Interpretability of Bayesian Network Classifiers: OBDD Approximation and Polynomial Threshold Functions

Karine Chubarian
University of Illinois at Chicago
kchuba2@uic.edu

György Turán *
University of Illinois at Chicago,
MTA-SZTE RGAI, Szeged
gyt@uic.edu

Abstract

Interpretability of learned models is considered from the point of view of a system using the model as a component, for tasks such as reasoning about its properties. We study the approximability of Bayesian network classifiers by Ordered Binary Decision Diagrams (OBDD). This generalizes an approach introduced by Chan and Darwiche. We show that for Tree Augmented Naive Bayes Classifiers (TAN) there is an efficiently computable approximation of polynomial size. Approximation error is measured with respect to the marginal distribution over the input variables generated by the classifier. This distribution can be approximated by a distribution of polynomial width. TAN can be represented by a quadratic threshold function of logarithmic path-width. The OBDD approximation algorithm applies to any such Boolean function and any distribution which can be approximated by a polynomial-width distribution.

1 Introduction

The broadening scope of machine learning applications brought to the fore several requirements beyond predictive accuracy for models obtained by machine learning. These aspects may have been around for a long time but the current scope of applications raises new aspects and motivates a reconsideration of previous approaches and the development of new ones. The requirements include interpretability, trust, fairness and safety.

Interpretability is a central aspect as it is a useful feature for the other requirements. Its long history is well illustrated by a sentence in the abstract of a 1996 paper by Sommer [28]: "This paper reopens the issue of understandability of induced theories, which, while prominent in the early days of ML, seems to have fallen out of favor in the sequel."

Interpretability is a many-faceted concept which is hard to formalize. Versions of interpretability include learning interpretable models, and *post hoc* approaches, such as developing interpretable global approximations to learned models and local explanations for individual inputs, e.g., for deep learning.

*Partially supported by the National Research, Development and Innovation Office of Hungary through the Artificial Intelligence National Excellence Program (grant no.: 2018-1.2.1-NKP-2018-00008).

Interpretability is usually meant from the point of view of the *human user*, for instance, in the case of a loan application. However, the "user" of a learned model can also be a *system*, which uses the learned model as a component. In that case one could specify the system requirements for the model such as the procedures which should be performed efficiently (e.g., building composite models or performing reasoning). Appropriate representations would then be those which provide a suitable data structure for the intended application.

In this paper we consider interpretability in the context of *Bayesian networks*. Although most interpretability research deals with deep learning, probabilistic models provide an interesting challenge as well.

Koller [8] noted that "Probabilistic models lie on a continuum between those that try to encode the domain structure in an interpretable way [...] and those that just try to capture the statistical properties of data." Interpretability of probabilistic graphical models appears to be simpler than that of deep neural networks, and therefore it may provide a first step towards understanding the general problem.

While Bayesian networks are considered to be interpretable in the sense of being informative about dependencies between the variables, they are not easily interpretable (either in the human or in the system sense) for inferring the mapping implicit in the network (e.g., for the classification of a disease based on symptoms) or for other inference purposes. Explainability of Bayesian networks has been studied for a long time (see, e.g., the survey of Lacave and Diez [19] and the recent work of Timmer *et al.* [31]).

A *Bayesian network classifier* has input variables $X_1, \ldots, X_n$ and a classification variable C such that C is a parent of every input variable. A special case is the Naive Bayes classifier, where there are no edges between the input variables. The Tree Augmented Naive Bayes Classifier (TAN), introduced by Friedman *et al.* [9], extends Naive Bayes by having an in-forest over the input variables.

Chan and Darwiche [2] proposed to represent Bayesian network classifiers by Ordered Binary Decision Diagrams (OBDD). As interpretability is so far an informal notion, the justification of particular frameworks is informal in general. For the relevance of OBDD one can provide two

arguments, which are still informal but arguably principled: *simulatability* (or *evaluatability*) and *"reason-ableness"*.

OBDD can be *evaluated* easily by a user, in the sense that the classification of an input can be determined by walking through the diagram *using the values of the input variables in a fixed order and in a single pass*. This resembles the way the simplest computational device, a finite automaton, evaluates an input string, and indeed, OBDD is a non-uniform generalization of finite automata. Bayesian network classifiers, neural networks, and even decision trees require more complicated evaluation procedures.

Also, corresponding to the system-based view of interpretability outlined above, OBDD is a basic *data structure* for representing Boolean functions. It provides a good compromise between expressibility and the tractability of implementing operations and answering queries. Thus OBDD is a "reason-able" representation for using a learned model as a component within a larger system.

Chan and Darwiche [2] give an algorithm to translate (or *compile*) a Naive Bayes classifier into an OBDD. The study of the *knowledge compilation* approach to interpretability is continued, e.g., in Shih *et al.* [26, 27].

As Naive Bayes classifiers are linear threshold functions, the translation of [2] is about turning linear threshold functions into OBDD. It was shown by Hosaka *et al.* [14] that this can be done with OBDD of size $O(2^{n/2})$, and Takenaga *et al.* [29] gave a $\Omega(2^{cn^{1-\varepsilon}})$ lower bound. Chan and Darwiche prove the upper bound and give further results.

In this paper we consider the construction of polynomial size OBDD *approximating* Bayesian network classifiers. One motivation is the lower bound of [29] showing that polynomial size exact representation is not possible even for Naive Bayes classifiers. Another motivation is that relaxing exact representation to approximation seems reasonable as learned models are supposed to be only approximately correct anyway.

We show that efficiently computable, polynomial size OBDD approximation is possible for Tree Augmented Naive Bayes Classifiers (TAN) having Boolean variables. The theorem gives a *fully polynomial approximation scheme (FPTAS)* in the context of knowledge compilation. *Approximate knowledge compilation* in this sense provides a formal approach to post-hoc interpretability.

One question that needs clarification is the distribution over the inputs used to measure the quality of approximation. A natural candidate here is the *input distribution* generated by the network, i.e., the marginal distribution over the input variables $X_1, \ldots, X_n$. This is the distribution used in the statement of Theorem 1.1 below.

Theorem 1.1. *For every Tree Augmented Naive Bayes Classifier (TAN) having n Boolean variables and d-bit conditional probabilities, and every $\varepsilon > 0$ there is an OBDD of size $poly(n, d, 1/\varepsilon)$ approximating the classifier with error at most ε with respect to the input distribution of the TAN. The OBDD can be constructed in time $poly(n, d, 1/\varepsilon)$.*

We use the fact that the Boolean function corresponding to a TAN can be represented as a *polynomial threshold function (PTF)*, in particular, as a *quadratic* threshold function (QTF) where the graph of the quadratic terms is a *forest*. Such PTF are called *forest QTF*. More generally, one can consider *path-width-k* QTF, where the graph of quadratic terms has path-width at most k. Naive Bayes is a special case, which corresponds to *linear* threshold functions. Forest QTF have logarithmic path-width. Thus an OBDD approximation of a TAN can be obtained as an OBDD approximation for forest QTF, or for QTF with logarithmic path-width.

We consider the class of probability distributions of *bounded width* for measuring approximation error. This is a class of syntactically defined discrete distributions, defined by Kamp *et al.* [16], and studied further in Gopalan *et al.* [11, 12] using a somewhat different definition. In these papers the class is called *small-space sources*. A bounded-width distribution is generated by a bounded-width OBDD with edge probabilities assigned to its edges. The probability of an input is the product of probabilities assigned to the edges of the path followed by the input.

In Theorem 1.2 we consider the OBDD approximability of QTF of bounded path-width, when the error is measured by a bounded-width distribution.

Theorem 1.2. *For every n-variable, path-width-k QTF with integer coefficients of at most d bits, every width-w distribution D over the input variables with d-bit probabilities and every $\varepsilon > 0$ there is an OBDD of size $poly(n, 2^k, d, w, 1/\varepsilon)$ approximating the QTF with error at most ε with respect to D. The OBDD can be constructed in time $poly(n, 2^k, d, w, 1/\varepsilon)$.*

It follows from Theorem 1.2 that every QTF of logarithmic path-width has a polynomial size OBDD approximation over any distribution of polynomial width. The construction generalizes the OBDD construction of Gopalan *et al.* [11, 12] for approximating the number of solutions of knapsack problems. The input distribution of a TAN is the mixture of two polynomial width distributions, and thus it is not of polynomial width in general. However, it can be approximated by a polynomial width distribution, and so applying Theorem 1.2 to this distribution implies Theorem 1.1.

The paper is structured as follows. After discussing related work in the next section, Sections 3-4 give preliminaries. Sections 5-6 describe exact, exponential-size OBDDs for a path-width-k QTF, in the second version extended with a bounded-width distribution to compute acceptance probabilities. Theorem 1.2 is proven in Section 7 by transforming the exact OBDD to a smaller approximate one. The approximability of the TAN input distribution, implying Theorem 1.1, is proven in Section 8.

2 Related work

Background on interpretability is given in Lipton [20] and Doshi-Velez and Kim [7]. Guidotti *et al.* [13] is a survey of interpretability providing a taxonomy of the various approaches. An early paper discussing theoretical aspects of interpretability is Golea [10]. Tickle *et al.* [30] is a survey of previous work on rule extraction.

The representational power of probabilistic classifiers is discussed by Jaeger [15] and Varando *et al.* [32]. Lacave and Diez [19] give a survey of explanation methods for Bayesian networks. Compilation results for Bayesian networks are described in Darwiche [3]. A general overview on knowledge compilation is given by Darwiche and Marquis [4]. Del Val [6] discusses the approximation quality of knowledge compilation using Horn LUB. Tractable operations and complexity aspects of OBDD are described in the monograph of Wegener [33].

Complexity aspects of polynomials with bounded tree-width term structure are studied by Makowsky and Meer [21, 22]. Amarilli *et al.* [1] give a characterization of OBDD representability of monotone CNF expressions in terms of path-width.

Gopalan *et al.* [11, 12] gave an approximation algorithm for counting the number of knapsack solutions using OBDD. Levelwise monotonicity of OBDD is used by Meka and Zuckerman [23] in the context of pseudo-random generation. The approximation algorithm of De and Servedio [5] for low-degree PTF uses a different approach. While multiplicative approximation is possible in the linear case, for degree at least two only additive approximation can be expected (see, e.g., [5]).

3 Preliminaries

Bayesian network classifiers

We use X_i, x_i, resp., a_i, to denote binary random variables, Boolean variables, resp., Boolean constants[1]. A *Bayesian network classifier* N is a directed acyclic graph (DAG) over binary *input variables* $X_1, \ldots, X_n$ and a binary *classifier variable* C, with local conditional probabilities specified for each vertex. It is assumed that (C, X_i) is an edge for every $i = 1, \ldots, n$. Let $\Pi_i = \{j : X_j \text{ is a parent of } X_i\}$ be the set of parents of X_i other than C. The *family* of i is $\{i\} \vee \Pi_i$. Let $d_N = 1 + \max_i |\Pi_i|$ be the maximal size of families in N, referred to as the *degree* of the Bayesian network. The local conditional probabilities are $p^0_c = P_N(C = c)$ and $p^i_{(a_i, a_{\Pi_i}, c)} = P_N(X_i = a_i \,|\, X_{\Pi_i} = a_{\Pi_i}, C = c)$ for $i = 1, \ldots, n$. Here X_{Π_i} is the vector formed by the components X_j of $X = (X_1, \ldots, X_n)$ such that $j \in \Pi_i$, and similarly for a_{Π_i}.

The directed acyclic graph (DAG) of the Bayesian network classifier over the X-nodes is denoted by G_N. In a *Naive Bayes Classifier* G_N is an empty graph. In a *Tree Augmented Naive Bayes Classifier (TAN)* the DAG is an in-forest with edge set E_N. Thus every node has at most one parent.

The joint distribution of the variables is

$$P_N(X_1 = a_1, \ldots, X_n = a_n, C = c) = p^0_c \prod_{i=1}^{n} p^i_{(a_i, a_{\Pi_i}, c)}.$$

The marginal distribution over the input variables, also referred to as the *input distribution*, is

$$P_{N,X}(X_1 = a_1, \ldots, X_n = a_n) = \\ = \textstyle\sum_{c=0}^{1} P_N(X_1 = a_1, \ldots, X_n = a_n, C = c).$$

We also use the simpler notations such as $P_N(a_1, \ldots, a_n, c)$ and $P_{N,X}(a_1, \ldots, a_n)$.

The Bayesian network classifier corresponding to N is a Boolean function $f_N(x_1, \ldots, x_n)$ where $f_N(a_1, \ldots, a_n) = 1$ iff

$$P_N(a_1, \ldots, a_n, 1) \geq P_N(a_1, \ldots, a_n, 0).$$

OBDD and GOBDD

An *ordered binary decision diagram (OBDD)* over Boolean variables $x_1, \ldots, x_n$ computes a Boolean function f. An OBDD is a DAG with two sinks labeled 0 and 1, and the other nodes labeled with variables. The DAG is assumed to be layered, with directed edges going from a layer to the next layer, and sinks on the last layer. There are $n + 1$ layers and a permutation $\pi(i)$ of $[n]$ such that nodes on the i'th layer are labeled with variable $x_{\pi(i)}$. On the first layer there is a single start node labeled $x_{\pi(1)}$. Every non-sink node has two outgoing edges, labeled with 0, resp., 1. For every truth assignment $x = (x_1, \ldots, x_n)$, $f(x)$ is the label of the sink reached by following edge labels corresponding to the bits in x, evaluated in the order given by the labels of the layers. The *width* of an OBDD is the maximal number of nodes in a layer.

A *generator OBDD (GOBDD)* D generates a probability distribution over $\{0, 1\}^n$. It is similar to an OBDD, except edges are also labeled with probabilities, and there is a single sink. A probability p_u is associated with every non-sink vertex u, and the 0-edge (resp. 1-edge) leaving u is labeled $p^0_u = p_u$ (resp., $p^1_u = 1 - p_u$). For every truth assignment $x = (x_1, \ldots, x_n)$, the GOBDD determines a path from the source to the sink, and $P_D(x)$ is the product of edge probabilities along the path. Product distributions are generated by width-one GOBDD.

The definitions of Kamp *et al.* [16] and Gopalan *et al.* [11, 12] differ in that [11, 12] use the definition above, while [16] allows several 0-edges or 1-edges leaving a vertex, and assigns a distribution to edges leaving a vertex.

4 Polynomial threshold function representation of Bayesian network classifiers

A *polynomial threshold function (PTF)* is a Boolean function of the form $sgn(p(x_1, \ldots, x_n))$, where sgn is the sign function and p is a multilinear polynomial. The degree of the representation is the degree of p, i.e., the maximal number of variables in a term. In particular, a *quadratic* threshold function (QTF) is specified by a quadratic polynomial

$$q(x_1, \ldots, x_n) = \sum_{(i,j) \in E} \beta_{ij} x_i x_j + \sum_{i=1}^{n} \alpha_i x_i + \gamma.$$

The *term-graph* of the quadratic polynomial q is the undirected graph $G_q = ([n], E)$. Thus the term-graph of a *linear* threshold function is the empty graph.

[1] For Bayesian networks, constants are usually denoted by x. We distinguish between random variables and Boolean variables.

Given an undirected graph $G = (V, E)$, a *tree-decomposition* of G is given by a tree T and "bags" $V_t \subseteq V$ for every node t of T, such that for every edge (u, v) of G there is a bag containing both u and v, and for every node v of G the nodes of T with bags containing v form a subtree of T. The width of a tree decomposition is the maximal bag size minus one, and the *tree-width* of G is the minimal width of a tree-decomposition of G. Trees are connected graphs of tree-width one. The *path-width* of a graph is defined the same way with trees restricted to paths. The tree-width (resp., path-width) of a QTF is the tree-width (resp., path-width) of its term-graph G_q. A QTF function which has a forest term-graph is called a *forest* QTF.

Varando *et al.* [32] formulated the representability of Bayesian network classifiers as PTF for the general case of categorical distributions. They assumed that all conditional probabilities are non-zero.

Proposition 4.1. *(see, e.g., Varando* et al. *[32]) Let N be a Bayesian network classifier with non-zero conditional probabilities. The classifier f_N is a degree-d_N PTF such that every term is a subset of a family.*

Proof. Let $I_a(x)$ be the binary indicator function, i.e., $I_1(x) = x$ and $I_0(x) = 1 - x$. It holds that $P_N(C = c) = (p_0^0)^{1-c}(p_1^0)^c = (p_0^0)^{I_0(c)}(p_1^0)^{I_1(c)}$ and

$$P_N(X_i = x_i \,|\, X_{\Pi_i} = x_{\Pi_i}, C = c) = \\ = \prod_{(a_i, a_{\Pi_i}, c)} p^i_{(a_i, a_{\Pi_i}, c)}{}^{I_{a_i}(x_i) \prod_{j \in \Pi_i} I_{a_j}(x_j)}.$$

Taking logarithms

$$\log P_N(X_i = x_i \,|\, X_{\Pi_i} = x_{\Pi_i}, C = c) = \\ = \sum_{(a_i, a_{\Pi_i}, c)} \log(p^i_{(a_i, a_{\Pi_i}, c)}) \cdot I_{a_i}(x_i) \prod_{j \in \Pi_i} I_{a_j}(x_j).$$

Thus

$$\log P_N(X_1 = x_1, \ldots, X_n = x_n, C = c) = \log p^0_c + \\ + \sum_{i=1}^{n} \sum_{(a_i, a_{\Pi_i}, c)} \log(p^i_{(a_i, a_{\Pi_i}, c)}) \cdot I_{a_i}(x_i) \prod_{j \in \Pi_i} I_{a_j}(x_j)$$

and the claim follows by the definition of f_N. □

Korach and Solel [18] showed that the path-width of every n-vertex tree T is at most $(2/\log 3) \log n$. Thus for TAN the following holds.

Corollary 4.2. *The classifier $f_N(x_1, \ldots, x_n)$ represented by a TAN N with non-zero conditional probabilities is a forest QTF, and thus a QTF of path-width at most $(2/\log 3) \log n$. It can be written as*

$$sgn\left(\sum_{(i,j) \in E_N} \beta_{ij} x_i x_j + \sum_{i=1}^{n} \alpha_i x_i + \gamma\right). \qquad (1)$$

Zero handling and precision

Proposition 4.1 and Corollary 4.2 remain valid in the general case as well, as replacing zero conditional probabilities by sufficiently small numbers (chosen depending on the non-zero entries) gives the same classifier.

We consider d-bit conditional probabilities, and therefore we need bounds for the coefficients and we also need to consider the complexity of computing the coefficients. The coefficients are obtained by taking logarithms. It can be shown that $O(nd)$ bits precision and $poly(n, d)$ time is sufficient in the case of non-zero probabilities. Thus one can assume that the coefficients are $O(nd)$ bit integers in the non-zero case. In the general case it then follows that $poly(n, d)$ precision and time is sufficient to handle the small numbers replacing zeros. Thus for the remainder of the paper we assume that *Bayesian network classifiers have non-zero conditional probabilities*.

The construction of approximate OBDD for TAN can be implemented using the original product form without taking logarithms and handling information about zero probabilities in the joint and input distributions symbolically (which could then also be used for reasoning). These modifications improve efficiency and interpretability.

5 The exact OBDD for bounded path-width QTF

In this section we describe an OBDD B to compute a path-width-k QTF

$$f(x_1, \ldots, x_n) = sgn\left(\sum_{(i,j) \in E} \beta_{ij} x_i x_j + \sum_{i=1}^{n} \alpha_i x_i + \gamma\right),$$

where the coefficients are integers, and the sum of their absolute values is at most W. The OBDD is an extension of the standard OBDD for linear threshold functions, keeping track of partial sums.

Given an ordering of the vertices of the term-graph $G([n], E)$ of the QTF, let

$$H_\ell = \{j : j \leq \ell \text{ and } (j, k) \in E \text{ for some } k > \ell\}.$$

For later reference, let us note that

$$H_{\ell+1} \subseteq H_\ell \cup \{\ell + 1\}, \qquad (2)$$

as $\ell + 1$ is the only additional candidate for consideration when forming $H_{\ell+1}$. The *vertex separation number* of G is the minimum of $\max\{|H_\ell| : 1 \leq \ell \leq n\}$ over all linear orderings of V. Kinnersley [17] proved that the vertex separation number of a graph equals its path-width. For trees, an optimal vertex ordering can be found efficiently (Scheffler [24]). We use this ordering of the variables for B.

Description of B

Nodes are of the form $v^\ell_{s,b}$, where $1 \leq \ell \leq n + 1$ is the level of the node, $-W \leq s \leq W$ and $b \in \{0, 1\}^{|H_{\ell-1}|}$. The start node on level 1 is $v^1_{0,\varnothing}$.

The children of a node $v^\ell_{s,b}$ are $v^{\ell+1}_{s_0,b_0}$ and $v^{\ell+1}_{s_1,b_1}$ corresponding to the evaluations $x_\ell = 0$ and $x_\ell = 1$. These are determined as follows: $s_0 = s + h_\ell(0) = s$ and $s_1 = s + h_\ell(1)$, where

$$h_\ell(x_\ell) = \left(\alpha_\ell + \sum_{\ell:(j,\ell) \in E, j < \ell} \beta_{j\ell} b_j\right) \cdot x_\ell.$$

Note that the j values involved in the sum are contained in $H_{\ell-1}$, so the bits b_j are those in b. The bit assignments b_0, b_1 of the children correspond to level update as in (2), i.e., bits corresponding to components which are not in H_ℓ are deleted and if $\ell \in H_\ell$ then x_ℓ is added.

On the last level there are nodes of the form $v^{n+1}_{s,\varnothing}$. Nodes with $s < 0$ (resp., $s \geq 0$) are replaced by the sink labeled 0 (resp., 1).

Lemma 5.1. *The OBDD B computes f.*

Note that B may be of exponential size, but it will be used in a "virtual" manner for the construction of a compressed, approximate version.

For a node $v^\ell_{s,b}$ on level ℓ, let the *acceptance set* $A^\ell_{s,b}$ be the set of partial truth assignments to variables $x_\ell, \ldots, x_n$ which are accepted when applied from $v^\ell_{s,b}$. The acceptance set for the 0-sink is $\varnothing$, the acceptance set for the 1-sink is the empty string, and for $\ell \leq n$ it holds that $A^\ell_{s,b} = 0 \cdot A^\ell_{s_0,b_0} \cup 1 \cdot A^\ell_{s_1,b_1}$, where $\cdot$ denotes concatenation. The sets $A^\ell_{s,b}$ have the following monotone property.

Lemma 5.2. *Let $v^\ell_{s_1,b}$ and $v^\ell_{s_2,b}$ be nodes such that $s_1 < s_2$. Then $A^\ell_{s_1,b} \subseteq A^\ell_{s_2,b}$.*

6 The exact OBDD for bounded path-width QTF with acceptance probabilities

The construction of the OBDD B in Section 5 is now extended to incorporate a width-w GOBDD D. It is analogous to the product of automata or OBDD.

Description of $B \times D$

Nodes are of the form $v^\ell_{s,b,u}$, where the additional component u is a node of D on level ℓ. The start node on level 1 is $v^1_{0,\varnothing,u_{start}}$, where u_{start} is the start node of D.

The children of a node $v^\ell_{s,b,u}$ are $v^{\ell+1}_{s_0,b_0,u_0}$ and $v^{\ell+1}_{s_1,b_1,u_1}$, where the additional parameters u_0, u_1 are the children of u in D. On the last level there are nodes of the form $v^{n+1}_{s,\varnothing,u_{sink}}$, where u_{sink} is the sink of D. Nodes with $s < 0$ (resp., $s \geq 0$) are replaced by the sink labeled 0 (resp., 1).

Acceptance sets are like in the previous section, and as acceptance does not depend on the u-component, it holds that $A^\ell_{s,b,u} = A^\ell_{s,b}$.

Acceptance probability $P_D(v^\ell_{s,b,u})$ is the probability that a random truth assignment to the variables $x_\ell, \ldots, x_n$ is accepted when started from $v^\ell_{s,b,u}$ when probabilities are evaluated in D starting from u. The acceptance probability of the 0-sink is 0, and that of the 1-sink is 1. For $\ell \leq n$ it holds that

$$P_D(v^\ell_{s,b,u}) = p^0_u P_D(v^{\ell+1}_{s_0,b_0,u_0}) + p^1_u P_D(v^{\ell+1}_{s_1,b_1,u_1}).$$

Thus acceptance probabilities can be computed in a bottom-up manner. The analog of Lemma 5.2 is as follows.

Lemma 6.1. *Let $v^\ell_{s_1,b,u}$ and $v^\ell_{s_2,b,u}$ be nodes such that $s_1 < s_2$. Then*
a) $A^\ell_{s_1,b,u} \subseteq A^\ell_{s_2,b,u}$,
b) $P_D(v^\ell_{s_1,b,u}) \leq P_D(v^\ell_{s_2,b,u})$.

7 Proof of Theorem 1.2

The approximate OBDD $\widetilde{B \times D}$ satisfying the requirements is constructed by compressing $B \times D$, processing its levels from the bottom up and within each level from left to right. We set $W = n\, 2^{d+2}$. Each level of $B \times D$ is partitioned into *blocks*

$$V^\ell_{b,u} = \{v^\ell_{s,b,u} : -W \leq s \leq W\}.$$

In each block a polynomial size set of *distinguished nodes*

$$S^\ell_{b,u} \subseteq V^\ell_{b,u}$$

is selected. These are the nodes of $\widetilde{B \times D}$. The children of distinguished nodes are modified to be the closest distinguished node with a larger s-value. The processing of a level also includes the calculation of modified acceptance probabilities $\tilde{P}_D(v^\ell_{s,b,u})$, which are the acceptance probabilities of the modified children. $B \times D$ is used implicitly, by doing binary search on the s values.

Description of $\widetilde{B \times D}$

The construction of $\widetilde{B \times D}$ uses the procedure $BUILD(v^\ell_{s,b,u})$.

Procedure $BUILD(v^\ell_{s,b,u})$
if $\ell = n$ **then** children and acceptance probabilities are unchanged
else
modify the children: the new 0-child is $v^{\ell+1}_{s',b_0,u_0}$, resp. the new 1-child is $v^{\ell+1}_{s'',b_1,u_1}$, where

$$s' = \min\{t : v^{\ell+1}_{t,b_0,u_0} \in S^{\ell+1}_{b_0,u_0}, s_0 \leq t\},$$
$$s'' = \min\{t : v^{\ell+1}_{t,b_1,u_1} \in S^{\ell+1}_{b_1,u_1}, s_1 \leq t\},$$

compute the new acceptance probability:

$$\tilde{P}_D(v^\ell_{s,b,u}) = p^0_u \tilde{P}_D(v^{\ell+1}_{s',b_0,u_0}) + p^1_u \tilde{P}_D(v^{\ell+1}_{s'',b_1,u_1}).$$

Applying the procedure $BUILD$ repeatedly, we find the set of distinguished vertices $S^\ell_{b,u}$ with s-values $s^*_0 < s^*_1 < \ldots$, where $s^*_0 = -W$ and

$$s^*_{i+1} = \begin{cases} s^*_i + 1 \text{ if } \tilde{P}_D(v^\ell_{s^*_i+1,b,u}) > (1+\delta)\tilde{P}_D(v^\ell_{s^*_i,b,u}) \\ \max\{t : \tilde{P}_D(v^\ell_{t,b,u}) \leq (1+\delta)\tilde{P}_D(v^\ell_{s^*_i,b,u})\} \text{ else,} \end{cases} \quad (3)$$

where δ is to be specified later. After all the distinguished sets are constructed, there may be nodes remaining which are not reachable from the start node; they are removed by one pass from the start node.

Note that it follows by induction from the definition of children that $\tilde{P}(v^\ell_{s,b,u})$ is monotonic in s. Let $\tilde{A}^\ell_{s,b,u}$ denote the acceptance sets in $\widetilde{B \times D}$.

Lemma 7.1. *For any level $\ell \le n$ and any distinguished node $v^\ell_{s,b,u} \in S^\ell_{b,u}$ it holds that*

a) $A^\ell_{s,b,u} \subseteq \tilde{A}^\ell_{s,b,u}$,

b) $P_D(v^\ell_{s,b,u}) \le \tilde{P}_D(v^\ell_{s,b,u}) \le (1+\delta)^{n-\ell} P_D(v^\ell_{s,b,u})$.

Proof. Part *a)* follows from Lemma 6.1 by noting that the s-values are always increased. For part *b)*, the first inequality follows from *a)*. For the second inequality we claim that

$$\tilde{P}_D(v^{\ell+1}_{s',b_0,u_0}) \le (1+\delta)\tilde{P}_D(v^{\ell+1}_{s_0,b_0,u_0}), \qquad (4)$$

where s_0 is the 0-child of s in $B \times D$, and s' is its new 0-child found by procedure $BUILD$. This holds by definition if $v^{\ell+1}_{s',b_0,u_0}$ is included in $S^{\ell+1}_{b_0,u_0}$ using the second case in (3). Otherwise $s_0 = s'$, so the claim holds again. The analogous statement holds for 1 instead of 0 as well.

Using the definition of $\tilde{P}_D$, (4) and induction

$$\begin{aligned}\tilde{P}_D(v^\ell_{s,b,u}) &= p^0_u \tilde{P}_D(v^{\ell+1}_{s',b_0,u_0}) + p^1_u \tilde{P}_D(v^{\ell+1}_{s'',b_1,u_1}) \le \\ &\le (1+\delta)(p^0_u \tilde{P}_D(v^{\ell+1}_{s_0,b_0,u_0}) + p^1_u \tilde{P}_D(v^{\ell+1}_{s_1,b_1,u_1})) \le \\ &\le (1+\delta)^{n-\ell}(p^0_u P_D(v^{\ell+1}_{s_0,b_0,u_0}) + p^1_u P_D(v^{\ell+1}_{s_1,b_1,u_1})) = \\ &= (1+\delta)^{n-\ell} P_D(v^\ell_{s,b,u}).\end{aligned}$$

□

By Lemma 7.1 *b)* choosing the value $\delta = O(\varepsilon/n)$ gives that for the root $v^1_{0,\varnothing,\varnothing}$ it holds that $P(v^1_{0,\varnothing,\varnothing}) \le \tilde{P}(v^1_{0,\varnothing,\varnothing}) \le (1+\varepsilon)P(v^1_{0,\varnothing,\varnothing})$. Lemma 7.1 *a)* implies that $\widetilde{B \times D}$ has one-sided error at most ε w.r.t. D.

The monotonicity of the acceptance probabilities implies that the next distinguished node s^*_{i+1} can be found by binary search over the interval $[s^*_i, W]$, calling the procedure $BUILD$ in each step to compute $\tilde{P}$. Binary search is polynomial in the parameters. As s-values of distinguished nodes increase exponentially, the number of distinguished nodes is also polynomial, and thus the size of the OBDD constructed and the running time of the algorithm are both polynomial.

8 Approximation of the input distribution and the proof of Theorem 1.1

In this section we use Theorem 1.2 to prove Theorem 1.1. Given a TAN N, we again assume that $X_1, \ldots, X_n$ is an optimal ordering and consider the variable ordering $\pi = (C, X_1, \ldots, X_n)$. Adding C increases path-width by one. We first show that the *joint* distribution P_N has polynomial width.

Theorem 8.1. *The* joint *distribution P_N with ordering π has width $O(n^{1.6})$.*

Proof. Consider a GOBDD generating P_N and let $C = c, X_1 = a_1, \ldots, X_{\ell-1} = a_{\ell-1}$ be a partial truth assignment. Then the product of edge weights along the corresponding path is $P_N(c, a_1, \ldots, a_{\ell-1})$. Thus the probability assigned to the a_ℓ-child of that node has to be $P_N(a_\ell \,|\, c, a_1, \ldots, a_{\ell-1})$. The construction of the GOBDD D' generating P_N is based on the following lemma.

Lemma 8.2.

$$\begin{aligned}&P(X_\ell = a_\ell \,|\, C = c, X_1 = a_1, \ldots, X_{\ell-1} = a_{\ell-1}) = \\ &= P(X_\ell = a_\ell \,|\, C = c, X_j = a_j \textit{ for every } j \in H_{\ell-1}).\end{aligned}$$

Proof. It has to be shown that $H_{\ell-1} \cup \{C\}$ is a d-separator of $\{\ell\}$ and $\{1, \ldots, \ell-1\} \setminus H_{\ell-1}$, i.e., every path between the two sets is blocked by $H_{\ell-1} \cup \{C\}$. If C is on the path then the valve containing it is not convergent (using the terminology of Darwiche [3]), and hence the path is blocked by C. Other paths cannot contain convergent valves. Hence it is sufficient to show that such paths must contain a vertex from $H_{\ell-1}$. This follows directly from the definition of $H_{\ell-1}$. □

Description of D'

From the start node C on level 0 there are two edges to the first level corresponding to x_1, with probabilities $P_N(C=0)$, resp., $P_N(C=1)$. On level ℓ evaluating x_ℓ there are $2^{|H_{\ell-1}|+1}$ nodes corresponding to truth assignments to C and to the variables x_j for $j \in H_{\ell-1}$. The nodes are denoted u^ℓ_b, where $b \in \{0,1\}^{|H_{\ell-1}|+1}$ is such a truth assignment.

The a_ℓ-child is $x_{\ell+1}$-node $u^{\ell+1}_{b_{a_\ell}}$. Here b_0 (resp., b_1) is the truth assignment obtained by switching from $H_{\ell-1}$ to H_ℓ, using $x_\ell = a_\ell$ if necessary. By (2), the truth assignment b and the value of x_ℓ (evaluated at u^ℓ_b) determine these nodes. The probability assigned to edge $(u^\ell_b, u^{\ell+1}_{b_{a_\ell}})$ is given by $P_N(X_\ell = a_\ell \,|\, C = c, X_j$ as in b for $j \in H_{\ell-1})$.

The conditional probabilities can be computed efficiently using standard methods. □

As the input distribution $P_{N,X}(a_1, \ldots, a_n)$ can be written as

$$\sum_{c=0}^{1} P_N(a_1, \ldots, a_n \,|\, c)\, P_N(c),$$

it is a mixture of two polynomial-width distributions. It is not necessarily of polynomial width itself (see Shen *et al.* [25]). On the other hand, we now show that it can be approximated by a polynomial-width distribution.

Note that the edge probabilities in a GOBDD for the input distribution can be written as

$$\begin{aligned}&P_N(a_\ell \,|\, a_1, \ldots, a_{\ell-1}) = \\ &= \frac{P_N(a_1,\ldots,a_\ell)}{P_N(a_1,\ldots,a_{\ell-1})} = \frac{\sum_{c=0}^1 P_N(a_1,\ldots,a_\ell,c)}{\sum_{c=0}^1 P_N(a_1,\ldots,a_{\ell-1},c)} \qquad (5)\end{aligned}$$

and the partial truth assignments in the last expression correspond to paths in D' beginning with the start node.

Lemma 8.3. *There is a distribution $D(X_1, \ldots, X_n)$ of width $poly(n, d, 1/\varepsilon)$ such that for every $a = (a_1, \ldots, a_n)$ it holds that*

$$(1-\varepsilon)P_D(a) \le P_{N,X}(a) \le (1+\varepsilon)P_D(a).$$

Proof. First we describe an auxiliary construction D'', which adds approximate evaluation of probabilities assigned to paths beginning at the start node of D', as suggested by (5). D'' is "almost" a GOBDD, except it has many sinks.

Description of D''

Levels are $0, \ldots, n+1$. Nodes are of the form (u, k), where u is a node of D' (including the sink), and $k \in \mathbf{N}$ (possible values for k are bounded by a polynomial in the parameters).

Assume that a partial truth assignment $C = c, X_1 = a_1, \ldots, X_{\ell-1} = a_{\ell-1}$ ends at u on level ℓ in D' and thus the product of edge probabilities along its path is $P_N(c, a_1, \ldots, a_{\ell-1})$. Then in D'' it ends in (u, k) where

$$(1-\delta)^{k+\ell} < P_N(c, a_1, \ldots, a_{\ell-1}) \leq (1-\delta)^k \qquad (6)$$

for a parameter δ to be determined later.

Thus each node in D' is split into several copies, collecting paths with similar probabilities. If u is a node in D' on level ℓ then the children of (u, k) in D'' are

$$(u^j, k + \lfloor \log_{1-\delta} p_u^j \rfloor),$$

where u^j is the j-child of u in D'. Then (6) follows by induction, considering $(1-\delta)^{t+1} < p_u^j \leq (1-\delta)^t$.

The OBDD D approximating $P_{N,X}$ is built using D''.

Description of D

The levels are now $1, \ldots, n+1$, with x_1 evaluated on level 1. Nodes on level ℓ are of the form (v_0, v_1), where v_0, v_1 are level ℓ nodes of D''. The start node is (v_0^*, v_1^*), where v_0^*, v_1^* are the children of the start node in D''. The j-child of (v_0, v_1) in D is (v_0^j, v_1^j), where v_i^j is the j-child of v_i in D''.

If $v_0 = (u_0, k_0)$ and $v_1 = (u_1, k_1)$ with children $v_i^j = (u_i^j, k_i^j)$ then the probability of the edge from (v_0, v_1) to its j-child (v_0^j, v_1^j) is the approximation of (5), i.e.,

$$\frac{(1-\delta)^{k_0^j} + (1-\delta)^{k_1^j}}{(1-\delta)^{k_0} + (1-\delta)^{k_1}}.$$

The multiple sinks of D'' are combined into a single sink.

In order to verify the approximation property of the distribution generated by D, consider a partial truth assignment $a_1, \ldots, a_\ell$. Let the nodes reached by the extensions $(c, a_1, \ldots, a_{\ell-1})$ in D'' be $(u_0, k_0), (u_1, k_1)$, and those reached by $(c, a_1, \ldots, a_\ell)$ be $(u_0^{a_\ell}, k_0^{a_\ell}), (u_1^{a_\ell}, k_1^{a_\ell})$. Then from (5) and (6) it follows that

$$\begin{aligned} \frac{(1-\delta)^{k_0^{a_\ell}+(\ell+1)} + (1-\delta)^{k_1^{a_\ell}+(\ell+1)}}{(1-\delta)^{k_0} + (1-\delta)^{k_1}} \leq \\ \leq P_N(a_\ell \,|\, a_1, \ldots, a_{\ell-1}) \leq \\ \leq \frac{(1-\delta)^{k_0^{a_\ell}} + (1-\delta)^{k_1^{a_\ell}}}{(1-\delta)^{k_0+\ell} + (1-\delta)^{k_1+\ell}}. \end{aligned} \qquad (7)$$

Note that

$$\frac{(1-\delta)^{k_0^{a_\ell}} + (1-\delta)^{k_1^{a_\ell}}}{(1-\delta)^{k_0} + (1-\delta)^{k_1}} = P_D(a_\ell \,|\, a_1, \ldots, a_{\ell-1}).$$

Hence (7) can be written as

$$\begin{aligned} (1-\delta)^{\ell+1} P_D(a_\ell \,|\, a_1, \ldots, a_{\ell-1}) \leq \\ \leq P_N(a_\ell \,|\, a_1, \ldots, a_{\ell-1}) \leq \\ \leq P_D(a_\ell \,|\, a_1, \ldots, a_{\ell-1})(1-\delta)^{-(\ell+1)}. \end{aligned} \qquad (8)$$

Multiplying the inequalities (8) we get

$$(1-\delta)^{n(n+1)} P_D(a) \leq P_{N,X}(a) \leq P_D(a)(1-\delta)^{-n(n+1)}$$

for every truth assignment a. Thus the theorem follows with choosing $\delta = \Theta(\varepsilon/n^2)$. □

Theorem 1.1 follows from applying Theorem 1.2 to the QTF obtained from the TAN N and the distribution D approximating the input distribution $P_{N,X}$, using error parameter $\varepsilon/3$ in both cases.

References

[1] A. Amarilli, M. Monet, and P. Senellart. Connecting width and structure in knowledge compilation. In *21st International Conference on Database Theory (ICDT 2018)*, volume 98 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 6:1–6:17, 2018.

[2] H. Chan and A. Darwiche. Reasoning about Bayesian network classifiers. In *UAI '03, Proceedings of the 19th Conference in Uncertainty in Artificial Intelligence*, pages 107–115, 2003.

[3] A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

[4] A. Darwiche and P. Marquis. A knowledge compilation map. *J. Artif. Intell. Res.*, 17:229–264, 2002.

[5] A. De and R. A. Servedio. Efficient deterministic approximate counting for low-degree polynomial threshold functions. *CoRR*, abs/1311.7178, 2013.

[6] A. del Val. An analysis of approximate knowledge compilation. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95*, pages 830–836, 1995.

[7] F. Doshi-Velez and B. Kim. A roadmap for a rigorous science of interpretability. *CoRR*, abs/1702.08608, 2017.

[8] M. Ford. *Architects of Intelligence: The truth about AI from the people building it*. Packt Publ., 2018.

[9] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163, 1997.

[10] M. Golea. On the complexity of rule extraction from neural networks and network querying. In *Rule Extraction for Trained Artificial Neural Networks Workshop*, pages 51–59, 1996.

[11] P. Gopalan, A. R. Klivans, and R. Meka. Polynomial-time approximation schemes for knapsack and related counting problems using branching programs. *CoRR*, abs/1008.3187, 2010.

[12] P. Gopalan, A. R. Klivans, R. Meka, D. Stefankovic, S. Vempala, and E. Vigoda. An FPTAS for #knapsack and related counting problems. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, pages 817–826, 2011.

[13] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. A survey of methods for explaining black box models. *ACM Comput. Surv.*, 51(5):93:1–93:42, 2019.

[14] K. Hosaka, Y. Takenaga, T. Kaneda, and S. Yajima. Size of ordered binary decision diagrams representing threshold functions. *Theor. Comput. Sci.*, 180(1-2):47–60, 1997.

[15] M. Jaeger. Probabilistic classifiers and the concepts they recognize. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003)*, pages 266–273, 2003.

[16] J. Kamp, A. Rao, S. P. Vadhan, and D. Zuckerman. Deterministic extractors for small-space sources. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 691–700, 2006.

[17] N. G. Kinnersley. The vertex separation number of a graph equals its path-width. *Inf. Process. Lett.*, 42(6):345–350, 1992.

[18] E. Korach and N. Solel. Tree-width, path-width, and cut-width. *Discrete Applied Mathematics*, 43(1):97–101, 1993.

[19] C. Lacave and F. Javier Díez. A review of explanation methods for Bayesian networks. *Knowledge Eng. Review*, 17(2):107–127, 2002.

[20] Z. C. Lipton. The mythos of model interpretability. *Commun. ACM*, 61(10):36–43, 2018.

[21] J. Makowsky and K. Meer. Polynomials of bounded tree-width. In D. Krob et al., editor, *Formal Power Series and Algebraic Combinatorics*, pages 692–703. Springer, 2000.

[22] K. Meer. Tree-width in algebraic complexity. *Fundam. Inform.*, 98(4):391–409, 2010.

[23] R. Meka and D. Zuckerman. Pseudorandom generators for polynomial threshold functions. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010*, pages 427–436, 2010.

[24] P. Scheffler. A linear algorithm for the pathwidth of trees. In R. Bodendiek and R. Henn, editors, *Topics in Combinatorics and Graph Theory*, pages 613–620. Physica-Verlag HD, 1990.

[25] Y. Shen, A. Choi, and A. Darwiche. Tractable operations for arithmetic circuits of probabilistic models. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016*, pages 3936–3944, 2016.

[26] A. Shih, A. Choi, and A. Darwiche. A symbolic approach to explaining Bayesian network classifiers. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pages 5103–5111, 2018.

[27] A. Shih, A. Choi, and A. Darwiche. Compiling Bayesian network classifiers into decision graphs. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, pages 7966–7974, 2019.

[28] E. Sommer. An approach to measuring theory quality. In *Advances in Knowledge Acquisition, 9th European Knowledge Acquisition Workshop, EKAW'96*, pages 195–211, 1996.

[29] Y. Takenaga, M. Nouzoe, and S. Yajima. Size and variable ordering of OBDDs representing threshold functions. In *Computing and Combinatorics, Third Annual International Conference*, pages 91–100, 1997.

[30] A. B. Tickle, R. Andrews, M. Golea, and J. Diederich. The truth will come to light: directions and challenges in extracting the knowledge embedded within trained artificial neural networks. *IEEE Trans. Neural Networks*, 9:1057–1068, 1998.

[31] S. T. Timmer, J.-J. Ch. Meyer, H. Prakken, S. Renooij, and B. Verheij. A two-phase method for extracting explanatory arguments from Bayesian networks. *Int. J. Approx. Reasoning*, 80:475–494, 2017.

[32] G. Varando, C. Bielza, and P. Larrañaga. Decision boundary for discrete Bayesian network classifiers. *Journal of Machine Learning Research*, 16:2725–2749, 2015.

[33] I. Wegener. *Branching Programs and Binary Decision Diagrams*. SIAM, 2000.