nodes are products and OR nodes are sums) with tree-structured Bayesian networks attached to each leaf node of the graph. Since exact posterior marginal and most probable explanation (MPE) inference is tractable on CNs, they often have better prediction accuracy as compared to probabilistic graphical models (PGMs) on which these inference tasks are intractable, even though the latter consistently have higher test-set log likelihood scores than the former (Rooshenas and Lowd, 2014; Rahman et al., 2019).

We investigate two classes of DCNs; the first (class) admits tractable exact inference algorithms and the second admits accurate particle filtering algorithms that generate samples from a high quality approximation of the posterior distribution. Both classes of DCNs use a templated representation having two components (time is discretized into slices): a prior distribution $P(\boldsymbol{X}^1)$ over all variables in time slice 1 and a conditional template for representing the transition distribution $P(\boldsymbol{X}^t|\boldsymbol{X}^{t-1})$, which has the same structure and parameters for all time slices $t > 1$. In DCNs, we represent $P(\boldsymbol{X}^1)$ using a CN and $P(\boldsymbol{X}^t|\boldsymbol{X}^{t-1})$ using a conditional cutset network template. The latter is based on a recently introduced conditionally tractable model called conditional cutset networks (CCN) (Rahman et al., 2019) that represents a potentially exponential number of CNs using calibrated classifiers (Niculescu-Mizil and Caruana, 2005)—specifically, one CN defined over $\boldsymbol{X}^t$ for each assignment of values to $\boldsymbol{X}^{t-1}$. We show that although exact inference is intractable in these DCNs, at each time slice t, we can generate samples from a good approximation to the posterior distribution over $\boldsymbol{X}^t$ given evidence at slice t and

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. Copyright 2021 by the author(s).

an assignment of values to all variables from time slice 1 to $t-1$. Since the posterior distribution is the ideal proposal distribution, DCNs facilitate efficient and potentially very accurate particle filtering algorithms.

To ensure tractability of exact inference, we put further restrictions on the CCN template. Specifically, we show that exact inference is tractable when the number of cutset networks in the CCN template is polynomial in the number of variables in $\boldsymbol{X}^{t-1}$ rather than exponential (as in conventional CCNs). We use this result to develop a new representation called dynamic AND/OR conditional cutset network.

This paper makes the following contributions:

- We develop a new probabilistic representation called dynamic cutset networks (DCNs) to represent and reason about uncertainty in temporal domains.
- We derive two classes of DCNs, one in which exact inference is tractable and another in which particle filtering algorithms have access to an accurate proposal distribution. These two classes of DCNs help us trade prediction accuracy with time.
- We experimentally evaluate DCNs on a wide variety of synthetic and benchmark temporal datasets comparing their generative and discriminative performance to dynamic Bayesian networks, dynamic sum product networks and long short term memory networks. Our experiments show that DCNs are superior to the competition, both in terms of discriminative and generative performance.

2 Related Work

Brandherm and Jameson (2004) attempted to introduce tractability into temporal models by compiling a DBN into a recursive network polynomial that could be represented as a Dynamic Arithmetic Circuit (DAC) (Darwiche, 2003). However, the compiled DAC can be exponential in size; so, there is no guarantee that inference will be tractable on the compiled models. Further, this method necessitates the learning of a DBN first, which is then compiled to a DAC.

(Peharz et al., 2014) introduced the HMM-SPN representation in which the emission distribution in a hidden Markov model (HMM), namely the conditional distribution of the observed variables given the hidden variable at each time slice is represented using a sum-product network (SPN) (Poon and Domingos, 2011). This architecture is limited in the sense that it does not fully exploit the power of SPNs in dynamic settings. To address these concerns, Melibari et al. (2016) proposed Dynamic Sum Product Networks (DSPNs), whose structure could be learned directly from data and that guaranteed tractability by fixing the size of the network.

DSPNs define a template network that, much like DBNs, can be unrolled to generate a large SPN that represents the joint distribution $P(\boldsymbol{X}^{1:T}, \boldsymbol{E}^{1:T})$ of a sequence $S = \{(\boldsymbol{x}^1, \boldsymbol{e}^1), \ldots, (\boldsymbol{x}^T, \boldsymbol{e}^T)\}$ where $\boldsymbol{X}^t = \{X_1^t, .., X_m^t\}$ and $\boldsymbol{E}^t = \{E_1^t, .., E_n^t\}$ are, respectively, the sets of query variables and evidence variables at time slice t. The structure of the template network is selected via local search on the space of possible template structures using the log-likelihood as a scoring function.

Note, however, that the result of the filtering query $P(\boldsymbol{X}^t|\boldsymbol{e}^{1:t})$ is dependent on the length of the sequence in DSPNs. In other words, $P(\boldsymbol{X}^t|\boldsymbol{e}^{1:t}, |S| = a) \neq P(\boldsymbol{X}^t|\boldsymbol{e}^{1:t}, |S| = b)$ where $|S|$ is the sequence length and $b > a > t$. The reason for this is that the template network imposes a structure that models the distribution in the reverse direction of time (since variables in the last time slice are near the root node of the directed acyclic graph associated with a DSPN). This results in different joint distributions for different sequence lengths. As a result, in DSPNs, learning a template structure that accurately represents the true joint distribution is challenging. In contrast, DCNs (our work) use both observed and hidden variables to represent the conditional distribution where the connections are always in the forward (chronological) direction. This property enables us to learn the conditional distributions more easily and improve inference accuracy.

Another disadvantage of DSPNs is that encoding prior knowledge in their SPN-based distributions can be challenging, which makes them unsuitable in cases where rich prior knowledge is already available from domain experts. Much like Bayesian networks, our framework easily allows the encoding of prior knowledge into our models.

3 Cutset Networks and Tractable Conditional Models

Let $\boldsymbol{X} = \{X_1, \ldots, X_n\}$ denote a set of discrete random variables and $\boldsymbol{x}$ a complete assignment to $\boldsymbol{X}$. Without loss of generality, we assume all variables are binary valued and take on values from the set $\{0, 1\}$. An AND/OR Cutset Network (AOCN) (Rahman and Gogate, 2016b; Rahman et al., 2014) is a tractable probabilistic model, specifically a model in which posterior marginal and most probable explanation queries can be answered in time that scales linearly with its size, that can be used for representing a joint probability distribution over $\boldsymbol{X}$. It is a rooted directed AND/OR graph $\mathcal{G}$ (Dechter and Mateescu, 2007) with a tree structured Bayesian network or Chow-Liu tree (Chow and Liu, 1968) attached to each leaf node of $\mathcal{G}$. For simplicity of exposition, throughout the paper, we focus on rooted AND/OR trees, but the methods we describe can be easily extended to rooted AND/OR graphs (cf. Dechter and Mateescu (2007); Mateescu and Dechter (2005); Rahman and Gogate (2016b)).

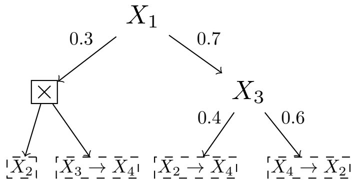

Figure 1: An AND/OR cutset network defined over the variables $\boldsymbol{X} = \{X_1, \ldots, X_4\}$. The variable nodes are OR nodes, the squares are AND nodes and the dotted rectangles are Chow-Liu trees.

An AND/OR tree (AOT) denoted by $\mathcal{T}$ over the set of variables $\boldsymbol{X}$ is a rooted tree (data structure) that can be recursively defined as follows:

- A leaf node labeled with the set $\boldsymbol{X}$ is an AOT.
- An OR node labeled with a variable $X_i \in \boldsymbol{X}$ and having two child AOTs, denoted by $\mathcal{T}_l$ and $\mathcal{T}_r$, and each defined over $\boldsymbol{X} \setminus \{X_i\}$ is an AOT.
- Given a partition $\{\boldsymbol{X}_1, \ldots \boldsymbol{X}_K\}$ of $\boldsymbol{X}$ (namely, $\boldsymbol{X}_1 \cap \ldots \cap \boldsymbol{X}_K = \emptyset$ and $\boldsymbol{X}_1 \cup \ldots \cup \boldsymbol{X}_K = \boldsymbol{X}$), an AND node having K child AOTs $\{\mathcal{T}_1 \ldots \mathcal{T}_K\}$ where each $\mathcal{T}_i$ is defined over the subset $\boldsymbol{X}_i$, is an AOT.

Each internal node in $\mathcal{T}$ is either a labeled OR node or an unlabeled AND node. AND nodes in $\mathcal{T}$ represent problem decomposition while each OR node represents conditioning over the variable that it is labeled with. Without loss of generality, given an OR node labeled by X_i, let the two child AOTs $\mathcal{T}_l$ and $\mathcal{T}_r$ denote the sub-trees obtained by conditioning on $X_i = 0$ and $X_i = 1$ respectively.

An AND/OR cutset network (AOCN), denoted by $\mathcal{C}$, is a triple $\langle \mathcal{T}, \Theta, \boldsymbol{B} \rangle$ where $\mathcal{T}$ is an AND/OR tree, $\Theta = \{\theta | \theta \in (0, 1)\}$ is a set of real numbers (parameters) and $\boldsymbol{B}$ is a set of tree Bayesian networks. Each parameter θ is associated with an arc from an OR node to its child nodes in $\mathcal{T}$ such that the parameters associated with the two child nodes $\mathcal{T}_l$ and $\mathcal{T}_r$ of the OR node sum to a 1. Each such parameter represents the conditional probability of the variable at an OR node taking on the values from $\{0, 1\}$ given the path (assignment of values to variables) from the root to the OR node. Each (tree) Bayesian network $\mathcal{B} \in \boldsymbol{B}$ is associated with a leaf node of $\mathcal{T}$ such that $\mathcal{B}$ represents a probability distribution over the set of variables that the leaf node is labeled with. Figure 1 shows an AOCN over the set of variables $\boldsymbol{X} = \{X_1, ..., X_4\}$.

Given an AOCN $\mathcal{C}$, the probability of a full assignment $\boldsymbol{x}$ to $\boldsymbol{X}$, denoted by $P_{\mathcal{C}}(\boldsymbol{x})$ can be computed as follows. Each full assignment induces a sub-tree $\mathcal{T}_{\boldsymbol{x}}$ of $\mathcal{T}$. Let $\Theta_{\boldsymbol{x}}$ and $\boldsymbol{B}_{\boldsymbol{x}}$ denote the set of parameters and Bayesian networks associated with the sub-tree $\mathcal{T}_{\boldsymbol{x}}$, then

$$P_{\mathcal{C}}(\boldsymbol{x}) = \left(\prod_{\theta \in \Theta_{\boldsymbol{x}}} \theta \right) \left(\prod_{\mathcal{B} \in \boldsymbol{B}_{\boldsymbol{x}}} P_{\mathcal{B}}(\boldsymbol{x}_{V(\mathcal{B})}) \right)$$

where $V(\mathcal{B})$ denotes the set of variables of $\mathcal{B}$, $\boldsymbol{x}_{V(\mathcal{B})}$ is the projection of $\boldsymbol{x}$ on $V(\mathcal{B})$ and $P_{\mathcal{B}}$ denotes the probability distribution represented by $\mathcal{B}$.

Posterior Marginal Estimation (MAR) and Most Probable Explanation (MPE) inference queries can be answered in time linear in the size of the AOCN (i.e. the number of nodes) (Rahman et al., 2014; Dechter and Mateescu, 2007). AOCNs often yield much smaller structured representations as compared to probabilistic graphical models (Pearl, 1988) because they take advantage of fine grained structural properties such as dynamic variable orders, context-specific independence, determinism and conditional independence (cf. Darwiche (2003); Chavira and Darwiche (2007); Gogate and Domingos (2010)).

3.1 Conditional Cutset Networks

Recently (Rahman et al., 2019) proposed a new representation called conditional cutset networks (CCNs) for compactly modeling the conditional distribution $P(\boldsymbol{Y}|\boldsymbol{X} = \boldsymbol{x})$ for all assignments $\boldsymbol{x}$. Similar to a AOCN, a CCN consists of a labeled AND/OR tree over $\boldsymbol{Y}$. However, the parameters Θ over the AND/OR tree and the conditional probabilities in each Bayesian network $\mathcal{B} \in \boldsymbol{B}$ are represented using weighting functions that take $\boldsymbol{x}$ as input and output a real number between 0 and 1. Rahman et al. (2019) proposed to use probabilistic classifiers called calibrated classifiers (e.g. logistic-regression, neural networks, random forests, etc.) as weighting functions. Calibrated classifiers are preferred over conventional classifiers because the former typically yield more accurate probability estimates than the latter.

An AOCN is obtained from a CCN by instantiating the weighting functions given an assignment $\boldsymbol{x}$. Thus, a CCN represents an exponentially large number of conditional tractable models.

Example 1. *Figure 2(a) shows a CCN over* $\boldsymbol{Y} = \{Y_1, \ldots, Y_4\}$ *and Figure 2(b) shows a AOCN obtained by instantiating the CCN with the assignment* $(X_1 = 0, X_2 = 1, X_3 = 0, X_4 = 1)$.

Note that a CCN allows exact tractable inference over the conditional multivariate joint distribution over $\boldsymbol{Y}$ only when $\boldsymbol{X}$ is fully-observed. In other words, exact inference is intractable when one or more variables in $\boldsymbol{X}$ are not observed or when we are interested in computing posterior marginals over $\boldsymbol{X}$. For example, $P(X_i|\boldsymbol{Y} = \boldsymbol{y})$ is intractable (NP-hard in general) to compute in CCNs under the assumption that a tractable representation for $P(\boldsymbol{X})$ is available. To this end, Rahman et al. (2019)

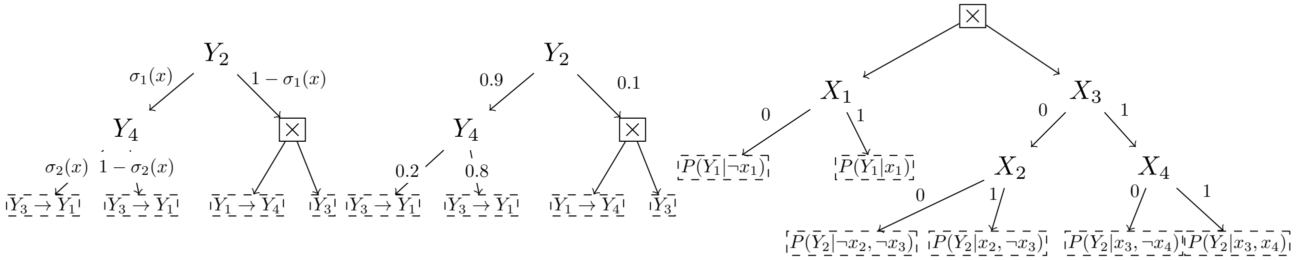

Figure 2: (a) A conditional cutset network (CCN) that models $P(\boldsymbol{Y}|\boldsymbol{X})$ for $\boldsymbol{Y} = \{Y_1, \ldots, Y_4\}$. The branch probabilities are calibrated classifiers $\sigma_i(\boldsymbol{x})$ where $\boldsymbol{x}$ is an assignment of values to all variables in the set $\boldsymbol{X}$. (b) AND/OR cutset network obtained from the CCN in Figure (a) given the assignment $X_1 = 0, X_2 = 1, X_3 = 0, X_4 = 1$ (c) An AOCCN that models $P(\boldsymbol{Y}|\boldsymbol{X})$ where $\boldsymbol{Y} = \{Y_1, Y_2\}$ and $\boldsymbol{X} = \{X_1, \ldots, X_4\}$. The leaves are conditional distributions of the form $P(\boldsymbol{Y}|\boldsymbol{x}')$ where $\boldsymbol{X}' \subseteq \boldsymbol{X}$. For example, instantiating the AOCCN given in (c) with $X_1 = 0, X_2 = 1, X_3 = 0, X_4 = 1$ will reduce the AOCCN to an AOCN that contains one AND node having two child leaf nodes that represent $P(Y_1|\neg x_1)$ and $P(Y_2|x_2, \neg x_3)$ respectively where $\neg x_i$ denotes the assignment $X_i = 0$ and x_i denotes the assignment $X_i = 1$.

proposed an approximate inference method based on Rao-Blackwellised importance sampling to compute the above.

Thus, an issue with using CCNs for modeling the transition distribution $P(\boldsymbol{X}^t|\boldsymbol{X}^{t-1})$ is that we cannot guarantee tractability of probabilistic inference. To address this problem, we present a new conditional representation based on AND/OR trees next.

3.2 AND/OR Conditional Cutset Networks

In this section, we introduce a new class of conditional models that represent the conditional distribution $P(\boldsymbol{Y}|\boldsymbol{X} = \boldsymbol{x})$ using an AND/OR tree $\mathcal{T}$ defined over $\boldsymbol{X}$ and a set $\boldsymbol{C}$ of AND/OR cutset networks defined over $\boldsymbol{Y}$ such that each leaf node of $\mathcal{T}$ is attached to a AND/OR cutset network from the set $\boldsymbol{C}$. We call these conditional models AND/OR tree based conditional cutset networks or AOCCNs in short. Formally, given two disjoint sets of variables $\boldsymbol{X}$ and $\boldsymbol{Y}$, an AOCCN $\mathcal{D}$ is a pair $\langle \mathcal{T}, \boldsymbol{C} \rangle$ where $\mathcal{T}$ is a labeled *AND/OR conditional tree* (defined below) over $\boldsymbol{X}$ and $\boldsymbol{Y}$ and $\boldsymbol{C}$ is a set of AOCNs defined over $\boldsymbol{Y}$ where each $\mathcal{C} \in \boldsymbol{C}$ is associated with a leaf node of $\mathcal{T}$. A labeled AND/OR conditional tree (AOCT) is recursively defined as follows:

- A leaf node labeled with the set $\boldsymbol{Y}$ is an AOCT.
- An OR node that is labeled with $X_i \in \boldsymbol{X}$ and has two child AOCTs, denoted by $\mathcal{T}_l$ and $\mathcal{T}_r$, and each defined over the pair $(\boldsymbol{X} \setminus \{X_i\}, \boldsymbol{Y})$ is an AOCT.
- Given a partition $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_K\}$ of $\boldsymbol{X}$ and a partition $\{\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_K\}$ of $\boldsymbol{Y}$, an AND node having K child AOCTs $\{\mathcal{T}_1, \ldots, \mathcal{T}_K\}$ where each $\mathcal{T}_i$ is defined over the pair $(\boldsymbol{X}_i, \boldsymbol{Y}_i)$ is an AOCT.

Given a AOCCN $\mathcal{D}$, the conditional probability $P_{\mathcal{D}}(\boldsymbol{y}|\boldsymbol{x})$ of the assignment $\boldsymbol{Y} = \boldsymbol{y}$ given $\boldsymbol{X} = \boldsymbol{x}$ can be computed as follows. Note that each assignment induces a sub-tree $\mathcal{T}_{\boldsymbol{x},\boldsymbol{y}}$ of $\mathcal{T}$. Let $\boldsymbol{C}_{\boldsymbol{x},\boldsymbol{y}}$ denote the set of AND/OR cutset networks at the leaves of the sub-tree $\mathcal{T}_{\boldsymbol{x},\boldsymbol{y}}$, then

$$P_{\mathcal{D}}(\boldsymbol{y}|\boldsymbol{x}) = \prod_{\mathcal{C} \in \boldsymbol{C}_{\boldsymbol{x},\boldsymbol{y}}} P_{\mathcal{C}}(\boldsymbol{y}_{V(\mathcal{C})}) \qquad (1)$$

where $V(\mathcal{C})$ denotes the set of variables of $\mathcal{C}$, $\boldsymbol{y}_{V(\mathcal{C})}$ is the projection of $\boldsymbol{y}$ on $V(\mathcal{C})$ and $P_{\mathcal{C}}$ denotes the probability distribution represented by $\mathcal{C}$.

Thus, given an assignment $\boldsymbol{x}$, a AOCCN yields an AND/OR tree cutset network over $\boldsymbol{y}$ (see Eq. (1)) and as a result is conditionally tractable in that given $\boldsymbol{x}$, both posterior marginal and most probable explanation queries can be answered in linear time in the size of the AOCCN.

Example 2. *Figure 2(c) shows a AOCCN over* $\boldsymbol{Y} = \{Y_1, Y_2\}$ *and* $\boldsymbol{X} = \{X_1, \ldots, X_4\}$. *The AOCCN yields a AOCN when instantiated with the assignment* $(X_1 = 0, X_2 = 1, X_3 = 0, X_4 = 1)$ *as described in the caption of Figure 2.*

4 Dynamic Cutset Networks

Consider a general discrete time state space model having a set of query (or hidden) variables $\boldsymbol{X}^t = \{X_1^t, .., X_n^t\}$ and a set of observed or evidence variables $\boldsymbol{E}^t = \{E_1^t, .., E_m^t\}$ at time slice t. For simplicity of exposition, we assume that the set of query and observed variables at each time slice is fixed. Similar to dynamic Bayesian networks (DBNs) (Murphy, 2002; Dean and Kanazawa, 1989), we will use the following template for representing the joint distribution over the state space:

1. A prior distribution $P(\boldsymbol{X}^1, \boldsymbol{E}^1)$, and
2. A transition distribution template representing $P(\boldsymbol{X}^t, \boldsymbol{E}^t|\boldsymbol{X}^{t-1}, \boldsymbol{E}^{t-1})$ for $t = 2, \ldots, T$ where T is the total number of time slices

We make two assumptions that are commonly used in temporal models. First, the variables at slice t are conditionally independent of all variables in all times slices before t given the variables in the previous time slice $t-1$ (1-Markov assumption). Second, we assume that the transition distribution is the same for all t (stationary assumption).

Given an integer $T > 1$, let $\boldsymbol{X}^{1:T} = \cup_{t=1}^{T} \boldsymbol{X}^t$ and $\boldsymbol{E}^{1:T} = \cup_{t=1}^{T} \boldsymbol{E}^t$ denote the set of query and evidence variables respectively from time slices 1 through T. Then, given the DCN template described above, the joint distribution over the set of variables $(\boldsymbol{X}^{1:T}, \boldsymbol{E}^{1:T})$ is given by

$$P(\boldsymbol{x}^{1:T}, \boldsymbol{e}^{1:T}) = P(\boldsymbol{x}^1, \boldsymbol{e}^1) \prod_{t=2}^{T} P(\boldsymbol{x}^t, \boldsymbol{e}^t | \boldsymbol{x}^{t-1}, \boldsymbol{e}^{t-1})$$

where $\boldsymbol{x}^{1:T}$ and $\boldsymbol{e}^{1:T}$ denote an assignment of values to all variables in the sets $\boldsymbol{X}^{1:T}$ and $\boldsymbol{E}^{1:T}$ respectively.

We consider two versions of DCNs. Our first version is obtained by using an AND/OR cutset network to model the prior distribution and the conditional cutset network (CCN) representation proposed in (Rahman et al., 2019) to represent the transition distribution template (see section 3.1). We call this version DCCN. Our second version is obtained by using an AND/OR cutset network to model the prior distribution and the AOCCN representation proposed in section 3.2 to represent the transition distribution template. We call this version DAOCCN.

In the next two sections, we describe inference algorithms for DAOCCNs and DCCNs respectively. We focus on the filtering task noting that the algorithms presented can be easily extended to other state space tasks such as smoothing and most probable explanation. Formally, the filtering task is defined as finding the following distribution:

$$P(\boldsymbol{x}^T | \boldsymbol{e}^{1:T}) \propto \sum_{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^{T-1}} P(\boldsymbol{x}^1, \boldsymbol{e}^1) \prod_{t=2}^{T} P(\boldsymbol{x}^t, \boldsymbol{e}^t | \boldsymbol{x}^{t-1}, \boldsymbol{e}^{t-1})$$

The above sum-product expression can be computed recursively using the forward algorithm (or the variable/bucket elimination algorithm (Dechter, 1999)), eliminating all variables at time slice $t-1$ before proceeding to time slice t. The recursion is given by:

$$\alpha(\boldsymbol{x}^t) = \sum_{\boldsymbol{x}^{t-1}} \alpha(\boldsymbol{x}^{t-1}) P(\boldsymbol{x}^t, \boldsymbol{e}^t | \boldsymbol{x}^{t-1}, \boldsymbol{e}^{t-1}) \qquad (2)$$

where $\alpha(\boldsymbol{x}^t) = P(\boldsymbol{x}^t, \boldsymbol{e}^{1:t})$ and $\alpha(\boldsymbol{x}^1) = P(\boldsymbol{x}^1, \boldsymbol{e}^1)$. The recursion can be solved using a message passing or a forward propagation algorithm where the message $\alpha(\boldsymbol{x}^t)$ is sent from the current time slice t to the next time slice $t+1$. In DBNs (Murphy, 2002), the message can be computed in time that scales exponentially in the size of the *forward interface*, which is the set of variables in time slice t that are connected via a directed edge to variables in time slice $t+1$. Thus, when the forward interface is bounded by a constant, the message passing algorithm runs in polynomial time. When, it is not bounded, we typically have to use approximate inference algorithms such as particle filtering (Doucet et al., 2001; Liu and Chen, 1998).

In the next section, we will show how a structured variant of the forward interface is bounded for DAOCCNs and as a result they admit tractable inference algorithms. For DCCNs, the forward interface equals all variables in the current time slice, namely it is not bounded by a constant and therefore we will derive efficient particle filtering algorithms.

4.1 Forward Algorithm for DAOCCNs

Next, we describe sufficient conditions for ensuring tractability of forward inference in DCNs. We begin by introducing two required definitions.

Definition (**Dominance**). *An AND/OR tree $\mathcal{T}_1$ dominates an AND/OR tree $\mathcal{T}_2$ such that the two trees are defined over the same set of variables if the set of context-specific conditional independencies represented by $\mathcal{T}_1$ are a subset of those represented by $\mathcal{T}_2$.*

In other words, $\mathcal{T}_1$ is an I-map of $\mathcal{T}_2$ (Pearl, 1988). A complete OR tree which represents no conditional independencies dominates all AND/OR trees while a AND/OR tree that has one AND node connected to n leaf nodes (n is the number of variables) is dominated by all other AND/OR trees.

Example 3. *The AOT associated with the AOCN given in Figure 1 represents the following context specific independencies: $\{X_3, X_4\}$ is conditionally independent of X_2 given $X_1 = 0$. It dominates the AOT which represents the following independence relation X_2, X_3 and X_4 are mutually independent of each other given $X_1 = 0$. The only difference between this AOT and the AOT given in Fig. 1 is that the AND node on the left sub-tree will have three child nodes labeled with X_2, X_3 and X_4 respectively.*

Definition (**Projection**). *Given an AND/OR tree $\mathcal{T}$ defined over a set of variables $\boldsymbol{X}$ and a subset $\boldsymbol{Y}$ of $\boldsymbol{X}$, the projection of $\mathcal{T}$ on $\boldsymbol{Y}$ is a AND/OR tree $\mathcal{T}_{\boldsymbol{Y}}$ such that $\mathcal{T}_{\boldsymbol{Y}}$ exactly captures all context-specific conditional independencies in $\mathcal{T}$ over $\boldsymbol{Y}$ and no more.*

The intuition behind developing the aforementioned definitions is the following. If an AND/OR tree $\mathcal{T}_1$ dominates another AND/OR tree $\mathcal{T}_2$, then it means that any probability distribution represented by $\mathcal{T}_2$ can also be represented using $\mathcal{T}_1$. Thus, by making appropriate parameter transformations, we can answer queries (exactly) by performing inference on $\mathcal{T}_1$ in lieu of $\mathcal{T}_2$. Notice that the recursion given in Eq. (2) involves multiplying a conditional distribution represented by the AOCCN, namely $P(\boldsymbol{x}^t, \boldsymbol{e}^t | \boldsymbol{x}^{t-1}, \boldsymbol{e}^{t-1})$ with a marginal distribution represented by a AOCN, namely $\alpha(\boldsymbol{x}^{t-1})$. If the AOCT projected on $\boldsymbol{X}^t$ in the AOCCN dominates the AOT of the AOCN, then the product of the two distributions, namely the joint can also be represented by a AOCN that has the same structure as the AOCT. In other words, we can appropriately transfer parameters (by performing inference) from the AOCN to the AOCCN to yield a AOCN that represents the joint distribution $P(\boldsymbol{X}^t, \boldsymbol{X}^{t-1}|\boldsymbol{e}^{1:t})$. Moreover, the size of the resulting AOCN will be bounded by the size of the AOCT and thus the recursion will run in time that scales linearly with the size of the representation.

Generalizing the above argument, we can show that:

Theorem 4. *The forward algorithm has linear time complexity in the size of the unrolled DAOCCN if the following condition is satisfied: the projection of the AOCT associated with the transition distribution on $\boldsymbol{X}^{t-1}$ dominates the AOT associated with the AOCN representation of $\alpha(\boldsymbol{x}^{t-1})$ for all $2 \leq t \leq T$ where T is the total number of time slices.*

Proof of Theorem 4 is included in the supplement.

An interesting corollary of the above theorem is that since an OR tree always dominates any AND/OR tree, when the AOCT is an OR tree, the filtering task can always be solved using the forward algorithm in linear time. Formally,

Corollary 5. *The forward algorithm has linear time complexity in the size of the unrolled DAOCCN if the AOCT associated with the transition distribution is an OR tree.*

When the condition in Theorem 4 is not satisfied, we can use the following (tractable) expectation propagation style approximation (Minka, 2001). Given a AOCN representing $\alpha(\boldsymbol{x}^{t-1})$, compute the parameter for each edge from the OR node to its child node in the AOCT associated with the transition distribution by performing marginal inference on the AOCN. With these parameters, the AOCT yields a AOCN that represents a joint distribution over the set $\boldsymbol{X}^t \cup \boldsymbol{X}^{t-1}$ of variables given evidence. Since the size of new AOCN is bounded by the size of the AOCT, forward inference remains tractable.

4.2 Forward Inference in DCCNs

The filtering task over DCCNs is intractable in general and can be approximately solved using the particle filtering algorithm (Doucet et al., 2001; Liu and Chen, 1998), a sequential importance sampling algorithm that generates samples from a proposal distribution and estimates the posterior distribution using a weighted average over the generated samples. The performance of the particle filtering algorithm is highly dependent on the quality of the proposal distribution; higher the quality better the estimate. It is known that the ideal proposal distribution is the posterior distribution $P(\boldsymbol{x}^{1:T}|\boldsymbol{e}^{1:T})$. Unfortunately, it is NP-hard to compute in DCCNs. Therefore, we propose to approximate the ideal proposal using the following

$$P(\boldsymbol{x}^{1:T}|\boldsymbol{e}^{1:T}) \approx P(\boldsymbol{x}^1|\boldsymbol{e}^1)\prod_{t=2}^{T} P(\boldsymbol{x}^t|\boldsymbol{x}^{1:t-1}, \boldsymbol{e}^{1:t})$$

Thus we propose to use evidence up to the current time slice to approximate the ideal proposal, namely we use $P(\boldsymbol{x}^t|\boldsymbol{x}^{1:t-1}, \boldsymbol{e}^{1:t})$ to approximate $P(\boldsymbol{x}^t|\boldsymbol{x}^{1:t-1}, \boldsymbol{e}^{1:T})$. It is easy to see that this approximation is likely to yield higher quality estimates as compared to conventional likelihood weighting approach which uses the prior distribution as the proposal. Note that $P(\boldsymbol{x}^t|\boldsymbol{x}^{1:t-1}, \boldsymbol{e}^{1:t})$ can be computed in linear time in DCCNs because they use CCNs, which are conditionally tractable models to represent the transition distribution.

4.3 Learning Dynamic Cutset Networks: Practical Considerations

The structure (and parameters) of the prior model $P(\boldsymbol{X}^1)$ and the transition distribution template $P(\boldsymbol{X}^t|\boldsymbol{X}^{t-1})$ can be learned from data in a straight forward manner by adapting the structure learning algorithms for learning unconditional as well as conditional models described in prior work (cf. (Rahman et al., 2014; Vergari et al., 2015; Di Mauro et al., 2015; Rahman et al., 2019; Mauro et al., 2017)). For lack of space, we describe them in the supplement.

To improve the practical performance of our learning algorithms, we propose to use mixtures of cutset networks as the prior distribution and mixtures of conditional cutset networks for modeling the transition distribution template. These mixture models can be learned from data either using the EM algorithm or via boosting and bagging (Rahman and Gogate, 2016a; Mauro et al., 2017). In our experiments, we used the EM algorithm (or the Baum-Welch algorithm). Sufficient statistics for this algorithm can be obtained by performing smoothing inference, namely computing $P(h^t|\boldsymbol{x}^{1:K})$ where H^t is the hidden variable at time slice t and K is the length of the sequence.

5 Experiments

We compare the performance of dynamic cutset networks to several state-of-the-art temporal models. We perform a number of experiments on both artificial and real-world problems. In both scenarios, we conduct two sets of experiments. The first set of experiments learns a model and computes the average test-set log-likelihood. The objective of these experiments is to see how well the models fit the data. The second set of experiments uses the learned models to evaluate their prediction/inference accuracy by comparing the average log-probability of evidence (evidence log-likelihood (ELL)) on the test sequences on a fraction of the state variables. The goal of these experiments is

Table 1: Average test set log-likelihood (LL) scores and average log-probability of evidence (ELL) on 25%, 50% and 75% of the variables on synthetic datasets. Each dataset *synthV* has *V* binary random variables, *1000V* training samples and *100V* test samples.

Model	**Dataset**					**Average**
	synth5	**synth10**	**synth15**	**synth20**	**synth25**	
			LL			
DSPN	-1.2917	-3.3442	-3.8867	-4.9907	-6.6554	-4.0337
CLDBN	-1.2726	-3.7315	-6.3186	-8.7462	-10.5283	-6.1194
LSTM	-0.3884	-1.4658	-1.3333	-2.3455	-4.0355	-1.9137
DAOCCN	**-0.3055**	-1.1981	-1.1123	-1.9994	-3.0268	-1.5284
DCCN	-0.3203	**-1.0519**	**-0.8071**	**-1.4780**	**-1.6129**	**-1.0541**
			ELL (25% Evidence)			
DSPN	-0.3429	-1.2035	-1.4965	-1.8447	-2.8677	-1.5511
CLDBN	-1.0275	-1.5729	-1.9624	-2.5828	-3.5345	-2.1360
LSTM	-0.6500	-2.5216	-2.5753	-4.4501	-5.1741	-3.0742
DAOCCN	**-0.1680**	**-0.6205**	**-0.4227**	**-0.1286**	**-0.9453**	**-0.4570**
DCCN	-1.0301	-1.7161	-1.6831	-2.4894	-2.3164	-1.8470
			ELL (50% Evidence)			
DSPN	-0.7077	-1.6886	-2.8986	-3.2960	-4.0901	-2.5362
CLDBN	-1.3741	-2.0797	-4.0102	-5.2187	-5.6115	-3.6588
LSTM	-1.3381	-1.7225	-3.6040	-4.2276	-8.2878	-3.8360
DAOCCN	**-0.2139**	**-0.8794**	**-0.6785**	**-1.3810**	**-1.9205**	**-1.0147**
DCCN	-0.9981	-1.5569	-1.7916	-2.2763	-2.9141	-1.9074
			ELL (75% Evidence)			
DSPN	-0.9282	-3.0691	-3.4214	-4.2832	-5.8286	-3.5061
CLDBN	-1.5773	-3.4166	-5.5778	-7.5724	-9.3606	-5.5009
LSTM	-1.4635	-1.9006	-2.9134	-3.0892	-4.9145	-2.8562
DAOCCN	**-0.2991**	**-1.0201**	**-0.9467**	**-1.6992**	**-2.0236**	**-1.1977**
DCCN	-1.0686	-1.5821	-1.3649	-2.1118	-2.1983	-1.6651

to measure the accuracy of inference since arbitrary non-evidence variables need to be summed out at each time-slice. A higher ELL score would indicate that the intermediate distributions obtained by summing out over the $\boldsymbol{X}^t$'s are being computed accurately. This makes it a good metric for measuring performance at prediction time.

In our experiments, we compare the performance of two classes of dynamic cutset networks: 1) A dynamic cutset network with a transition distribution modeled by a conditional cutset network with logistic regression classifiers; denoted by DCCN and 2) A DCN with transition distribution modeled using our proposed AND/OR conditional cutset network (see section 3.2); denoted by DAOCCNs. We compare both the modeling capacity and inference accuracy of the learned DCNs to the following diverse classes of models falling into both generative and discriminative categories: 1) dynamic sum-product networks (DSPNs) (Melibari et al., 2016; Kalra et al., 2018) 2) Tree Structured Dynamic Bayesian Networks (CLDBNs) and 3) Long-Short-Term-Memory (LSTMs) networks (Hochreiter and Schmidhuber, 1997). A tree structured dynamic Bayesian network is a dynamic Bayesian network in which both the initial distribution $P(\boldsymbol{X}^1)$ and the transition distribution $P(\boldsymbol{X}^t|\boldsymbol{X}^{t-1} = \boldsymbol{x}^{t-1})$ are tree structured Bayesian networks. Details on learning algorithms for these representations are provided in the supplement.

5.1 Artificial Problems

We generated training and test samples from complex, multi-modal, high-dimensional Dynamic Bayesian Networks (DBNs). Both the structure and parameters of the DBNs were generated randomly ensuring that the resulting graph is a DAG in both the initial state and transition distributions. From the generated DBNs, we sampled fixed length (length of 10) sequences as training and test data. Table 1 compares the average log-likelihood and average log-probability of evidence on sampled test data.

5.2 Real World Problems

We chose five real-world datasets *(#train, #test, #avg_train_length, #avg_test_length)* from the Time Series Classification website (Bagnall et al., 2017) and the UCI Repository (Dua and Graff, 2017) – *diabetes* $(56, 14, 425, 388)$, *racketsport* $(151, 152, 30, 30)$, *airquality* $(1, 1, 6379, 562)$, *handwriting* $(150, 850, 152, 152)$ and *japanvowels* $(512, 128, 16, 16)$. The datasets contain a large number of continuous variables which were first discretized using Symbolic Aggregate Approximation (SAX) (Lin et al., 2003) into bins of sizes 8, 16, 32 and 64. The only exception was the dataset *diabetes* which had all binary attributes except the class variable which was discretized into 8 bins. The discretized datasets were then binarized using log-encoding. Table 2 shows the average log-likelihood scores w.r.t. bin sizes 8 and 16 and the average log-probability of evidence achieved by the models on the test data. (Detailed experimental results and analysis is included in the supplementary material.)

5.3 Model's Fit to Data

We compare the average test set log-likelihood scores of the learned models on both the synthetic and real-world datasets in Tables 1 and 2 respectively. It is observed that DCNs outperform the competitors in terms of generalization accuracy. DCCNs with logistic regression classifier CPDs achieved the highest average test set log-likelihood scores in all cases followed by DAOCCNs. This shows that the DCN framework is flexible enough to allow for highly expressive representations that can model the data well. DSPNs generally performed better than CLDBNs while LSTMs outperformed both DSPNs and CLDBNs. This implies that the expressivity of dynamic models is significantly improved by latent variables. However, it is to be noted that learning DSPN structures is generally much harder in practice, which is probably why they are inferior to CLDBNs on real data.

5.4 Inference/Prediction Accuracy

From each test sequence we randomly set 25%, 50% and 75% of the variables in the domain as evidence variables

Table 2: Experimental results on real datasets. The metrics used are (1) Average test-set log-likelihood scores (LL) (2) Average test-set evidence log-likelihood scores for randomly chosen 25%, 50% and 75% variables. A *B/V* value is specified below each dataset where *B* is the total number of bins used for binarization and *V* is the total number of variables in the final binarized dataset.

Model	**Dataset**									**#wins**
	diabetes	**racketsport**		**airquality**		**handwriting**		**japanvowels**		
	8/23	**8/20**	**16/26**	**8/36**	**16/48**	**8/14**	**16/17**	**8/36**	**16/48**	
					LL					
DSPN	-3.4704	-11.7904	-15.8884	-20.9276	-28.5568	-8.3088	-9.8981	-22.5540	-30.9468	0
CLDBN	-4.2973	-10.0665	-14.0407	-17.5278	-25.7498	-3.5313	-5.0874	-17.4166	-25.6584	0
LSTM	-2.8913	-10.8268	-15.0085	-18.7659	-28.5243	-3.0549	-4.2957	-16.1458	-24.4584	0
DAOCCN	-2.9783	-9.4796	-13.4243	-18.2101	-26.4619	-3.1417	-4.7081	-17.7284	-25.9749	0
DCCN	**-2.7937**	**-9.3250**	**-13.3693**	**-14.1960**	**-22.2492**	**-2.9192**	**-4.2143**	**-14.3364**	**-23.9097**	**9**
					ELL (25% Evidence)					
DSPN	-1.9445	-3.0569	-4.3808	-4.5045	-6.4277	-2.4523	-2.9725	-5.8527	-7.9281	0
CLDBN	-2.0814	-3.5298	-4.8550	-5.0975	-7.1543	-1.8299	-2.6761	-4.3883	-6.3977	0
LSTM	-2.1654	-3.9112	-5.2564	-7.4319	-8.9709	-1.9939	-3.3351	-4.5935	-6.9295	0
DAOCCN	**-1.0421**	**-2.7403**	**-4.0347**	-4.5915	-6.6539	**-1.0132**	**-1.8496**	**-3.7340**	**-5.7508**	**7**
DCCN	-2.0704	-3.4099	-4.7212	**-4.1872**	**-6.1439**	-1.9845	-2.4061	-4.1333	-6.2628	2
					ELL (50% Evidence)					
DSPN	-2.4445	-6.1893	-8.2456	-8.2161	-11.8838	-4.0216	-4.9167	-11.4185	-15.6623	0
CLDBN	-2.8427	-6.5708	-8.5814	-9.2754	-13.3765	-3.3787	-4.4094	-8.3541	-12.4169	0
LSTM	-2.5282	-7.2410	-9.3001	-11.1287	-15.8784	-3.4381	-4.3014	-8.5292	-12.6195	0
DAOCCN	**-1.7502**	**-5.6880**	-7.7075	-8.8828	-11.6483	**-2.4452**	**-3.4493**	-7.7883	**-10.5403**	**5**
DCCN	-2.4330	-6.3249	**-6.1389**	**-7.2728**	**-11.0823**	-2.9491	-3.7885	**-7.3508**	-11.5892	4
					ELL (75% Evidence)					
DSPN	-2.6355	-8.6471	-12.7824	-13.7354	-19.4720	-6.7459	-8.0359	-16.5135	-22.7662	0
CLDBN	-3.7654	-9.0854	-12.9670	-13.0501	-20.1111	-4.2973	-5.8580	-11.3934	-18.1961	0
LSTM	-2.7393	-10.5267	-14.5238	-15.3121	-22.1672	-3.7216	-4.9373	-11.8173	-17.8363	0
DAOCCN	**-2.2617**	**-8.0616**	**-11.9545**	-12.8790	-18.7257	**-3.2241**	**-4.7035**	-10.9022	**-16.4302**	**6**
DCCN	-2.5129	-8.5676	-12.5698	**-10.8123**	**-16.6209**	-3.5826	-4.8709	**-10.7863**	-17.3695	3

while the remaining variables as unobserved/hidden and compute the average log-probability of evidence (ELL) for the entire sequence. We perform exact inference in DSPNs and DAOCCNs and use particle filtering with 500 particles in CLDBNs, LSTMs and DCCNs.

The ELL scores are shown in Tables 1 and 2 for synthetic and real world datasets respectively. DCNs in general have higher ELL scores than other state-of-the-art models. This outcome is expected as DCNs performed significantly better than other models in terms of LL scores. Although it was observed that DCCNs had higher LL scores than DAOCCNs, approximate inference in DCCNs resulted in lower ELL scores compared to DAOCCNs. Moreover, the ELL scores of DAOCCNs are significantly higher than DCCNs and others as more variables are unobserved. LSTM also does well in the modeling task but shows poor performance in the inference task. We speculate that this is because LSTMs use imputation methods which are known to perform poorly in presence of multi-modal distributions.

6 Conclusion

In this work, we developed a tractable framework for modeling temporal and sequential data called Dynamic Cutset Networks. We demonstrated that while the tractability of exact inference cannot be guaranteed for a transition distribution modeled using arbitrary AND/OR graph structure, efficient approximate inference can still be performed using particle filtering since the posterior probabilities can be computed exactly. We then proposed a new conditional model called dynamic AND/OR Conditional Cutset networks (DAOCCN) that incorporates certain constraints in its structure which, in turn, allows for exact inference in time that scales linearly in the size of the model. Finally, we empirically evaluated our models on several synthetic and real-world datasets. Our experiments clearly show the promise and efficacy of our algorithms.

Future work includes developing online learning algorithms; investigating the use of expectation propagation algorithms when the dominance constraint is violated; developing discriminative dynamic architectures; using our framework to model spatio-temporal data; etc.

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

A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31:606–660, 2017.

C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in bayesian networks. In *Proceedings of the Twelfth International Conference on Uncertainty in Artificial Intelligence*, page 115–123, 1996.

B. Brandherm and A. Jameson. An extension of the differential approach for bayesian network inference to dynamic bayesian networks. *International Journal of Intelligent Systems*, 19(8):727–748, 2004.

J. Carpenter, P. Clifford, and P. Fearnhead. Improved particle filter for nonlinear problems. *IEE Proceedings - Radar, Sonar and Navigation*, 146(1):2–7, 1999.

M. Chavira and A. Darwiche. Compiling Bayesian Networks Using Variable Elimination. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, pages 2443–2449, 2007.

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

C. K. Chow and C. N. Liu. Approximating Discrete Probability Distributions with Dependence Trees. *IEEE Transactions on Information Theory*, 14:462–467, 1968.

A. Darwiche. A Differential Approach to Inference in Bayesian Networks. *Journal of the ACM*, 50:280–305, 2003.

T. Dean and K. Kanazawa. A model for reasoning about persistence and causation. *Computational intelligence*, 5 (2):142–150, 1989.

R. Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113:41–85, 1999.

R. Dechter and R. Mateescu. AND/OR Search Spaces for Graphical Models. *Artificial Intelligence*, 171(2-3): 73–106, 2007.

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, 2015.

A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer, New York, 2001.

D. Dua and C. Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

V. Gogate and P. Domingos. Formula-Based Probabilistic Inference. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 210–219, 2010.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

A. Kalra, A. Rashwan, W.-S. Hsu, P. Poupart, P. Doshi, and G. Trimponias. Online structure learning for feed-forward and recurrent sum-product networks. In *Proceedings of the Thirty-Second International Conference on Neural Information Processing Systems*, pages 6944–6954, 2018.

J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the Eighth ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 2–11, 2003.

J. S. Liu and R. Chen. Sequential monte carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93(443):1032–1044, 1998.

R. Mateescu and R. Dechter. AND/OR cutset conditioning. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, pages 230–235, 2005.

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

M. Melibari, P. Poupart, P. Doshi, and G. Trimponias. Dynamic sum product networks for tractable inference on sequence data. In *Proceedings of the Eighth International Conference on Probabilistic Graphical Models*, pages 345–355, 2016.

T. P. Minka. Expectation Propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pages 362–369, 2001.

K. P. Murphy. *Dynamic Bayesian networks: representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

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.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, CA, 1988.

R. Peharz, G. Kapeller, P. Mowlaee, and F. Pernkopf. Modeling speech with sum-product networks: Application to bandwidth extension. In *Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3699–3703, 2014.

H. Poon and P. Domingos. Sum-Product Networks: A New Deep Architecture. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 337–346, 2011.

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

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, 2016b.

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 2014 Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 630–645, 2014.

T. Rahman, S. Jin, and V. Gogate. Cutset bayesian networks: a new representation for learning rao-blackwellised graphical models. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 5751–5757, 2019.

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.

A. Vergari, N. Di Mauro, and F. Esposito. Simplifying, regularizing and strengthening sum-product network structure learning. In *Proceedings of the 2015 Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 343–358, 2015.