OBDD-NET: End-to-End Learning of Ordered Binary Decision Diagrams

Junming Qiu
Sun Yat-sen University
Guangzhou, China
Peng Cheng Laboratory
Shenzhen, China
qiujm9@mail2.sysu.edu.cn

Rongzhen Ye
Sun Yat-sen University
Guangzhou, China
yerzh@mail2.sysu.edu.cn

Weilin Luo
Sun Yat-sen University
Guangzhou, China
luowlin5@mail.sysu.edu.cn

Kunxun Qi
Sun Yat-sen University
Guangzhou, China
kunxunqi@foxmail.com

Hai Wan*
Sun Yat-sen University
Guangzhou, China
wanhai@mail.sysu.edu.cn

Yue Yu*
Peng Cheng Laboratory
Shenzhen, China
yuy@pcl.ac.cn

Abstract

Learning Ordered Binary Decision Diagrams (OBDDs) from large-scale datasets is an important topic of explainable artificial intelligence. However, existing search-based methods are still limited in scalability regarding dataset size, since they must explicitly encode the satisfaction of all examples in a dataset. To tackle this challenge, we introduce an OBDD encoding method to parameterize a neural network. This method frees satisfaction encoding of all examples in a dataset while leveraging mini-batch training techniques to enhance learning efficiency. Our main theoretical contribution is to prove that our approach enables the simulation of OBDD inference within a continuous space. Besides, we identify faithful OBDD encoding to fulfill the properties required by OBDDs, allowing to interpret an OBDD directly from the learned parameter assignment. With faithful OBDD encoding, we present an end-to-end neural model named `OBDD-NET`, being capable of coping with large-scale datasets. Experimental results exhibit better scalability and competitive prediction performance of `OBDD-NET` compared to state-of-the-art OBDD learners. Valuable insights about faithful OBDD encoding are derived from the ablation study. The implementation is available at: https://github.com/jmq-design/OBDD-NET.

CCS Concepts

• **Computing methodologies** → **Rule learning**.

Keywords

OBDD learning, End-to-End learning, Faithful encoding

*Both authors are corresponding authors.

CIKM '25, Seoul, Republic of Korea

ACM ISBN 979-8-4007-2040-6/2025/11
https://doi.org/10.1145/3746252.3761195

ACM Reference Format:
Junming Qiu, Rongzhen Ye, Weilin Luo, Kunxun Qi, Hai Wan, and Yue Yu. 2025. OBDD-NET: End-to-End Learning of Ordered Binary Decision Diagrams. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM '25), November 10–14, 2025, Seoul, Republic of Korea.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3746252.3761195

1 Introduction

Learning an interpretable model from large-scale data is a crucial topic in eXplainable Artificial Intelligence (XAI). Compared to (Binary) Decision Tree (DT), Binary Decision Diagram (BDD) is a more compact interpretable model due to the node sharing [5]. In particular, Ordered BDD (OBDD) [4], a tractable subset of BDD, has recently received increasing attention in XAI [5, 15, 23, 24, 33]. The main attractions of the OBDD representation on XAI are twofold. On the one hand, its compact graph structure (especially with small size and depth) is commonly human-understandable [5, 15]. On the other hand, the excellent tractability of OBDDs (e.g., supporting polytime satisfiability and model counting check) is useful for decision explanation and quantitative robustness analysis of complex classifiers that are not easy to understand directly [7, 24, 33].

Informally, the ordering property imposed on an OBDD requires that decision features appear in the same order on any path in the Directed Acyclic Graph (DAG) representation of the BDD. It can be view as a hierarchical DAG where all vertices at a specific non-terminal level should be associated with the same unique feature. The *depth* and *size* of an OBDD refer to the number of features and vertices it contains, respectively. Figure 1(a) depicts an OBDD involving three binary features, which compactly represents two IF-THEN sentences: (1) if (NOT f_1) AND f_3, then predict true; and (2) if f_1 AND f_2 AND f_3, then predict true.

In practice, learning OBDDs with limited depth H still turns out to be quite challenging. Even for a fixed feature ordering, the search space reaches at least double-exponential, due to the fact that there are 2^{2^H} (logically) distinguishing H-ary Boolean functions.

Existing studies for learning OBDDs with limited depth from data consist of two categories. Classic heuristic approaches [19] rely on mutual information heuristics, but often produce OBDDs with

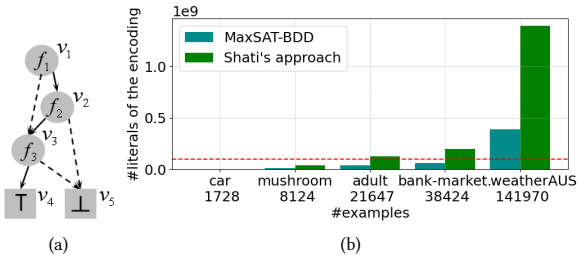


Figure 1: (a) An OBDD of depth 3 and size 5. (b) Encoding size for learning OBDDs in limited depth 6 for some datasets in experiments.'#x' means the number of x.

poor prediction quality [15]. Recently, two exact methods: MaxSAT-BDD [15] and Shati et al.'s approach[23], have been proposed for learning optimal (reduced) OBDDs maximizing classification accuracy, where the learning problem is reduced to a Maximum Satisfiability (MaxSAT) optimization problem. However, these approaches face limitations in terms of scalability. They treat learning OBDDs as a constraint programming problem in which the satisfaction towards all examples in the entire dataset needs to be encoded at a time. From Figure 1(b), the encoding size (i.e., the number of literals of the encoded MaxSAT instance) of their approaches grows gradually with the dataset size. When applied to large-scale datasets, the encoding size become extremely large (e.g., greater than 100 million for both approaches when dataset size reaches 100 thousand), being quite challenging to be solved by modern MaxSAT solvers.

In this paper, we address the challenge by treating the OBDD learning problem as a gradient-based structure learning problem, without needing to explicitly encode large-scale datasets to constrain programming instances. Our main contributions include:

(1) We introduce an OBDD encoding method to parameterize a neural network and design a reachability-based procedure for evaluating the satisfaction relation between an OBDD encoding and an example. The reachability-based procedure avoids explicit satisfaction encoding towards the whole dataset by computing the satisfaction relation directly. We prove that the reachability-based procedure enables the simulation of OBDD inference within a continuous space.
(2) We identify a subset of the OBDD encoding, named faithful OBDD encoding, that fulfills the properties required by OBDDs. It allows an OBDD to be directly interpreted from the learned parameter assignment of the neural network. With the faithful OBDD encoding, we develop an end-to-end neural model named `OBDD-NET`, which enables the utilization of gradient descent optimization and mini-batch training techniques for efficient learning.
(3) Experimental results on 10 small and 8 large datasets show that `OBDD-NET` achieves better scalability than state-of-the-art (SOTA) OBDD learners, scaling to million-size datasets, while maintaining competitive predictive performance.

2 Related Work

Rule learning approaches can be categorized into three categories.

Heuristic-based Approaches. These methods include some traditional algorithms for learning decision trees, such as C4.5 [22] and CART [3], which employ some locally optimal strategies (e.g., information gain, Gini index) to iteratively construct tree. Besides, there are also some approaches to learn OBDDs: top-down [18] and bottom-up [19] learning algorithms [1], which rely on mutual information heuristics. An advantage of the latter is that the depth of the built OBDD could be controlled as a preset parameter [14].

Constraint Solver-based Approaches. Several approaches have been proposed for learning optimal (reduced) OBDDs. Hu et al. [15] proposed a MaxSAT-based learning approach (MaxSAT-BDD) for learning optimal OBDDs in limited depth to maximize the binary classification accuracy. Further, Shati et al. [23] presented a lifted MaxSAT-based encoding for a multi-terminal variant of OBDDs (MTBDDs). Besides, Cabodi et al. [5, 6] developed an iterative SAT-based approach for deriving optimal OBDDs with minimum size, however focuses on correctly classifying all examples in the datasets. Florio et al. [10] proposed an Mixed-Integer Linear Programming based encoding for learning optimal decision diagrams (ODDs) with pre-defined skeletons. However, their targeted decision diagrams do not satisfy the ordering property, hence are not OBDDs.

Gradient-based Approaches. Recently, there has been a growing interest in leveraging the gradient-based structure learning technique to mine logical rules, such as AND-OR rules [21, 28, 29], first-order rules [11, 13, 30], circuits [27]. The main insight is to conduct a continuous relaxation on the discrete architecture representation, so as to leverage gradient descent techniques for an efficient optimization of the architecture. However, this technique has not yet been applied to the OBDD learning problem.

Different from these approaches, we apply the gradient-based structure learning technique to the problem of learning OBDDs with limited depth. Accordingly, we consider MaxSAT-BDD [15] and Shati et al.'s approach[23] as the most suitable SOTA of exact approaches, and OODG [19] as the SOTA of heuristic approaches.

3 Preliminaries

Throughout this paper, we use a lowercase (resp. uppercase) bold letter for a vector (resp. matrix). $(\mathbf{v})_i$ denotes the i-th element of a vector $\mathbf{v}$, and $(\mathbf{M})_{i,j}$ denotes the element of a matrix $\mathbf{M}$ at the i-th row and the j-th column, with i and j starting from 1. $(\mathbf{M})_i$ denotes the i-th row of $\mathbf{M}$. $\mathbf{M}^T$ denotes the transpose of the matrix $\mathbf{M}$, so is $\mathbf{v}^T$.

Classification. In this paper, we consider binary classification problems defined on a finite set $\mathcal{F} = \{f_1, \cdots, f_m\}$ of binary features $f_i \in \{\mathbf{0}, \mathbf{1}\}$, and a set $\mathcal{K} = \{\mathbf{0}, \mathbf{1}\}$ of binary classes. A classifier is to compute a *classification function* mapping feature space $\mathbb{F} = \{\mathbf{0}, \mathbf{1}\}^m$ into the set of classes $\mathcal{K}$. A *literal* on binary feature $f_i \in \mathcal{F}$, represented as f_i or $\neg f_i$, denotes that the feature takes Boolean value $\mathbf{1}(\top)$ or $\mathbf{0}(\bot)$, respectively. An *example* ω is characterized by a pair $(\mathbf{w}, c)$, where $\mathbf{w} \in \mathbb{F}$ and $c \in \mathcal{K}$. For ease of description, we reuse the term *example* to refer to $\mathbf{w}$, leaving c implicit; and represent it as a sequence of literals over $\mathcal{F}$. A dataset (or examples) $\mathcal{E} = \{\omega_1, \cdots, \omega_M\}$ is partitioned into positive examples $\mathcal{E}^+$ and negative ones $\mathcal{E}^-$, according to the ground truth label $\text{lab}(\omega_i)$ of each ω_i.

[1] Another name, called Oblivious Read-Once Decision Graph (OODG), is used by them, which is equivalent to OBDD in binary classification for binary datasets[15].

3.1 Binary Decision Diagrams

A *Binary Decision Diagram* (BDD) [1] is a rooted DAG, denoted by $(\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r)$, where

- $\mathcal{V}$ is the set of vertices with each *leaf* in $\{\top, \bot\}$; and each *internal* vertex v associated to a Boolean *decision variable* $dV(v) \in \mathcal{F}$ and two children left(v), right(v);
- $\mathcal{T} \subseteq \mathcal{V} \times (\mathcal{F} \cup \{\top, \bot\})$ is the set of vertex-tag pairs.
- $\mathcal{E}_l, \mathcal{E}_r \subseteq \mathcal{V} \times \mathcal{V}$ are the set of all **1**-edges (i.e., $(v$,left$(v))$) and **0**-edges (i.e., $(v$,right$(v))$), respectively.

Definition 1. [[4].] Let π be a total variable ordering over $\mathcal{F}$. An *Ordered Binary Decision Diagram* (OBDD) respecting the ordering π is a BDD that satisfies the *ordering property*: for any internal vertex v, it holds that $\mathsf{rank}_\pi(dV(v)) < \mathsf{rank}_\pi(dV(\text{left}(v)))$ and $\mathsf{rank}_\pi(dV(v)) < \mathsf{rank}_\pi(dV(\text{right}(v)))$, where $\mathsf{rank}_\pi(f_i)$, also written $\mathsf{rank}(f_i)$, is the rank of the variable f_i in the ordering π.

Intuitively, the ordering property requires that decision variables appear in the same order on any path. A *Reduced OBDD* (ROBDD) is an OBDD that contains neither redundant internal vertex v with left(v) = right(v), nor distinct vertices v_1 and v_2 s.t the sub-graphs rooted by them are isomorphic. For any OBDD, there is a unique ROBDD representation, and the reduction can be done effectively in polytime [4]. Note that, the reduction results in an OBDD logically equivalent to the original one.

Let $\psi = (\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r)$ be an OBDD over $\mathcal{F}$. The *size* of ψ is the number of its vertices $|\mathcal{V}|$. The *depth* of ψ is the number of variables appearing in ψ (i.e., $|\{f_i \in \mathcal{F} \mid (v, f_i) \in \mathcal{T}, v \in \mathcal{V}\}|$). Note that an OBDD could have a depth greater than the length of the longest path from the root to any leaf.

Given an example $\omega = [l_1, \cdots, l_m]$ and a BDD classifier $\psi = (\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r)$ over $\mathcal{F}$, we say ω *satisfies* ψ, denoted by $\omega \models \psi$, if for some $1 \le n \le |\mathcal{F}|$, there is a sequence of vertices $q_1, \cdots, q_{n+1}$ such that $q_1 = v_1$, $(q_{n+1}, \top) \in \mathcal{T}$, and for every $1 \le i \le n$, if $(q_i, f_k) \in \mathcal{T}$ then either (1) $(q_i, q_{i+1}) \in \mathcal{E}_l$ and $l_k = f_k$; or (2) $(q_i, q_{i+1}) \in \mathcal{E}_r$ and $l_k = \neg f_k$. Intuitively, $\omega \models \psi$ indicates that there is a path κ of length n of ψ from the root vertex v_1 to the leaf vertex $\top$ such that κ is in consistence with the example ω. Given two BDDs $\varphi = (\mathcal{V}', \mathcal{T}', \mathcal{E}_l', \mathcal{E}_r')$ and $\psi = (\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r)$ over $\mathcal{F}$, we say φ and ψ are *isomorphic*, if there is a one-to-one correspondence ρ between $\mathcal{V}'$ and $\mathcal{V}$ s.t. for any $v_i', v_j' \in \mathcal{V}'$: (1) $(v_i', t) \in \mathcal{T}'$ iff $(\rho(v_i'), t) \in \mathcal{T}$, where $t \in (\mathcal{F} \cup \{\top, \bot\})$; and (2) $(v_i', v_j') \in \mathcal{E}_l'$ iff $(\rho(v_i'), \rho(v_j')) \in \mathcal{E}_l$; and (3) $(v_i', v_j') \in \mathcal{E}_r'$ iff $(\rho(v_i'), \rho(v_j')) \in \mathcal{E}_r$.

Example 3.1 illustrates a (reduced) OBDD classifier that satisfied by the example $\omega = [\neg f_1, f_2, f_3]$.

Example 3.1. Let $\mathcal{F} = \{f_1, f_2, f_3\}$ and π a variable ordering $[f_1, f_2, f_3]$. As shown in Figure 1(a) (solid (dotted) lines correspond to **1**-edges (**0**-edges)), consider an OBDD $\psi = (\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r)$ respecting π of size 5 and depth 3 where $\mathcal{V} = \{v_1, \cdots, v_5\}$, $\mathcal{T} = \{(v_1, f_1), (v_2, f_2), (v_3, f_3), (v_4, \top), (v_5, \bot)\}$, $\mathcal{E}_l = \{(v_1, v_2), (v_2, v_3), (v_3, v_4)\}$ and $\mathcal{E}_r = \{(v_1, v_3), (v_2, v_5), (v_3, v_5)\}$. Given an example $\omega = [\neg f_1, f_2, f_3]$. Consider a path $\kappa = v_1, v_3, v_4$ of length 2 in ψ. As $(v_1, f_1) \in \mathcal{T}$, $(v_1, v_3) \in \mathcal{E}_r$ and $l_1 = \neg f_1$, $(v_3, f_3) \in \mathcal{T}$, $(v_3, v_4) \in \mathcal{E}_l$ and $l_3 = f_3$, we get that $\omega \models \psi$.

Learning OBDD Problem. We focus on the problem of learning (reduced) OBDD in a limited depth. Given that the reduction of an OBDD can be done effectively[4], somewhat similar to previous works[15, 23], we first learn a non-reduced OBDD then apply post-processing to get the corresponding ROBDD.

$P(\mathcal{E}, H)$: Given a set of examples $\mathcal{E}$ for binary classification, the goal is to find an OBDD ψ of maximum depth H that maximizes the accuracy of ψ on $\mathcal{E}$.

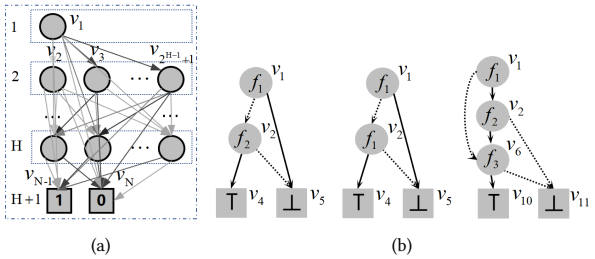


Figure 2: (a) Illustration structure for inducing the parameters of OBDD-NET. (b) Visualisations of the BDDs in Example 4.1 (the leftmost two graphs) and 4.3 (the rightmost graph).

4 OBDD Encoding and Inference

In this section, we first introduce the model structure of OBDD-NET and present the definition of OBDD encoding, followed by an inference process of OBDD-NET on classification. Then, a faithful subset of the OBDD encoding is provided and shown to have the equivalent inference to the OBDD representation. In such case, an OBDD with the same classification behavior as OBDD-NET can be interpret directly from the learned parameter assignment.

4.1 Model Structure of OBDD-NET

Similarly to a binary classifier, OBDD-NET takes as input an example and outputs an evaluation on the satisfaction relation between OBDD-NET and the example. We start by the parameter structure of OBDD-NET for inducing the definition of the parameter set. As shown in Figure 2(a), it consists of a collection of nodes $v_1, \cdots, v_N$ partitioned into $H + 1$ distinct levels, where each level except the root and terminal levels contains $W = 2^{H-1}$ nodes. This allows to cover the full search space of the learning problem $P(\mathcal{E}, H)$, due to the fact that there are at most 2^{H-1} vertices at the last internal level of an OBDD of depth H.

In the rest of the paper, we use H to denote the *depth* (i.e., the number of internal levels) of OBDD-NET. Accordingly, the *size* of OBDD-NET (i.e., the number of the whole nodes), is $N = 2^{H-1} \cdot (H - 1) + 3$. We denote $\text{lev}(i)$ the level index of node v_i in OBDD-NET, i.e., $\text{lev}(i) = 1$ if $i = 1$ and $\text{lev}(i) = (i - 2)/2^{H-1} + 2$ otherwise.

As depicted in Figure 2(a), we treat the OBDD learning as a (labelled) leveled graph learning. In the final learned graph, each level is expected to be assigned to a unique feature and each edge to be categorized as **1**-edge or **0**-edge. To explore a compact OBDD, cross-layer connections between nodes from top to bottom are permitted. The trainable parameters of OBDD-NET are defined as follows.

Definition 2. Let $\mathcal{F}$ be a set of binary features. The *parameter set* Γ of OBDD-NET [2] of depth $H \leq |\mathcal{F}|$ is defined as $\Gamma = \Gamma_{\text{dec}} \cup \Gamma_{\text{left}} \cup \Gamma_{\text{right}}$, where $\Gamma_{\text{dec}} = \{(\Gamma_{\text{dec}})_{i,k} \in \mathbb{R} \mid 1 \leq i \leq H, 1 \leq k \leq |\mathcal{F}|\}$, $\Gamma_{\text{left}} = \{(\Gamma_{\text{left}})_{i,j} \in \mathbb{R} \mid 1 \leq i \leq N-2, i < j \leq N, \text{lev}(i) < \text{lev}(j)\}$ and $\Gamma_{\text{right}} = \{(\Gamma_{\text{right}})_{i,j} \in \mathbb{R} \mid 1 \leq i \leq N-2, i < j \leq N, \text{lev}(i) < \text{lev}(j)\}$.

Note that, for brevity, we use the same symbol Γ to represent the parameter set of OBDD-NET and an assignment of it interchangeably, so does the symbol θ introduced later.

To establish the relationship between OBDD-NET and the DAG of an OBDD, we constrain the parameter assignment Γ within the range $[0, 1]$. In Definition 3, we formally define the restricted parameter assignment as an *OBDD encoding*.

Definition 3. Let Γ be a parameter set of OBDD-NET of depth H. An *OBDD encoding* θ of OBDD-NET of depth H is defined as $\theta = \{\varepsilon \in \Gamma | \varepsilon \in \mathbb{R}^{[0,1]}\}$, where $\mathbb{R}^{[0,1]}$ denotes the real value range from 0 to 1. θ_{dec}, θ_{left} and θ_{right} are defined as the counterparts of Γ_{dec}, Γ_{left} and Γ_{right}, respectively.

An OBDD encoding is capable of representing an OBDD. For any $1 \leq i \leq H$, the parameter $(\theta_{\text{dec}})_{i,k}$ indicates the degree of probability that the level i is associated to a feature $f_k \in \mathcal{F}$. In this way, θ_{dec} allows to automatically select some important features from the whole set $\mathcal{F}$ (in the case where $H < |\mathcal{F}|$) and enables the feature ordering to be optimized in an end-to-end fashion. For any pair of nodes v_i and v_j with v_i precedes v_j w.r.t. the level, the parameter $(\theta_{\text{left}})_{i,j}$ (resp. $(\theta_{\text{right}})_{i,j}$) determines the degree of probability that the left (resp. right) child of v_i is v_j. We illustrate an OBDD encoding with Example 4.1.

Example 4.1. Let $\mathcal{F} = \{f_1, f_2\}$ and θ be an OBDD encoding of depth $H = 2$ (and size $N = 5$) where $(\theta_{\text{dec}})_{1,1} = 1$, $(\theta_{\text{dec}})_{2,1} = 0.2$, $(\theta_{\text{dec}})_{2,2} = 0.8$, $(\theta_{\text{left}})_{1,5} = (\theta_{\text{left}})_{2,4} = (\theta_{\text{right}})_{1,2} = (\theta_{\text{right}})_{2,5} = 1$ and the other parameters are assigned 0. As shown in Figure 2(b), the BDD represented by θ is most likely to be $(\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r)$ where $\mathcal{V} = \{v_1, v_2, v_4, v_5\}$, $\mathcal{T} = \{(v_1, f_1), (v_2, f_2), (v_4, \top), (v_5, \bot)\}$, $\mathcal{E}_l = \{(v_1, v_5), (v_2, v_4)\}$ and $\mathcal{E}_r = \{(v_1, v_2), (v_2, v_5)\}$. While it may also to be a slightly different BDD with $\mathcal{T} = \{(v_1, f_1), (v_2, f_1), (v_4, \top), (v_5, \bot)\}$.

4.2 Inference for the Satisfaction Relation

Definition 4 offers an inference process on satisfaction relation between an OBDD encoding θ of OBDD-NET and a given example ω.

Before giving the formal definition, we introduce some notations as follows. We use the corresponding bold symbols $\boldsymbol{\theta}_{\text{dec}}$ for the matrix representation of the encoding parameter θ_{dec}; and $\boldsymbol{\theta}_{\text{left}}$ and $\boldsymbol{\theta}_{\text{right}}$ for the ones obtained from θ_{left} and θ_{right} by filling with 1s the last two diagonal elements, and with 0s the other unused elements of θ_{left} and θ_{right}, respectively. For example, $\boldsymbol{\theta}_{\text{left}}$ is an $N \times N$ matrix where $(\boldsymbol{\theta}_{\text{left}})_{N-1,N-1} = (\boldsymbol{\theta}_{\text{left}})_{N,N} = 1$, $(\boldsymbol{\theta}_{\text{left}})_{i,j} = (\theta_{\text{left}})_{i,j}$ if $(\theta_{\text{left}})_{i,j} \in \theta_{\text{left}}$, and $(\boldsymbol{\theta}_{\text{left}})_{i,j} = 0$ otherwise. In this case, the **1**-leaf and **0**-leaf have both outgoing **1**-edge and **0**-edge to themselves. Hereafter, we define an $N \times N$ transition matrix $\mathbf{M}^{\omega}$ (on the OBDD encoding θ) under a given example ω as follow:

- for $1 \leq i \leq N-2$, $(\mathbf{M}^{\omega})_i = p_i \cdot (\boldsymbol{\theta}_{\text{left}})_i + (1 - p_i) \cdot (\boldsymbol{\theta}_{\text{right}})_i$, where $p_i = (\boldsymbol{\theta}_{\text{dec}})_{\text{lev}(i)} \cdot \mathbf{w}^T$ indicates the probability of choosing **1**-edge at node v_i under ω, where $\mathbf{w}^T$ denotes the transpose of the vector representation $\mathbf{w}$ of ω.
- for $N-1 \leq i \leq N$, $(\mathbf{M}^{\omega})_i = (\boldsymbol{\theta}_{\text{left}})_i$.

Intuitively, $(\mathbf{M}^{\omega})_{i,j}$ indicates the transition probability from node v_i to v_j via **1**-edge or **0**-edge.

Definition 4. Let θ be an OBDD encoding of OBDD-NET of depth H. Given an example ω over $\mathcal{F}$. OBDD-NET recursively computes (k)-reachability vector $(\mathbf{r}^{\omega})^{(k)}$ of the root node v_1 under ω as follows:

$$\begin{aligned} (\mathbf{r}^{\omega})^{(1)} &= (\mathbf{M}^{\omega})_1, \\ (\mathbf{r}^{\omega})^{(k)} &= \sigma_{01}((\mathbf{r}^{\omega})^{(k-1)} \cdot \mathbf{M}^{\omega}), k > 1, \end{aligned} \tag{1}$$

where $\sigma_{01}(x) = \max(0, \min(1, x))$. Then, OBDD-NET outputs the evaluation of satisfaction relation between θ and ω, denoted by $\text{ESat}(\theta, \omega)$, as $((\mathbf{r}^{\omega})^{(H)})_{N-1}$.

The vector $(\mathbf{r}^{\omega})^{(k)}$ represents the probability that the root node v_1 can reach to other nodes through (exact) k-step transition. In particular, as the OBDD encoding θ has a loop at **1**-leaf, $\text{ESat}(\theta, \omega)$ (i.e., $((\mathbf{r}^{\omega})^{(H)})_{N-1}$) computes the summation of probability to the k-reachability from root node v_1 to **1**-leaf v_{N-1} for $1 \leq k \leq H$. In this case, $\text{ESat}(\theta, \omega)$ evaluates, within a continuous space, the satisfaction relation between θ and a given example ω, by simulating the inference of an OBDD on classification. A more specific connection between the two inference process will be discussed later (Theorem 4.5). We illustrate the inference process ESat with Example 4.2.

Example 4.2. Consider again the OBDD encoding θ in Example 4.1. Its matrix representation is: $\boldsymbol{\theta}_{\text{dec}} = \begin{bmatrix} 1 & 0 \\ 0.2 & 0.8 \end{bmatrix}$,

$$\boldsymbol{\theta}_{\text{left}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \boldsymbol{\theta}_{\text{right}} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Given an example $\omega = [\neg f_1, f_2]$, its vector representation is $\mathbf{w} = [0\ 1]$. We have that $p_1 = (\boldsymbol{\theta}_{\text{dec}})_1 \cdot \mathbf{w}^T = [1\ 0] \cdot [0\ 1]^T = 0$, $p_2 = p_3 = (\boldsymbol{\theta}_{\text{dec}})_2 \cdot \mathbf{w}^T = [0.2\ 0.8] \cdot [0\ 1]^T = 0.8$. By Definition 4, we get the transition matrix $\mathbf{M}^{\omega}$ and reachability vector $(\mathbf{r}^{\omega})^{(2)}$ as follows.

$$\mathbf{M}^{\omega} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.8 & 0.2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$(\mathbf{r}^{\omega})^{(2)} = \sigma_{01}\left(\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}^T \cdot \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.8 & 0.2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0.8 \\ 0.2 \end{bmatrix}^T.$$

Finally, OBDD-NET evaluates the satisfaction between θ and ω as $\text{ESat}(\theta, \omega) = ((\mathbf{r}^{\omega})^{(2)})_4 = 0.8$.

4.3 The Faithful Subclass of OBDD Encoding

Obviously, not any OBDD encoding corresponds to a valid OBDD. To guarantee the correspondence between an OBDD encoding and

[2]Note that, H is the (unique) hyper-parameter to define the network structure of OBDD-NET, while neither W nor N is.

OBDD, we hereafter define a faithful subset of OBDD encoding, called *faithful OBDD encoding*.

Definition 5. Let θ be an OBDD encoding of OBDD-NET of depth H. We say θ is *faithful* if it satisfies the following:

1. $\forall \varepsilon \in \theta : \varepsilon = 0 \vee \varepsilon = 1$
2. $\forall i \in [1, H] : \sum_{k=1}^{|\mathcal{F}|} (\theta_{\text{dec}})_{i,k} = 1$
3. $\forall i \in [1, N-2] : \sum_{1 \le i < j \le N, \text{lev}(i) < \text{lev}(j)} (\theta_{\text{left}})_{i,j} = 1$ $\wedge \sum_{1 \le i < j \le N, \text{lev}(i) < \text{lev}(j)} (\theta_{\text{right}})_{i,j} = 1$
4. $\forall k \in [1, |\mathcal{F}|] : \sum_{i=1}^{H} (\theta_{\text{dec}})_{i,k} \le 1$

To begin with, constraint 1 restricts each parameter to be assigned a Boolean value 0 or 1. With this basic restriction, the rest of the constraints can fulfill the whole properties imposed on OBDDs. Specifically, constraint 2 ensures that every non-terminal level is associated to exactly one feature. While condition 3 forces that each non-terminal node has exactly one left and right child respectively, and must has a node with greater level index as its child. The last constraint requires each feature to appear at most once at the non-terminal levels. In this case, the associated features of the non-terminal levels form an order over $\mathcal{F}$. This, together with the constraint 3, implies that any path obeys the same feature ordering. So, the faithfulness fulfills the ordering property required by OBDDs.

Faithful OBDD Encoding vs OBDD. We claim that any arbitrary OBDD can be represented as a faithful OBDD encoding of an equal or greater depth. Given an arbitrary OBDD ψ of depth H (i.e., with H different features), a simple way is to construct an OBDD encoding θ of the same depth H and maintain the topology of ψ, where some redundant nodes directing to the terminal nodes may appear at the end of each level. Of course, this also applies to a faithful OBDD encoding with greater depth.

Reversely, Definition 6 offers a decoding function mapping a faithful OBDD encoding to an OBDD.

Definition 6. Let θ be a faithful OBDD encoding of OBDD-NET of depth H. The decoding function decode(θ) computes a 4-tuple $(\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r)$ defined below:

1. $v_1 \in \mathcal{V}$, and $\forall j \in (1, N] : v_j \in \mathcal{V}$ if there is $i < j$ s.t. $v_i \in \mathcal{V}$ and either $(\theta_{\text{left}})_{i,j} = 1$ or $(\theta_{\text{right}})_{i,j} = 1$.
2. $(v_{N-1}, \top) \in \mathcal{T}$ if $v_{N-1} \in \mathcal{V}$; and $(v_N, \bot) \in \mathcal{T}$ if $v_N \in \mathcal{V}$.
3. $\forall i \in [1, N-2] : (v_i, f_k) \in \mathcal{T}$ if $v_i \in \mathcal{V}$ and $(\theta_{\text{dec}})_{\text{lev}(i),k} = 1$.
4. $\forall i \in [1, N-2] : (v_i, v_j) \in \mathcal{E}_l$ if $v_i \in \mathcal{V}$ and $(\theta_{\text{left}})_{i,j} = 1$; and $(v_i, v_j) \in \mathcal{E}_r$ if $v_i \in \mathcal{V}$ and $(\theta_{\text{right}})_{i,j} = 1$.

Lemma 1 states that decode guarantees to interpret a BDD satisfying the ordering property from a faithful OBDD encoding θ.

Lemma 1. *For any faithful OBDD encoding θ,* decode(θ) *computes an OBDD representation.*

Moreover, Theorem 4.4 shows the existence of a faithful OBDD encoding syntactically corresponding to an arbitrary OBDD (under the decoding function decode). This means that the faithful OBDD encoding is a complete encoding method for the OBDD representation. Therefore, it makes sense to apply the faithful OBDD encoding in OBDD-NET to learn OBDDs. Example 4.2 shows an OBDD encoding θ faithful to the OBDD ψ in Example 3.1 and the decoding result decode(θ).

Algorithm 1: Interpreting OBDD Representation

Input: An OBDD encoding θ of depth H
Output: A ROBDD $(\mathcal{V}^*, \mathcal{T}^*, \mathcal{E}_l^*, \mathcal{E}_r^*)$ interpreted from θ

reset the maximum of each $(\theta_{\text{left}})_i$ (resp. $(\theta_{\text{right}})_i$ to 1 and all other values to 0s;
calculate $\mathcal{V}$ by the first operation of decode(θ);
$L \leftarrow \{\text{lev}(i) \mid v_i \in \mathcal{V}, 1 \le i \le N\}$; $F \leftarrow \emptyset$;
for $i \in [1, H]$ *and* $i \in L$ **do**
  $k^\star \leftarrow \arg\max_{k \in [1, |\mathcal{F}|] \wedge k \notin F} (\theta_{\text{dec}})_{i,k}$;
  reset $(\theta_{\text{dec}})_{i,k^\star}$ of $(\theta_{\text{dec}})_i$ to 1, the others to 0s;
  $F \leftarrow F \cup \{k^\star\}$;
select an arbitrary unique $k' \notin F$ for remaining $(\theta_{\text{dec}})_i$, and reset $(\theta_{\text{dec}})_{i,k'}$ to 1 and others to 0s;
$(\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r) \leftarrow$ decode(θ);
$(\mathcal{V}^*, \mathcal{T}^*, \mathcal{E}_l^*, \mathcal{E}_r^*) \leftarrow$ reduce($(\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r)$);
return $(\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r)$;

Example 4.3. Recall the OBDD ψ shown in Example 3.1. A faithful OBDD encoding θ of depth $H = 3$ (and size $N = 11$) of ψ is as follows: $(\theta_{\text{dec}})_{1,1} = (\theta_{\text{dec}})_{2,2} = (\theta_{\text{dec}})_{3,3} = 1$; $(\theta_{\text{left}})_{1,2} = (\theta_{\text{left}})_{2,6} = (\theta_{\text{left}})_{6,10} = 1$; $(\theta_{\text{right}})_{1,6} = (\theta_{\text{right}})_{2,11} = (\theta_{\text{right}})_{6,11} = 1$; $(\theta_{\text{left}})_{i,10} = (\theta_{\text{right}})_{i,11} = 1$ for each i in $\{3, 4, 5, 7, 8, 9\}$; and the other parameters are assigned 0. Reversely, by Definition 6, as shown in Figure 2(b), decode(θ) computes a 4-tuple $(\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r)$ where $\mathcal{V} = \{v_1, v_2, v_6, v_{10}, v_{11}\}$, $\mathcal{T} = \{(v_1, f_1), (v_2, f_2), (v_6, f_3), (v_{10}, \top), (v_{11}, \bot)\}$, $\mathcal{E}_l = \{(v_1, v_2), (v_2, v_6), (v_6, v_{10})\}$ and $\mathcal{E}_r = \{(v_1, v_6), (v_2, v_{11}), (v_6, v_{11})\}$. Clearly, it represents an OBDD isomorphic to ψ.

Theorem 4.4. *For any OBDD representation ψ of depth H', there exists a faithful OBDD encoding θ of* OBDD-NET *of depth $H \ge H'$ such that* decode(θ) *and ψ are isomorphic.*

Proof Sketch. Let $\mathcal{V}' = \{v'_1, \cdots v'_{N'}\}$. Let ψ be an OBDD $(\mathcal{V}', \mathcal{T}', \mathcal{E}'_l, \mathcal{E}'_r)$ of depth H' respecting $\pi = [f_1, \cdots, f_{H'}]$ over $\mathcal{F}$. Let map : $\{1, \cdots, N'\} \to \{1, \cdots, N\}$, where map(1) = 1, and for $1 < i \le N'$, map(i) = $N - 1$ (resp. map(i) = N) if $(v_i, \top) \in \mathcal{T}$ (resp. $(v_i, \bot) \in \mathcal{T}$), otherwise map($i$) = $(k-2) \cdot 2^{H-1} + j + 1$ when v_i is the j-th vertex of ψ s.t. rank($dV(v_i)$) = k. With the function map, roughly as described at the beginning of this subsection, we can construct a faithful OBDD encoding of depth $H \ge H'$ for ψ, denoted by $\theta_{\psi(H)}$. By Lemma 1, decode($\theta_{\psi(H)}$) is an OBDD representation. Let φ = decode($\theta_{\psi(H)}$) = $(\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r)$. Let $\rho : \mathcal{V}' \to \mathcal{V}$ defined as: for any $1 \le i \le N'$, $\rho(v'_i) = v_{\text{map}(i)}$. Then, by the construction of $\theta_{\psi(H)}$ and Definition 6, we can prove that (1) the function ρ is a one-to-one correspondence between $\mathcal{V}'$ and $\mathcal{V}$; (2) for any $1 \le i \le N'$, $(v'_i, t) \in \mathcal{T}'$ iff $(v_{\text{map}(i)}, t) \in \mathcal{T}$, where $t \in (\mathcal{F} \cup \{\top, \bot\})$; (3) for any $1 \le i < j \le N'$, $(v'_i, v'_j) \in \mathcal{E}'_l$ iff $(v_{\text{map}(i)}, v_{\text{map}(j)}) \in \mathcal{E}_l$; and $(v'_i, v'_j) \in \mathcal{E}'_l$ iff $(v_{\text{map}(i)}, v_{\text{map}(j)}) \in \mathcal{E}_l$. So, φ and ψ are isomorphic. □

4.4 Inference of OBDD-NET vs Inference of OBDD

We hereafter bridge the inference $\models$ of OBDDs and the inference ESat (in Definition 4) of OBDD-NET, by showing the consistency between them under condition of faithfulness. This allows to treat the OBDD classifier learning problem as a joint optimization of OBDD-NET in terms of maximizing the consistency with dataset and the faithfulness to the OBDD representation.

Lemma 2 states that the H-reachability evaluation $((\mathbf{r}^{\omega})^{(H)})_{N-1} = 1$ precisely when there is a path from v_1 to v_{N-1} within H length and in consistence with ω. Similar to the context of OBDDs, given an example $\omega = [l_1, \cdots, l_m]$ and a faithful OBDD encoding θ of depth H, we say v_i can reach v_j in $1 \leq n \leq H$ step in consistence with ω, if there is a sequence of nodes $v_{s1}, \cdots, v_{s(n+1)}$ (a path of length n in θ) such that $v_{s1} = v_i$, $v_{s(n+1)} = v_j$, and for every $1 \leq i \leq n$, if $(\theta_{\text{dec}})_{\text{lev}(si),k} = 1$ then either (1) $(\theta_{\text{left}})_{si,s(i+1)} = 1$ and $l_k = f_k$; or (2) $(\theta_{\text{right}})_{si,s(i+1)} = 1$ and $l_k = \neg f_k$.

Lemma 2. *Let θ be a faithful OBDD encoding of depth H, and ω an example $[l_1, \cdots, l_m]$. Then, v_1 can reach v_{N-1} in n step in consistence with ω for some $1 \leq n \leq H$ iff $((\mathbf{r}^{\omega})^{(H)})_{N-1} = 1$.*

Theorem 4.5 shows the equivalence of the inference on classification of an OBDD ψ and that of a faithful OBDD encoding θ whose decoding result $\text{decode}(\theta)$ is isomorphic to ψ. That is, for any arbitrary OBDD ψ, there exists a well-trained faithful OBDD encoding θ s.t. the evaluation $\text{ESat}(\theta, \omega)$ of the satisfaction relation between θ and ω coincides with that between the example ω and the OBDD. In this case, an OBDD can be interpreted directly from the OBDD encoding without any gap between them on classification performance.

Theorem 4.5. *Let ψ be an OBDD of depth H', and θ be a faithful OBDD encoding of depth $H \geq H'$ such that $\text{decode}(\theta)$ and ψ are isomorphic. Then, for any example $\omega = [l_1, \cdots, l_m]$, it holds that $\text{ESat}(\theta, \omega) = 1$ if $\omega \models \psi$, and $\text{ESat}(\theta, \omega) = 0$ otherwise.*

Proof Sketch. Let $\psi = (\mathcal{V}', \mathcal{T}', \mathcal{E}_l', \mathcal{E}_r')$. Let $\varphi = \text{decode}(\theta) = (\mathcal{V}, \mathcal{T}, \mathcal{E}_l, \mathcal{E}_r)$. We first prove that $\omega \models \psi$ only if $\text{ESat}(\theta, \omega) = 1$. Assume that $\omega \models \psi$. There is a path κ of length $1 \leq n \leq H'$ from the root v_1' of ψ to the $\top$-leaf such that κ is in consistence with ω. Since ψ and φ are isomorphic, we get that the root v_1 of φ can reach the $\top$-leaf in n step in consistence with ω. By Definition 6, in θ, v_1 can reach v_{N-1} in n step in consistence with ω. As θ is a faithful OBDD encoding and $1 \leq n \leq H$, it follows from Lemma 2 that $((\mathbf{r}^{\omega})^{(H)})_{N-1} = 1$ (i.e., $\text{ESat}(\theta, \omega) = 1$).

We now prove that $\omega \not\models \psi$ only if $\text{ESat}(\theta, \omega) = 0$ by contraposition. Assume that $\text{ESat}(\theta, \omega) \neq 0$. Since θ is a faithful OBDD encoding, it can be verified that $\text{ESat}(\theta, \omega) = 1$. By Lemma 2 and Definition 6, v_1 can reach v_{N-1} ($\top$-leaf) in n step in consistence with ω for some $1 \leq n \leq H$. Since ψ and φ are isomorphic, we get that the root v_1' of ψ can reach $\top$-leaf in n step in consistence with ω, i.e., $\omega \models \psi$. □

5 Learning OBDDs by OBDD Encoding

Exploiting the (faithful) OBDD encoding, a gradient-based OBDD learning approach is developed.

We first build network structure of OBDD-NET parameterized by an OBDD encoding θ via a softmax operation as shown in Equation (2) [3]. It fulfills constraints 2 and 3 of the faithfulness defined in Definition 5, which captures the basic structural properties of a BDD.

$$(\theta_{\text{left}})_{i,j} = \frac{e^{(\Gamma_{\text{left}})_{i,j}}}{(\eta_{\text{left}})_i}, (\theta_{\text{right}})_{i,j} = \frac{e^{(\Gamma_{\text{right}})_{i,j}}}{(\eta_{\text{right}})_i}, (\theta_{\text{dec}})_{i,k} = \frac{e^{(\Gamma_{\text{dec}})_{i,k}}}{(\eta_{\text{dec}})_i}, \quad (2)$$

where $(\eta_{\text{left}})_i = \sum_{\substack{1 \leq i < j \leq N \\ \text{lev}(i) < \text{lev}(j)}} e^{(\Gamma_{\text{left}})_{i,j}}$, $(\eta_{\text{right}})_i = \sum_{\substack{1 \leq i < j \leq N \\ \text{lev}(i) < \text{lev}(j)}} e^{(\Gamma_{\text{right}})_{i,j}}$, $(\eta_{\text{dec}})_i = \sum_{1 \leq k \leq |\mathcal{F}|} e^{(\Gamma_{\text{dec}})_{i,k}}$.

Then, we train the parameterized neural network to jointly explore an assignment of the parameters approximately faithful to the OBDD representation and optimize classification. The basic optimization objective to maximize the consistency between the dataset $\mathcal{E}$ and θ, is formulated as $\mathcal{L}_0$. To overcome the vanishing gradient when training, we replace the function σ_{01} used in $\text{ESat}(\theta, \omega)$ of $\mathcal{L}_0$ with its differentiable approximation σ_{01}' defined as follow: $\sigma_{01}'(x) = c_0 x$ if $x < 0$, $\sigma_{01}'(x) = c_0 x + 1 - c_0$ if $x > 1$, otherwise $\sigma_{01}'(x) = x$, where $c_0 = 0.01$ a small positive constant. Besides, the remaining constraints 1 and 4 of the faithfulness are characterized as regularization loss $\mathcal{L}_1$ to $\mathcal{L}_3$.

$$\begin{aligned}
\mathcal{L}_0 &= \frac{1}{|\mathcal{E}|} \cdot \textstyle\sum_{\omega \in \mathcal{E}} (\text{ESat}(\theta, \omega) - \text{lab}(\omega))^2, \\
\mathcal{L}_1 &= \textstyle\sum_{k=1}^{|\mathcal{F}|} \text{Relu}(\sum_{i=1}^{H} (\theta_{\text{dec}})_{i,k} - 1), \\
\mathcal{L}_2 &= -\textstyle\sum_{i=1}^{H} \sum_{k=1}^{|\mathcal{F}|} ((\theta_{\text{dec}})_{i,k} \cdot \log((\theta_{\text{dec}})_{i,k} + c_1)), \\
\mathcal{L}_3 &= -\textstyle\sum_{i=1}^{N-2} \sum_{j=i+1}^{N} ((\theta_{\text{left}})_{i,j} \cdot \log((\theta_{\text{left}})_{i,j} + c_1) \\
&\quad + (\theta_{\text{right}})_{i,j} \cdot \log((\theta_{\text{right}})_{i,j} + c_1)),
\end{aligned} \quad (3)$$

where $c_1 = 1e\text{-}5$ is a small positive constant bias.

The regularization loss $\mathcal{L}_1$ is obtained from the condition 4 in Definition 5 by replacing constraint like $x \leq y$ with a differentiable one $\text{Relu}(x - y)$. The regularization loss $\mathcal{L}_2$ and $\mathcal{L}_3$ are used to approximately fulfill the constraint 1 by bootstrapping binarization of assignment of the involved parameters. The final joint optimization objective for training OBDD-NET is: $\mathcal{L} = \mathcal{L}_0 + \sum_{i=1}^{3} \alpha_i \mathcal{L}_i$, where α_i is the regularization coefficient.

Finally, we interpret an OBDD from the trained model (which corresponds to an approximately faithful OBDD enconding). As shown in Algorithm 1, we first convert it to a faithful one, and then an OBDD representation can be acquired by calling the routine decode defined in Definition 6. To achieve faithfulness, it is sufficient to realize both the constraints 1 and 4 in Definition 5, that is, to binarize the assignment of all the parameters in θ while ensuring that there is no $1 \leq k \leq |\mathcal{F}|$ s.t. two different parameters $(\theta_{\text{dec}})_{i,k}$ and $(\theta_{\text{dec}})_{j,k}$ are simultaneously assigned the value 1. This can be done by the following step: (1) reset the maximum in each $(\theta_{\text{left}})_i$ and $(\theta_{\text{right}})_i$ to 1 and all other values to 0s, allowing to generate a BDD-style topological DAG with root v_1; (2) calculate the involved vertices $\mathcal{V}$ and the levels L; (3) iteratively assign a unique feature f_k to each level of L in turn, by selecting the one with maximal value $(\theta_{\text{dec}})_{i,k}$ from the available features (Lines 4 to 7). Similar operation for the remaining levels is done by the following line. After decoding an OBDD representation from θ , the procedure reduce (Line 10) first applies the dd package [4] to produce a preliminary ROBDD (but in a slightly different OBDD form with complement edges), and then converts it back to the simple form (without complement edges) adopted in this paper. In this way, we obtain a final ROBDD classifier.

[3] We remark that in our implement, a more complicate variant of the softmax operation, named Gumbel Softmax function [16], is used to promote structural exploration. Gumbel softmax allows to approximate a discrete categorical distribution. Its advantage on gradient-based structure learning has been demonstrated in [27].

[4] https://github.com/tulip-control/dd/tree/main

Table 1: The statistics of the datasets. $|\mathcal{E}|$ indicates the number of examples; $|\mathcal{F}_{org}|$ indicates the number of original features; $|\mathcal{F}|$ indicates the number of binarized features; $|\mathcal{E}^+|/|\mathcal{E}|$ indicates the percentage of positive examples in the dataset.

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{F}_{org}\rvert$	$\lvert\mathcal{F}\rvert$	$\lvert\mathcal{E}^+\rvert/\lvert\mathcal{E}\rvert$
anneal	812	42	89	0.77
tic-tac-toe	958	9	27	0.65
car	1,728	6	21	0.30
splice-1	3,190	60	287	0.52
kr-vs-kp	3,196	36	73	0.52
hypothyroid	3,247	43	86	0.91
christine	5,418	1636	4031	0.50
musk2	6,174	166	1829	0.15
mushroom	8,124	21	112	0.52
heloc	9,488	23	87	0.46
magic04	10,167	10	79	0.50
adult	21,647	14	144	0.24
secondary mushroom	22,037	20	157	0.55
bank marketing	38,424	20	125	0.12
higgs	97,276	28	93	0.53
weatherAUS	141,970	22	246	0.22
BNG (labor)	615,870	16	95	0.57
BNG (credit-g)	988,973	20	90	0.70

6 Experiments

Baselines. We compare OBDD-NET with three SOTA OBDD learners: OODG [19] for heuristic approach, and MaxSAT-BDD[15] and Shati et al. [23]'s approach for exact ones. For OODG, as we did not find the publicly available implementation of the authors, we use its recent implementation [5] of [15] instead.

Datasets. We take 10 small and 8 large-scale public binary classification datasets from CP4IM[6], UCI [9], Open-ML [25] and Kaggle repository [7].For small datasets, we follow those in [15] that contain more than 800 examples, while including three additional datasets from UCI and OpenML. Besides, as in [12], for assessing scalability on large-scale classification problems, we also consider 8 significantly larger datasets with increasing numbers of examples from about 10,000 to 1,000,000. The features in the datasets are discretized by entropy-based discretization and then binarized with the one-hot encoding to get binary features. We perform these discretization and binarization using the tool Orange [8], and then remove duplicate examples. The statistics of these datasets is shown in Table 1.

6.1 Experimental Settings

All experiments were conducted on a Linux system equipped with an Intel(R) Xeon(R) Gold 5218R CPU with 2.1 GHz and 64 GB RAM and a NVIDIA RTX 3090 GPU. For MaxSAT-BDD and Shati et al.'s approach, we follow their settings and apply the Loandra [2] MaxSAT solver to solve their encoding instances. For our approach, we use Adam [17] to optimize the parameters in our model. We perform grid search for the hyper-parameters tuning, the best setting for them is as follows: the learning rate is set to 0.2, the number of epochs is set to 3000 for small datasets and 800 for large datasets, the batch size is set to 512, the regularization coefficients $\alpha_1, \alpha_2, \alpha_3$ are set to 1, 1e-6, 1e-6, respectively.

The depth H of OBDD-NET and the baselines are all set to 6. For all datasets, we split them uniformly and perform cross-validation evaluations of each classifier on the same split datasets. For all classifiers including our approach, the time limit for training is 900 seconds for each small dataset, and 1,800 seconds for each large dataset [8].

6.2 Performance Measurement

For each dataset, we perform 5-fold cross-validation and report the average testing performance over the split 5 pairs of training-testing sets. Specifically, we use (average) testing accuracy and macro-F1 score for classification performance evaluation, and (average) size of the final produced ROBDD for rule-size evaluation. Note that, for OODG which generates non-reduced OBDDs, we apply the post-processing to produce ROBDDs as in our approach. In this way, the final OBDDs we compared are all ROBDDs and in the same object.

6.3 Classification Performance and Rule Size

Comparison of 5-fold cross validated (average) testing accuracy (%), macro-F1 score (%) and size among the generated ROBDDs on the datasets is shown in Table 2. Our approach outperforms the SOTAs on 14 out of 18 datasets on classification performance.

First, OBDD-NET achieves competitive testing accuracy and macro-F1 on most small datasets compared to the two MaxSAT-based baselines, expect for the three smallest ones (containing fewer than 2000 examples). The underperformance of OBDD-NET on these very small datasets aligns with the intuition that neural methods typically require more data while constraint programming solver-based methods are more efficient when search space is small. We can observe that OBDD-NET outperforms them on datasets with more examples and features. For OODG, as observed in [15], despite occasional good performance, it usually produces classifiers with poor classification performance. A timeout also occurs in OODG when solving the christine and musk2 datasets, which have more than 1000 features, due to the large exploration space for feature selection.

For the large datasets, OBDD-NET achieves significantly higher prediction quality than the three SOTAs, with improvements more than 3% on accuracy and 4% on macro-F1 in most cases. For the rule-size, OODG performs best, followed by OBDD-NET and Shati's approach. However, OODG has significant poor prediction quality on most of datasets. To sum up, OBDD-NET achieves competitive classification and rule-size performance w.r.t. the SOTA OBDD learners.

6.4 Scalability

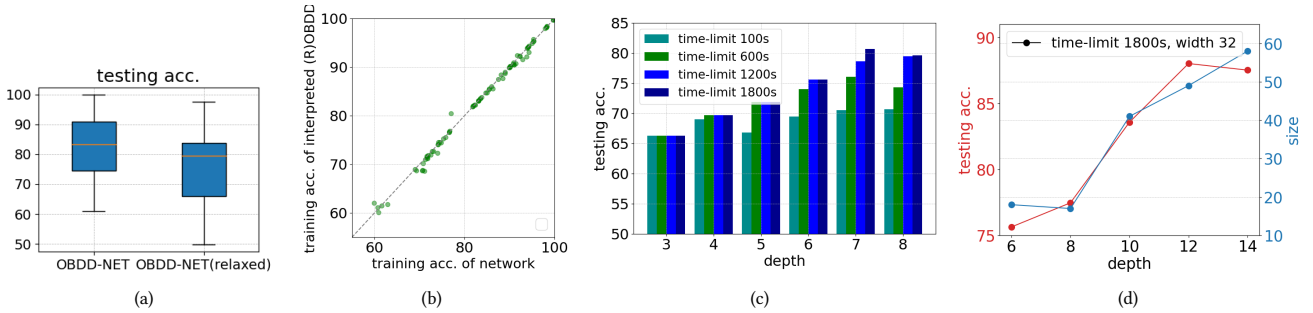
From Table 2, we observe that the three SOTAs, especially OODG and MaxSAT-BDD, have significantly lower classification performance than OBDD-NET on the large datasets. In particular, they fail to handle the last three datasets in the time limit, which contains hundreds of thousands to a million of examples. For the two MaxSAT-based SOTAs, they need to explicitly encode the satisfaction towards the whole dataset, leading to the linear growth of encoding size (i.e., the total number of literals in the encoded MaxSAT instance)

[5] https://gitlab.laas.fr/hhu/bddencoding
[6] https://dtai.cs.kuleuven.be/CP4IM/datasets/
[7] https://www.kaggle.com/datasets

[8] The time limits for MaxSAT-BDD and Shati et al.'s approach were set by their APIs, while for OODG with no relevant API was provided, we do that by setting the timeout threshold for its execution.

Table 2: Comparison of 5-fold cross validated (average) testing accuracy (%), macro-F1 score (%) and size among the generated ROBDDs by OBDD-NET and other OBDD learners. "OOT" abbreviates timeout. Numbers in bold means the best performing results.

Datasets	# examples	OBDD-NET			MaxSAT-BDD			Shati's approach			OODG		
		Acc.	Macro-F1	Size	Acc.	Macro-F1	Size	Acc.	Macro-F1	Size	Acc.	Macro-F1	Size
anneal	812	83.37	70.23	8.4	**84.85**	**75.98**	15.2	83.25	74.18	9.8	81.4	60.57	**7.2**
tic-tac-toe	958	78.38	74.8	19.0	82.25	79.39	24.8	**83.61**	**80.79**	22.8	79.22	71.63	**14.0**
car	1,728	91.15	89.28	12.0	93.4	92.48	13.4	**93.87**	**93.07**	9.6	91.9	91.0	**8.0**
splice-1	3,190	**93.76**	**93.76**	11.4	67.81	65.86	13.4	75.8	75.63	13.4	63.39	56.19	**11.0**
kr-vs-kp	3,196	94.56	94.55	9.2	94.46	94.45	12.2	**94.59**	**94.57**	10.6	75.6	73.36	**4.8**
hypothyroid	3,247	**98.15**	**94.38**	8.2	98.09	94.18	13.6	98.06	94.08	10.8	91.35	49.05	**6.8**
christine	5,418	**67.66**	**67.64**	**12.4**	56.98	56.02	20.4	55.04	54.88	19.5	OOT	OOT	OOT
musk2	6,174	**90.51**	**77.71**	**10.4**	82.22	65.44	15.4	61.76	50.55	12.0	OOT	OOT	OOT
mushroom	8,124	**99.8**	**99.8**	12.8	98.77	98.77	16.4	99.63	99.63	12.0	99.61	99.61	**10.2**
heloc	9,488	**71.54**	**71.19**	11.8	56.26	54.45	19.8	65.2	64.58	16.2	46.26	31.68	**3.2**
magic04	10,167	**73.98**	**73.91**	11.2	53.39	42.56	17.2	72.96	72.51	10.0	50.25	33.44	**3.8**
adult	21,647	**83.16**	**68.17**	8.2	59.29	49.98	14.0	79.34	63.97	10.4	24.23	19.85	**4.4**
sec mushroom	22,037	**75.83**	**75.28**	16.4	56.96	55.98	18.2	64.3	63.18	14.4	56.74	40.37	**9.0**
bank marketing	38,424	**89.97**	**70.63**	9.0	74.99	53.71	14.2	88.81	60.37	18.4	11.97	10.7	**8.0**
higgs	97,276	**61.03**	**60.14**	**8.4**	OOT	OOT	OOT	50.69	45.34	13.33	52.9	34.64	11.8
weatherAUS	141,970	**82.05**	**65.03**	**7.0**	OOT	OOT	OOT	OOT	OOT	OOT	OOT	OOT	OOT
BNG (labor)	615,870	**88.2**	**87.87**	**10.8**	OOT	OOT	OOT	OOT	OOT	OOT	OOT	OOT	OOT
BNG (credit-g)	988,973	**72.29**	**62.73**	**6.6**	OOT	OOT	OOT	OOT	OOT	OOT	OOT	OOT	OOT
# best		**14**	**14**	6	1	1	0	3	3	0	0	0	**12**



(a) (b) (c) (d)

Figure 3: (a) 5-fold cross-validated (average) testing accuracy of the generated ROBDDs by OBDD-NET and OBDD-NET(relaxed). (b) Training accuracy of networks of OBDD-NET and that of the interpreted ROBDDs. (c) Testing accuracy of the generated ROBDDs by OBDD-NET with different depths on the sec-mushroom dataset. (d) Testing accuracy and size of the generated ROBDDs by OBDD-NET with different depths under fixed width on the sec-mushroom dataset.

w.r.t. dataset size. As shown in Figure 1(b) in Introduction, when applied to large-scale datasets, the encoding size of them become extremely large (e.g., greater than 100 million for both approaches when dataset size reaches 100 thousand), being quite challenging to be solved by modern MaxSAT solvers. For the heuristic-based approach OODG, it relies on a lot of computation of mutual information on the dataset for feature selection and turns out to be time consuming when handling the large-scale datasets. In contrast, OBDD-NET does not suffer from these problems and is able to scale to million-size datasets. These exhibit better scalability of OBDD-NET.

6.5 Ablation Study

To evaluate the effect of the faithfulness of the OBDD-NET on classification performance, we build a relaxed model of OBDD-NET, namely OBDD-NET(relaxed), by replacing the softmax operation in Equation (2) with the sigmoid operation and setting all the regularization coefficient α_i to 0. It can be view as a neural network model with much weaker faithfulness guarantee, since applying the sigmoid operation on a parameter assignment commonly produces only a trivial OBDD encoding far from fulfilling constraints 2 and 3 of the faithfulness. We then conduct 5-fold cross-validation experiments for OBDD-NET and OBDD-NET(relaxed) on all datasets as before. As can be seen from Figure 3(a), OBDD-NET(relaxed) have significant poorer classification performance than OBDD-NET.This confirms the effectiveness of the faithfulness in OBDD-NET.

Besides, Figure 3(b) compares the training accuracy achieved by the network and the interpreted ROBDD of OBDD-NET on the 90 runs (5 folds × 18 datasets). We can find that the performance gap is small, which suggests that the trained network is most often highly faithful to the OBDD representation.

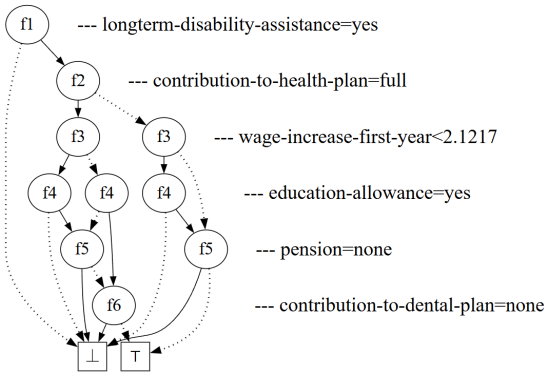


Figure 4: A ROBDD learned on the BNG(labor) dataset.

6.6 Impact of depth

We conduct experiments with various depths H under different time-limits on the sec-mushroom dataset. We use 80% of the examples for training and the left for testing. From Figure 3(c), as H grows, the accuracy first increases generally then remains stable. This is because higher H allows to build a complicate OBDD classifier with more different features, but is usually harder to train due to exponentially greater search space. The exponential growth of the encoding size causes OBDD-NET fail to handle OBDD learning problems with large depth. Note that, the two competitive SOTAs also face the challenge of exponential explosion [15, 23]. In practice, we can alleviate the affect via taking a smaller width $W < 2^{H-1}$. From Figure 3(d), under $W = 32$, OBDD-NET can handle a greater depth and learn a more complicate ROBDD with better prediction quality.

6.7 Case study

In the BNG(labor) dataset, the classifier is to predict whether a given labor contract is acceptable. Although ROBDD classifiers are learned on binarized features, in practice we find that it is usually interpretable enough due to the small depth (i.e., the number of used features). Figure 4 shows an ROBDD classifier learned by OBDD-NET, which has depth 6 and size 12. The feature ordering is ['longterm-disability-assistance=yes', 'contribution-to-health-plan=full', 'wage-increase-first-year<2.1217', 'education-allowance=yes', 'pension =none', 'contribution-to-dental-plan=none']. There are 5 paths from the root to the $\top$-leaf. For instance, the second right-most path says a contract that provides long-term disability assistance without full contribution to health-plan, with a first-year wage increase below 2.1217%, but offers education allowance and pension, is acceptable.

From the classifier, an essential feature is 'longterm-disability-assistance=yes'. That is, all contracts to be acceptable should offer long-term disability assistance. Consider now a contract that includes long-term disability benefits, partial health-plan coverage, a first-year wage increase of at least 2.1217%, as well as education allowance and pension provisions, which corresponds to a (partial) example ω: $[f_1, \neg f_2, \neg f_3, f_4, \neg f_5]$. It can be verified that this contract is acceptable by the classifier shown in Figure 4. One may wonder why such an example triggers the positive decision of the classifier. To this end, we can compute the *complete reason* and the *sufficient reason* (aka. *prime implicant explanation*) behind this decision, where the former captures the necessary and sufficient condition to trigger a decision [7]. The resulting complete reason is $f_1 \wedge \neg f_2 \wedge \neg f_5 \wedge (\neg f_3 \vee f_4)$, and we can get two sufficient reasons: (1) $f_1 \wedge \neg f_2 \wedge \neg f_3 \wedge \neg f_5$ and (2) $f_1 \wedge \neg f_2 \wedge f_4 \wedge \neg f_5$. In other words, any contract that offers long-term disability assistance, partial health-plan coverage, pensions, as well as at least one of the following: a minimum wage increase of 2.1217% in the first year or an education allowance, will be accepted by this classifier.

Note that the computation of the complete reason for a decision can be done in linear time for an OBDD classifier [7]. Recent studies [7, 24, 32, 33] have fully demonstrated the effectiveness of OBDD representations in facilitating the computation of decision explanation (including complete reason analysis and prime implicant explanations) as well as quantitative robustness analysis. We will no longer illustrate them here. Please refer to them for details.

7 Discussions and Future Work

The limitations of the current design of OBDD-NET are two-fold. (1) OBDD-NET is more suitable for learning OBDDs of small depth, similar to other successful applications of gradient-based structure learning techniques in mining succinct logical rules, e.g., Linear Temporal Logic on finite traces (LTLf) formula [26] and Finite-State Automata (FSA) [20]. In practice, it is usually effective enough, as a relatively complex formula could possibly be simplified to a more concise form. (2) OBDD-NET is limited to binary classification with binarized features, relying on preprocessing with conventional feature binarization approaches. This also hinders the extension to end-to-end learning more general decision diagrams, due to non-differentiability of these binarization approaches. Future work will extend our approach to suit classification with real-valued features and Multi-valued Decision Diagrams (MDDs), by integrating some advanced differentiable binarization techniques [31].

8 Conclusions

In this paper we have developed an end-to-end OBDD learning neural model named OBDD-NET, which exploits an OBDD encoding method to parameterize a neural network and allows the utilization of gradient descent optimization for efficient learning. Experimental results on 10 small and 8 large datasets show that OBDD-NET scales better than existing OBDD learners with competitive performance.

Acknowledgments

Our experiments were conducted on an RTAI cluster, which is supported by School of Computer Science and Engineering as well as Institute of Artificial Intelligence in Sun Yat-sen University. This article was supported by National Natural Science Foundation of China (No. 62276284), Guangdong Basic and Applied Basic Research Foundation (No. 2023A1515011470, 2022A1515011355), Guangzhou Science and Technology Project (No. 202201011699), Shenzhen Science and Technology Program (KJZD2023092311405902).

GenAI Usage Disclosure

No GenAI tools were used in any stage of the research, nor in the writing.

References

[1] Sheldon B. Akers. 1978. Binary Decision Diagrams. *IEEE Trans. Comput.* 27, 6 (1978), 509–516.

[2] Jeremias Berg, Emir Demirovic, and Peter J. Stuckey. 2019. Core-Boosted Linear Search for Incomplete MaxSAT. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR-2019)*. 39–56.

[3] Leo Breiman, J. H. Friedman, Richard A. Olshen, and C. J. Stone. 1984. *Classification and Regression Trees*.

[4] Bryant. 1986. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Trans. Comput.* 35, 8 (1986), 677–691.

[5] Gianpiero Cabodi, Paolo E. Camurati, Alexey Ignatiev, João Marques-Silva, Marco Palena, and Paolo Pasini. 2021. Optimizing Binary Decision Diagrams for Interpretable Machine Learning Classification. In *Proceedings of the 24th Design, Automation & Test in Europe Conference & Exhibition (DATE-2021)*. IEEE, 1122–1125.

[6] Gianpiero Cabodi, Paolo E. Camurati, João Marques-Silva, Marco Palena, and Paolo Pasini. 2024. Optimizing Binary Decision Diagrams for Interpretable Machine Learning Classification. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 43, 10 (2024), 3083–3087.

[7] Adnan Darwiche and Auguste Hirth. 2023. On the (Complete) Reasons Behind Decisions. *J. Log. Lang. Inf.* 32, 1 (2023), 63–88.

[8] Janez Demsar, Tomaz Curk, Ales Erjavec, Crtomir Gorup, Tomaz Hocevar, Mitar Milutinovic, Martin Mozina, Matija Polajnar, Marko Toplak, Anze Staric, Miha Stajdohar, Lan Umek, Lan Zagar, Jure Zbontar, Marinka Zitnik, and Blaz Zupan. 2013. Orange: data mining toolbox in python. *J. Mach. Learn. Res.* 14, 1 (2013), 2349–2353.

[9] Dheeru Dua and Casey Graff. 2017. doi:ml

[10] Alexandre M. Florio, Pedro Martins, Maximilian Schiffer, Thiago Serra, and Thibaut Vidal. 2023. Optimal Decision Diagrams for Classification. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence (AAAI-2023)*. 7577–7585.

[11] Kun Gao, Katsumi Inoue, Yongzhi Cao, and Hanpin Wang. 2022. Learning First-Order Rules with Differentiable Logic Program Semantics. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*. ijcai.org, 3008–3014.

[12] Bishwamittra Ghosh, Dmitry Malioutov, and Kuldeep S. Meel. 2022. Efficient Learning of Interpretable Classification Rules. *J. Artif. Intell. Res.* 74 (2022), 1823–1863.

[13] Claire Glanois, Zhaohui Jiang, Xuening Feng, Paul Weng, Matthieu Zimmer, Dong Li, Wulong Liu, and Jianye Hao. 2022. Neuro-Symbolic Hierarchical Rule Induction. In *Proceedings of the 39th International Conference on Machine Learning (ICML-2022)*. 7583–7615.

[14] Hao Hu. 2022. *Interpretable Machine Learning Models via Maximum Boolean Satisfiability*. Ph. D. Dissertation. INSA Toulouse, France.

[15] Hao Hu, Marie-José Huguet, and Mohamed Siala. 2022. Optimizing Binary Decision Diagrams with MaxSAT for Classification. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI-2022)*. 3767–3775.

[16] Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical Reparameterization with Gumbel-Softmax. In *Proceedings of the 5th International Conference on Learning Representations (ICLR-2017)*.

[17] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR-2015)*. 1–15.

[18] Ron Kohavi. 1994. Bottom-Up Induction of Oblivious Read-Once Decision Graphs. In *Proceedings of the 7th European Conference on Machine Learning (ECML-1994)*, Vol. 784. 154–169.

[19] Ron Kohavi and Chia-Hsin Li. 1995. Oblivious Decision Trees, Graphs, and Top-Down Pruning. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-1995)*. 1071–1079.

[20] Weilin Luo, Tingchen Han, Junming Qiu, Hai Wan, Jianfeng Du, Bo Peng, Guohui Xiao, and Yanan Liu. 2025. NADA: Neural Acceptance-Driven Approximate Specification Mining. *Proc. ACM Softw. Eng.* 2 (2025), 1795–1817.

[21] Litao Qiao, Weijia Wang, and Bill Lin. 2021. Learning Accurate and Interpretable Decision Rule Sets from Neural Networks. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI-2021)*. 4303–4311.

[22] J. Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*.

[23] Pouya Shati, Eldan Cohen, and Sheila A. McIlraith. 2023. SAT-Based Learning of Compact Binary Decision Diagrams for Classification. In *Proceedings of the 29th International Conference on Principles and Practice of Constraint Programming (CP-2023)*, Vol. 280. 33:1–33:19.

[24] Andy Shih, Adnan Darwiche, and Arthur Choi. 2019. Verifying Binarized Neural Networks by Angluin-Style Learning. In *Proceedings of the 22nd Theory and Applications of Satisfiability Testing (SAT-2019)*, Vol. 11628. 354–370.

[25] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luís Torgo. 2013. OpenML: networked science in machine learning. *SIGKDD Explor.* 15 (2013), 49–60.

[26] Hai Wan, Pingjia Liang, Jianfeng Du, Weilin Luo, Rongzhen Ye, and Bo Peng. 2024. End-to-End Learning of LTLf Formulae by Faithful LTLf Encoding. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence (AAAI-2024)*. 9071–9079.

[27] Xuan Wang, Zheyu Yan, Chang Meng, Yiyu Shi, and Weikang Qian. 2023. DASALS: Differentiable Architecture Search-Driven Approximate Logic Synthesis. In *IEEE/ACM International Conference on Computer Aided Design (ICCAD-2023)*. 1–9.

[28] Zhuo Wang, Wei Zhang, Ning Liu, and Jianyong Wang. 2021. Scalable Rule-Based Representation Learning for Interpretable Classification. In *Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS-2021)*. 30479–30491.

[29] Zhuo Wang, Wei Zhang, Ning Liu, and Jianyong Wang. 2024. Learning Interpretable Rules for Scalable Data Representation and Classification. *IEEE Trans. Pattern Anal. Mach. Intell.* 46, 2 (2024), 1121–1133.

[30] Fan Yang, Zhilin Yang, and William W. Cohen. 2017. Differentiable Learning of Logical Rules for Knowledge Base Reasoning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NeurIPS-2017)*. 2319–2328.

[31] Wei Zhang, Yongxiang Liu, Zhuo Wang, and Jianyong Wang. 2023. Learning to Binarize Continuous Features for Neuro-Rule Networks. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence (IJCAI-2023)*. 4584–4592.

[32] Yedi Zhang, Zhe Zhao, Guangke Chen, Fu Song, and Taolue Chen. 2021. BDD4BNN: A BDD-Based Quantitative Analysis Framework for Binarized Neural Networks. In *Proceedings of the 33rd International Conference on Computer Aided Verification (CAV-2021)*, Vol. 12759. 175–200.

[33] Yedi Zhang, Zhe Zhao, Guangke Chen, Fu Song, and Taolue Chen. 2023. Precise Quantitative Analysis of Binarized Neural Networks: A BDD-based Approach. *ACM Trans. Softw. Eng. Methodol.* 32, 3 (2023), 62:1–62:51.