Learning Small Decision Trees With Few Outliers: A Parameterized Perspective

Harmender Gahlawat* **and Meirav Zehavi***

Ben-Gurion University of the Negev, Beersheba, Israel.
harmendergahlawat@gmail.com and zehavimeirav@gmail.com

Abstract

Decision trees are a fundamental tool in machine learning for representing, classifying, and generalizing data. It is desirable to construct "small" decision trees, by minimizing either the *size* (s) or the *depth* (d) of the *decision tree* (DT). Recently, the parameterized complexity of DECISION TREE LEARNING has attracted a lot of attention. We consider a generalization of DECISION TREE LEARNING where given a *classification instance* E and an integer t, the task is to find a "small" DT that disagrees with E in at most t examples. We consider two problems: DTSO and DTDO, where the goal is to construct a DT minimizing s and d, respectively. We first establish that both DTSO and DTDO are W[1]-hard when parameterized by $s+\delta_{max}$ and $d+\delta_{max}$, respectively, where δ_{max} is the maximum number of features in which two differently labeled examples can differ. We complement this result by showing that these problems become FPT if we include the parameter t. We also consider the kernelization complexity of these problems and establish several positive and negative results for both DTSO and DTDO.

Introduction

Decision trees is a fundamental tool in the realm of machine learning with applications spanning classification, regression, anomaly detection, and recommendation systems (Larose and Larose 2014; Murthy 1998; Quinlan 1986). Because of their ability to represent complex labeled datasets through a sequence of simple binary decisions, they provide a highly interactive and interpretable model for data representation (Darwiche and Hirth 2020; Doshi-Velez and Kim 2017; Goodman and Flaxman 2017; Lipton 2018; Monroe 2018). It is of interest to have *small trees*, as they require fewer tests to classify data and are easily interpretable. Moreover, small trees are expected to generalize better to new data, i.e., minimizing the number of nodes reduces the chances of overfitting (Bessiere, Hebrard, and O'Sullivan 2009). However, as problem instances grow in size and complexity, efficiently learning small decision trees becomes a challenging task. In particular, it is NP-hard to decide if a given dataset can be represented using a decision tree of a certain size or depth (Laurent and Rivest 1976). To deal with this complexity barrier, implementations of several heuristics based on constraint-based and SAT-based techniques have been proposed to learn small decision trees (Bessiere, Hebrard, and O'Sullivan 2009; Narodytska et al. 2018; Avellaneda 2020; Schidler and Szeider 2021). In fact, the classical CART heuristic herein is among the top 10 algorithms of data mining chosen by the ICDM (Steinberg and Colla 2009; Wu et al. 2008).

Despite these efforts, our understanding of the computational complexity of learning small decision trees is limited. Recently, research of the parameterized complexity of learning small decision trees has attracted a lot of attention (Eiben et al. 2023; Kobourov et al. 2023; Ordyniak and Szeider 2021; Komusiewicz et al. 2023). Parameterized complexity theory provides a framework for classifying computational problems based on both their input size and a parameter that captures the inherent difficulty of the problem. The key notion in parameterized complexity is that of *fixed parameter tractability* (FPT), which restricts the exponential blowup in the time to be a function only of the chosen *parameter*. Due to its efficacy, parameterized complexity theory has been extensively used to understand the complexity of several problems arising in AI and ML; see, e.g. (Bäckström et al. 2012; Bessiere et al. 2008; Bredereck et al. 2017; Ganian et al. 2018; Gaspers et al. 2017). An important framework within parameterized complexity is that of *kernelization*, polynomial-time preprocessing with a parametric guarantee. Due to its profound impact, kernelization was termed "the lost continent of polynomial time" (Fellows 2006). Kernelization is specifically useful in practical applications as it has shown tremendous speedups in practice (Gao 2009; Guo and Niedermeier 2007; Niedermeier and Rossmanith 2000; Weihe 1998).

The input to a decision tree learning algorithm is a *classification instance* (CI) E, which is a set of *examples* labeled either positive or negative, where each *example* is defined over the same set of *features* F that can take values from a linearly ordered domain D. We define these concepts (along with other definitions) in Section . Now, in DTS (resp., DTD), the input is a positive integer s (resp., d) and a CI E, and the goal is to decide whether there exists some *decision tree* (DT) of *size* at most s (resp., *depth* at most d) that "*classifies*" each example of E correctly. The *size* of a DT T is the number of *test nodes* in T and the *depth* of a T is the

*These authors contributed equally.

maximum number of test nodes on a leaf to root path of T. Some parameters of the input that are of interest, in addition to s and d, are: $|F|$ being the number of features in E, D_{max} being the maximum value a feature of an example can take, and δ_{max} being the maximum number of features a positive and a negative example can differ in. All of these features are often small compared to the size of the input instance and hence are good choices to achieve FPT algorithms (for e.g., see Table 1 in (Ordyniak and Szeider 2021)).

We consider a generalization of DECISION TREE LEARNING where, given a CI E and an integer t, the goal is to compute a DT T of minimum size or depth such that T disagrees with at most t examples of E. Notably, even when we are allowed to have 1 *outlier* (i.e., $t = 1$), the instance used to prove W[1]-hardness of DTS and DTD (Ordyniak and Szeider 2021) admits a DT of size and depth 0. Hence, a little slack (a few outliers) can decrease the complexity of the resulting tree significantly. We term the corresponding problems for DTD and DTS as DTDO and DTSO, respectively. Finally, observe that DTDO and DTSO model DTD and DTS, respectively, when $t = 0$.

Previous Work. The parameterized complexity of learning small decision trees has attracted significant attention recently. Ordyiak and Szeider (2021) established that both DTD and DTS are W[1]-hard parameterized d and s, respectively, even for binary instances. On the positive side, they proved that DTS (resp., DTD) is FPT parameterized by $s + \delta_{max} + D_{max}$ (resp., $d + \delta_{max} + D_{max}$). Kobourov et al. (2023) established that DTS is W[1]-hard parameterized by $|F|$, XP parameterized by $|F|$, and FPT when parameterized by $s + |F|$. More recently, Eiben et al. (2023) established that both DTS and DTD are FPT parameterized by $s + \delta_{max}$ and $d + \delta_{max}$, respectively.

Our Contribution. We study the parameterized and kernelization complexity of DTDO and DTSO. The notations used in this section are definded in Section . We start by establishing that DTDO and DTSO are W[1]-hard parameterized by $d + \delta_{max}$ and $s + \delta_{max}$, respectively. For this purpose, we provide an involved reduction from PARTIAL VERTEX COVER to DTSO and DTDO.

Theorem 1. DTDO *and* DTSO *are W[1]-hard when parameterized by* $d+\delta_{max}$ *and* $s+\delta_{max}$*, respectively. Further, they are W[1]-hard parameterized by* s *and* d*, respectively, even if* $\delta_{max} \leq 3$*.*

We complement our W[1]-hardness result with FPT algorithms for DTSO and DTDO by including t as an additional parameter. We build on the algorithms provided by Eiben et al. (2023) for DTS and DTD parameterized $s + \delta_{max}$ and $d + \delta_{max}$, respectively.

Theorem 2. DTDO *and* DTSO *are* FPT *parameterized by* $d + \delta_{max} + t$ *and* $s + \delta_{max} + t$*, respectively.*

Next, we consider the kernelization complexity of DTSO and DTDO. Since DTD and DTS are the special cases of DTDO and DTSO when $t = 0$, we prove negative kernelization results for DTS and DTD, which imply the same for DTSO and DTDO. We first observe that the reduction provided by Ordyniak and Szeider (2021) from HITTING SET (HS) to DTS and DTD to prove W[2]-hardness is a polynomial parameter transformation. Since HS is unlikely to admit a polynomial compression parameterized by $k + |\mathcal{U}|$ (Proposition 2), we have the following theorem.

Observation 1. DTS *and* DTD *parameterized by* $s+|F|+D_{max}$ *and* $d + |F| + D_{max}$*, respectively, do not admit a polynomial compression even when* $D_{max} = 2$*, unless* NP $\subseteq$ coNP/poly*.*

Since HS admits a kernel of size at most $k^{\mathcal{O}(\Delta)}$ (Fomin et al. 2019) where Δ is the *arity* of the HS instance, it becomes interesting to determine the kernelization complexity of DTSO and DTDO when δ_{max} is a fixed constant (as $\delta_{max} = \Delta$ in the reduction). Towards this, we establish that DTSO and DTDO admit trivial polynomial kernels parameterized by $D_{max} + |F|$ when δ_{max} is a fixed constant.

Theorem 3. DTS *and* DTD *parameterized by* $D_{max} + |F|$ *admit a trivial polynomial kernel when* δ_{max} *is a constant.*

Next, we establish the incompressibility of DTD parameterized by $d+|F|$ even when δ_{max} is a fixed constant. To this end, we provide a non-trivial AND-composition for DTD parameterized by $d + |F|$ such that $\delta_{max} \leq 3$.

Theorem 4. DTD *parameterized by* $d+|F|$ *does not admit a polynomial compression even when* $\delta_{max} \leq 3$*, unless* NP $\subseteq$ coNP/poly*.*

Let $|E|$ denote the number of examples in E. Notice that, possibly, $|F| > |E|$. Since, HS and SET COVER are dual problems, we observe that this duality can be used alongwith the reduction provided by Ordyniak and Szeider (2021) (from HS to DTD and DTS) to get a polynomial parameter transformation from SET COVER to DTS and DTD, which in turn imply their incompressibility parameterized by $|E|$.

Observation 2. *Let* $|E|$ *be the number of examples in* E*.* DTD *and* DTS *parameterized by* $|E| + D_{max}$ *do not admit a polynomial compression even when* $D_{max} \leq 2$*, unless* NP $\subseteq$ coNP/poly*.*

Preliminaries

For $\ell \in \mathbb{N}$, let $[\ell] = \{1, \ldots, \ell\}$. For the graph theoretic notations not defined explicitly here, we refer to (Diestel 2006). Proofs of the results marked with $(*)$ are removed to respect the space constraints.

Classification Problems. An example e is a function $e : F(e) \to D$ defined over a finite set $F(e)$ of *features* and a (possibly infinite) linearly ordered *domain* $D \subset \mathbb{Z}$. A *classification instance (CI)* $E = E^+ \bigcup E^-$ is the disjoint union of two sets of *positive examples* E^+ and *negative exapmles* E^-, defined over the same set of features, i.e., for $e_1, e_2 \in E$, $F(e_1) = F(e_2)$. Here, $F(E) = F(e)$ for some $e \in E$. A set of examples $X \in E$ is *uniform* if $X \in E^+$ or $X \in E^-$; otherwise, X is *non-uniform*. When it is clear from the context, we denote $F(E)$ by F. For two examples $e_1, e_2 \in E$ and some feature $f \in F$, if $f(e_1) = f(e_2)$, then we say that e_1 and e_2 *agree* on f, else, we say that e_1 and e_2 *disagree* on f. A subset $S \subseteq F$ is said to be a *support set* of E if any two examples $e^+ \in E^+$ and $e^- \in E^-$ disagree in

at least one feature of S. It is NP-hard to compute a support set of minimum size (Ibaraki, Crama, and Hammer 2011).

For two examples $e, e' \in E$, let $\delta(e, e')$ denote the set of features where e and e' disagree. Moreover, let $\delta_{max}(E) = \max_{e^+ \in E^+ \wedge e^- \in E^-} |\delta(e^+, e^-)|$ denote the maximum number of features any two non-uniform examples disagree on.

For a feature $f \in F$, let $D_E(f)$ denote the set of domain values appearing in any example of E, i.e., $D_E(f) = \{e(f) \mid e \in E\}$. Moreover, let D_{max} denote the maximum size of $D_E(f)$ over all features of E, i.e., $D_{max} = \max_{f \in F} |D_E(f)|$. If $D_{max} = 2$, then the classification instance is said to be *Boolean*. Let $E[\alpha]$ denote the set of examples that agree with the assignment $\alpha : F' \to D$, where $F' \subseteq F(E)$, i.e., $E[\alpha] = \{e \mid e(f) = \alpha(f) \wedge f \in F'\}$. Moreover, for an example e and feature f, let $e(f)$ denote the value of example e for feature f.

Decision Tree. A *decision tree* (DT) is a rooted tree T, with vertex set $V(T)$ and arc set $A(T)$, where each non-leaf node $v \in V(T)$ is labeled with a *feature* $f(v)$ and an integer *threshold* $\lambda(v)$, each non-leaf node has exactly two outgoing arcs, a *left arc* and a *right arc*, and each leaf is either a *positive leaf* or a *negative leaf*. Let $F(T) = \{f(v) \mid v \in V(T)\}$. A non-leaf node of T is referred to as a *test node*. For $v \in V(T)$, let T_v denote the subtree of T rooted at v.

Consider a CI E and DT T with $F(T) \subseteq F(E)$. For each node $v \in T$, let $T_E(v)$ be the set of all examples $e \in E$ such that for each left (resp., right) arc uw on the unique path from the root of T to v, we have $e(F(u)) \leq \lambda(u)$ (resp., $e(F(u)) > \lambda(u)$). T *correctly classifies* an example $e \in E$ if e is a positive (resp. negative) example and $e \in T_E(v)$ for a poistive (resp., negative) leaf v. Similarly, T *correctly classifies* an instance E if T correctly classifies every example of E. Here, we also say that T is a DT for E. The *size* of T, denoted by $|T|$, is the number of test nodes in T. Similarly, the *depth* of T, denoted by $dep(T)$, is the maximum number of test nodes in any root-to-leaf path of T. DTS (resp., DTD) is now the problem of deciding whether given CI E and a natural number s (resp., d), is there a DT of size at most s (resp., depth at most d) that classifies E.

Decision Trees With Outliers. Let E be a CI, and T be a DT corresponding to E, which does not necessarily classifies E. Let v be a positive (resp., negative) leaf of T. Then, we say that an example $e \in E$ is an *outlier for* T if $e \in E^+$ (resp., $e \in E^-$) and $e \in T_E(v)$ for some negative (resp., positive leaf) v. We just say that e is an outlier when it is clear from the context. Let $O(T, E)$ denote the set of outliers in E for T. Then, $|O(T, E)|$ is the number of outliers for T in E. DTSO (resp., DTDO) is now the problem of deciding whether given CI E and natural number s and t (resp., d and t), is there a DT of size at most s (resp., depth at most d) such that $|O(T, E)| \leq t$.

Parameterized Complexity. In the framework of parameterized complexity, each problem instance is associated with a non-negative integer, called a *parameter*. A parametrized problem Π is *fixed-parameter tractable* (FPT) if there is an algorithm that, given an instance (I, k) of Π, solves it in time $f(k) \cdot |I|^{\mathcal{O}(1)}$ for some computable function $f(\cdot)$. Central to parameterized complexity is the W-hierarchy of complexity classes: FPT $\subseteq$ W[1] $\subseteq$ W[2] $\subseteq \ldots \subseteq$ XP. Specifically, FPT $\neq$ W[1], unless ETH fails.

Two instances I and I' are *equivalent* when I is a Yes-instance iff I' is a Yes-instance. A *compression* of a parameterized problem Π_1 into a (possibly non-parameterized) problem Π_2 is a polynomial-time algorithm that maps each instance (I, k) of Π_1 to an equivalent instance I' of Π_2 such that size of I' is bounded by $g(k)$ for some computable function $g(\cdot)$. If $g(\cdot)$ is polynomial, then the problem is said to admit a *polynomial compression*. A *kernelization algorithm* is a compression where $\Pi_1 = \Pi_2$. Here, the output instance is called a *kernel*. Let Π_1 and Π_2 be two parameterized problems. A *polynomial parameter transformation* from Π_1 to Π_2 is a polynomial-time algorithm that, given an instance (I, k) of Π_1, generates an equivalent instance (I', k') of Π_2 such that $k' \leq p(k)$, for some polynomial $p(\cdot)$. It is well-known that if Π_1 does not admit a polynomial compression, then Π_2 does not admit a polynomial compression (Cygan et al. 2015). An *AND-composition* from a problem P to a parameterized problem Q is an algorithm that takes as input N instances $I_1, \ldots, I_N$ of P, and in time polynomial in $\sum_{j \in [N]} |I_j|$, outputs an instance $(\mathcal{I}, k)$ of Q such that: (1) $(\mathcal{I}, k)$ is a Yes-instance iff I_j is a Yes-instance for each $j \in [N]$, and (2) k is bounded by a polynomial in $\max_{j \in [N]} |I_j| + \log N$. It is well known that if P is NP-hard, then Q does not admit a polynomial compression parameterized by k, unless NP $\subseteq$ coNP/poly (Fomin et al. 2019). We refer to the books (Cygan et al. 2015; Fomin et al. 2019) for details on parameterized complexity.

Hitting Set, Set Cover, and Decision Trees. Given a family of sets $\mathcal{F}$ over some universe U and an integer k, the HITTING SET (HS) problem asks whether $\mathcal{F}$ has a *hitting set* of size k, i.e., a subset H of U of size at most k such that $X \cap H \neq \emptyset$ for every $X \in \mathcal{F}$. The *maximum arity* Δ of a HS instance is the size of a largest set in $\mathcal{F}$. Ordyniak and Szeider (2021) provided the following reduction from HS to DTS and DTD: For an instance $\mathcal{I} = (\mathcal{F}, U, k)$ of HS, let $E(\mathcal{I})$ be the CI that has a (Boolean) feature for every element in U, one positive example p with $p(u) = 0$ for every $u \in U$; and one negative example n_X for every $X \in \mathcal{F}$ such that $n_X(u) = 1$ for every $u \in X$ and $n_X(u) = 0$, otherwise. It is easy to see here that δ_{max} of $E(\mathcal{I})$ is the same as the Δ (arity) of $\mathcal{I}$ and $D_{max} = 2$. Similarly to above construction, we define $\overline{E}(\mathcal{I})$ by turning each positive example to negative example and each negative example to positive, i.e., $p \in \overline{E}^-(\mathcal{I})$ and $n_X \in \overline{E}^+(\mathcal{I})$, for $X \in \mathcal{F}$. Observe that $E(\mathcal{I})$ admits a DT of size s (resp., depth d) iff $\overline{E}(\mathcal{I})$ admits a DT of size s (resp., depth d). Ordyniak and Szeider (2021) proved the following result.

Proposition 1 ((Ordyniak and Szeider 2021)). *$\mathcal{I}$ has a hitting set of size at most k iff $E(\mathcal{I})$ (resp. $\overline{E}(\mathcal{I})$) admits a* DT *of depth (resp., size) at most k.*

In an instance of SET COVER $\mathcal{I} = (\mathcal{F}, U, k)$, we are given a family of sets $\mathcal{F}$ over some universe U and the problem asks whether there exist k sets $X_1, \ldots, X_k \in \mathcal{F}$ such that $\bigcup_{i \in [k]} X_i = U$. Since HS and SET COVER are dual problems, it is not surprising that we have the following con-

struction, which is similar to the construction in the reduction from HS to DTS and DTD: Let $\mathcal{I} = (\mathcal{F}, U, k)$ be an instance of SET COVER. We construct the following boolean CI $E(\mathcal{I})$. Here, the set $F(E(\mathcal{I}))$ (feature set of $E(\mathcal{I})$) corresponds to $\mathcal{F}$ and each negative example corresponds to an element of U; additionally, we have a positive dummy example. More formally, we have a positive example p with $p(X) = 0$ for every $X \in \mathcal{F}$; and one negative example n_u for every $u \in U$ such that $n_u(X) = 1$ if $u \in X$ and $n_u(X) = 0$, otherwise. Here, observe that $D_{max} = 2$. We have the following straightforward observation.

Observation 3. *$\mathcal{I}$ has a set cover of size k iff $E(\mathcal{I})$ admits a* DT *of depth (resp., size) at most k.*

It is well known that HS and SET COVER parameterized by $k + |U|$ are unlikely to admit a polynomial kernel:

Proposition 2 ((Dom, Lokshtanov, and Saurabh 2009))**.** HS *and* SET COVER *parameterized by $k + |U|$ do not admit a polynomial compression, unless NP $\subseteq$ coNP/poly.*

Observe that the reductions provided above from HS and SET COVER to DTS and DTD are polynomial parameter transformations, which implies the results from Observations 1 and 2.

Hardness for DTSO and DTDO

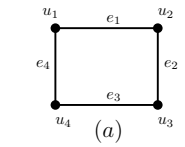
It was recently established that DTS (resp., DTD) is FPT when parameterized by $s + \delta_{max}$ (resp., $d + \delta_{max}$) (Eiben et al. 2023). In this section, we establish that DTSO (resp., DTDO) is W[1]-hard when parameterized by $s + \delta_{max}$ (resp., $d + \delta_{max}$). For this purpose, we first define the problem PARTIAL VERTEX COVER (PVC). In PVC, given a graph G and integers $k, p \in \mathbb{N}$, the goal is to decide if there is a subset $U \subseteq V(G)$ such that $|U| \leq k$ and $|\{uv \mid uv \in E(G) \wedge \{u, v\} \cap U \neq \emptyset)\}| \geq p$.

Proposition 3. *(Guo, Niedermeier, and Wernicke 2007)* PVC *parameterized by k is W[1]-hard.*

For the purpose of this section, let m denote $|E(G)|$ and n denote $|V(G)|$. Now, we provide a construction that we will be using to prove W[1]-hardness for DTSO and DTDO.

Construction. Let (G, k, p) be an instance of PVC. Fix an ordering $e_1, \ldots, e_m$ of the edges of G. We consider the following CI E' with feature set $F = V(G) \cup \{d_0\}$. See Figure 1 for a reference. Formally, for every vertex $u \in V(G)$, we have a feature u in F, along with a *dummy feature* d_0. The features corresponding to vertices in $V(G)$ are said to be *vertex features*. For each edge $e_i \in E(G)$, we add one negative example X_{2i-1} such that $X_{2i-1}(d_0) = 2i - 1$ and for a vertex feature v, $X_{2i-1}(v) = 1$ if v is an endpoint of e_i, and $X_{2i-1}(v) = 0$, otherwise. Moreover, we add m positive examples in the following manner. For $i \in [m]$, we add a positive example Y_{2i} such that $Y_{2i}(d_0) = 2i$ and for each vertex feature v, $Y_{2i}(v) = 0$.

Now, we make a CI E by taking η copies of examples in E' in the following manner. The value of η will be fixed later for the proofs of DTSO and DTDO separately. Let $\ell = m - p$, and $F(E) = F(E') = V(G) \cup \{d_0\}$. To make each example coming from some copy of E' unique, we will



(a)

	d_0	u_1	u_2	u_3	u_4	
X_1	1	1	1	0	0	−
Y_2	2	0	0	0	0	+
X_3	3	0	1	1	0	−
Y_4	4	0	0	0	0	+
X_5	5	0	0	1	1	−
Y_6	6	0	0	0	0	+
X_7	7	1	0	0	1	−
Y_8	8	0	0	0	0	+

(b)

	d_0	u_1	u_2	u_3	u_4	
B_1	1	1	1	0	0	−
	2	0	0	0	0	+
	3	0	1	1	0	−
	4	0	0	0	0	+
	5	0	0	1	1	−
	6	0	0	0	0	+
	7	1	0	0	1	−
	8	0	0	0	0	+
B_2	9	1	1	0	0	−
	10	0	0	0	0	+
	11	0	1	1	0	−
	12	0	0	0	0	+
	13	0	0	1	1	−
	14	0	0	0	0	+
	15	1	0	0	1	−
	16	0	0	0	0	+

(c)

Figure 1: Here (a) is the graph G and (b) is the CI E' corresponding to G. In (c), we illustrate an example for constructing E from E' for $\eta = 2$.

set the values of the feature d_0 in increasing order. More formally, let $E'_1, \ldots E'_\eta$ be η copies of the CI E'. Now, for $i \in [\eta]$, if $e' \in E'_i$, then we add an example $e \in E$ such that $e(d_0) = 2m(i-1) + e'(d_0)$ and for every vertex feature $v \in V(G)$, $e(v) = e'(v)$. Let B_i, later referred to as *block* B_i, be the set of examples in E corresponding to the examples in the copy E'_i of E. Finally, we set $s = d = k$ and $t = \ell\eta$. This completes our construction.

Some Preliminaries and Observations. Each vertex feature in $F(E)$ corresponds to a unique vertex in $V(G)$. Depending on the feature $f(v)$ of a test node v, we categorize the test nodes in two categories: *vertex test node* where $f(v)$ is a vertex feature or a *dummy test node* where $f(v) = d_0$. The positive examples in E (E^+) are said to be *dummy examples* and the negative examples in E (E^-) are said to be *edge examples*; each edge example corresponds to a unique edge, and each edge corresponds to exactly η edge examples. We say that an edge example $e \in E$, corresponding to an edge $e' \in E(G)$, is *hit* by a vertex feature u if u is an endpoint of e'. Accordingly, let $H(u)$ denote the set of all edge examples hit by the vertex feature u.

Since we want to learn "small" decision trees, we assume that our DT T has the following properties. For each test node v, $T_E(v)$ are non-uniform; otherwise, we can replace T_v by a leaf node v' to get a "smaller" DT with the same classification power. Similarly, for each test node v with children l and r, $T_E(l) \neq \emptyset$ and $T_E(r) \neq \emptyset$, otherwise we can get a smaller DT with the same classification power. To see this, let $T_E(l) = \emptyset$, then T' obtained by replacing T_v by T_r is the desired DT. We have the following observation.

Observation 4 $(*)$**.** *Let v be a vertex test node in a* DT *T for E, and $f(v) = x$. Moreover, let l and r be the left and right child of v, respectively. Then, $\lambda(v) = 0$, r is a negative leaf, and $T_E(r)$ contains only negative examples. More specifically, $T_E(r)$ contains all of the edge examples in $T_E(v)$ that are hit by the vertex x.*

The following observation follows from Observation 4.

Observation 5. *Let T be a* DT*, for E, with root v_0, and let v_{i-1} be a node of T having v_i as its left child. Moreover, let $v_0, v_1 \ldots, v_{i-1}, v_i$ be the unique (v_0, v_i)-path in T and each node v_j, for $j \in [i-1]$, is a vertex test node. Then, $T_E(v_i) = E \setminus (\bigcup_{j \in [i-1]} H(f(v_j)))$.*

Next, we have the following easy lemma that will be used to prove one side of our reduction.

Lemma 1 ($*$)**.** *If (G, k, p) is a Yes-instance of* PVC*, then there is a* DT *T such that $|T| = dep(T) \leq k$ and $|O(T, E)| \leq \ell\eta$.*

Let B_i, for $i \in \eta$, be a block of E, and T be a DT for E. Then, we say that a (dummy) test node v of T *intersects* B_i if $f(v) = d_0$ and $2(i-1)m+1 \leq \lambda(v) \leq 2im$. Observe that a (dummy) test node can intersect at most one block B_i. Thus, T can have at most $|T|$ many blocks that are intersected by some test, and hence at least $\eta - |T|$ (we assume that $\eta \geq |T|$) blocks of E are not intersected by any test node of T. Let l' be a leaf of a DT T for E. Moreover, we say that a block B_i is *contained* in l' if each positive example e of B_i is in $T_E(l')$, i.e., $T_E(l') \cap E^+ \cap B_i = B_i \cap E^+$. Finally, we have the following observation.

Observation 6 ($*$)**.** *If a block B_i is not intersected by any test node of T, then B_i is contained in some leaf of T. Thus, there are at least $\eta - |T|$ blocks of E that are contained in some leaf of T.*

Let T be a DT for E. Moreover, let B be a block of E such that B is contained in some leaf l of T. Then, let $N(l, B)$ denote the set of negative examples of block B that are in $T_E(l)$, i.e., $N(l, B) = \{B \cap E^- \cap T_E(l)\}$. Next, we have the following lemma.

Lemma 2 ($*$)**.** *If $|N(l, B)| > \ell$ for each block B such that B is contained in some leaf l of T, then $|O(T, E)| \geq (\eta - |T|)(\ell + 1)$.*

The following lemma establishes that if there is some block B of E such that B is contained in some leaf l of T and $|N(l, B)| \leq \ell$, then G has a partial vertex cover of size at most $dep(T)$ that hits at least $p = m - \ell$ edges.

Lemma 3 ($*$)**.** *Let T be a* DT *for E. If there exists some block B of E such that B is contained in some leaf l of T and $|N(l, B)| \leq \ell$, then $(G, dep(T), p)$ is a Yes-instance of* PVC*.*

W[1]-hardness proofs. Now, we present our main lemmas. The following lemma completes our reduction for DTSO.

Lemma 4 ($*$)**.** *Let $\eta = s(\ell + 2)$. Then, (G, k, p) is a Yes-instance of* PVC *iff (E, s, t) is a YES-instance of* DTSO*.*

Next, we have the following lemma, whose proof is similar to the proof of Lemma 4.

Lemma 5 ($*$)**.** *Let $\eta = 2^d(\ell + 2)$. Then, (G, k, p) is a Yes-instance of* PVC *iff (E, d, t) is a YES-instance of* DTDO*.*

Finally, we have the following theorem.

Theorem 1. DTDO *and* DTSO *are W[1]-hard when parameterized by $d+\delta_{max}$ and $s+\delta_{max}$, respectively. Further, they are W[1]-hard parameterized by s and d, respectively, even if $\delta_{max} \leq 3$.*

Proof. Recall that in our construction of CI E from an instance (G, k, t), $\delta_{max} \leq 3$. Moreover, we have that $d = s = k$. Hence, the proof follows from our construction, Lemmas 4 and 5, and Preposition 3. □

FPT Algorithms for DTSO and DTDO

In this section, we complement Theorem 1 by establishing that DTSO and DTDO are FPT when parameterized by $s + \delta_{max} + t$ and $d + \delta_{max} + t$, respectively. To prove this we extend the FPT algorithms for DTD and DTS parameterized by $d + \delta_{max}$ and $s + \delta_{max}$ provided by Eiben et al. (2023). Here, we only present an outline of our algorithm to respect the space constraints. Moreover, we use the notions used by (Eiben et al. 2023) without explicitly defining them. One important component of both these algorithms is that we can enumerate all the minimal support sets of size k in FPT (parameterized by $k + \delta_{max}$) time. We extend this result by establishing that we can enumerate all minimal support sets of size at most k that allow at most t outliers in FPT (parameterized by $k + \delta_{max} + t$) time. Then, the remaining ingredient of our algorithm is the notion of maintaining t outliers in each of their definitions. Moreover, in each check, instead of checking if T is a valid DT for E, we check if $|O(T, E)| \leq t$, which increases the dependency on the running time of their procedures by time FPT in t. Hence, following their algorithm, we can first enumerate all minimal support sets of size at most k ($k = s$ for DTSO and $k = 2^d$ for DTDO) that allow at most t outliers. Then, for each such support set S, we build all possible DT patterns T such that $F(T) = S \cup \{\blacksquare\}$ of size at most s (resp., 2^d) for DTSO (resp., DTDO). Then, for each of these DT patterns, we compute the branching sets $(S, |T|)$. Finally, for each subset S' of $(S, |T|)$ branching set we construct the minimum sized (resp., minimum depth) DT T' with feature set $S' \cup F(T)$. Finally, we have the following theorem.

Theorem 2. DTDO *and* DTSO *are* FPT *parameterized by $d + \delta_{max} + t$ and $s + \delta_{max} + t$, respectively.*

Kernelization Complexity

Observation 1 indicates that it is unlikely for DTS and DTD to attain polynomial compressibility parameterized when δ_{max} is considered to be a part of the input. Since HS admits a kernel of size at most $k^{\mathcal{O}(\Delta)}$, for example, using sunflower lemma (Fomin et al. 2019), it becomes an interesting question to determine the kernelization complexity of DTS and DTD when δ_{max} is considered to be a constant (as $\delta_{max} = \Delta$ in the reduction).

Trivial Kernelization W.R.t. $D_{max} + |F|$.

Here, we establish that DTS and DTD admit a trivial polynomial kernel parameterized by $D_{max} + |F|$ when δ_{max} is a fixed constant. The proof follows from simple arguments that upper bounds the number of examples in any classification instance by $(|F|D_{max})^{2\delta_{max}}$.

Lemma 6 ($*$)**.** *A classification instance can contain at most $2\binom{|F|}{\delta_{max}}(D_{max})^{\delta_{max}}$ examples.*

Thus, Lemma 6 implies the following theorem.

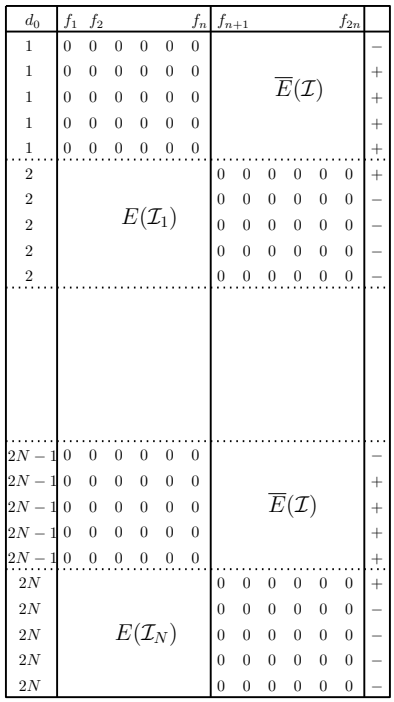


Figure 2: The construction of E from $\overline{E}(\mathcal{I})$ and $E(\mathcal{I}_j)$, for $j \in [N]$.

Theorem 3. DTS *and* DTD *parameterized by* $D_{max} + |F|$ *admit a trivial polynomial kernel when* δ_{max} *is a constant.*

Incompressibility for DTD W.R.t. $d + |F|$

Next, we establish that even when $\delta_{max} = 3$, DTD parameterized by $d+|F|$ does not admit a polynomial compression, unless NP $\subseteq$ coNP/poly. To this end, we provide a non-trivial AND-composition for DTD parameterized by $|F|+d$ such that $\delta_{max} = 3$. We will use the following construction.

Construction. An instance (G, k) of VERTEX COVER can be modeled as an instance $(\mathcal{F}, U, k)$ of HS with arity at most two such that $U = V(G)$ and $\mathcal{F} = E(G)$. Let $I_1 = (G_1, k), \ldots, I_N = (G_N, k)$ be N instances of VERTEX COVER such that for $i, j \in N$, $|V(G_i)| = |V(G_j)| = n$. Assume WLOG that $V(G_i) = V(G_j)$. Moreover, assume that $N = 2^\ell$, for some $\ell \in \mathbb{N}$, as otherwise, we can add some dummy instances to make N a power of 2. Furthermore, let $\mathcal{I}_1, \ldots, \mathcal{I}_N$ be the corresponding instances of HS, each having arity at most 2, and let $E(\mathcal{I}_j)$ be the CI corresponding to the HS instance $\mathcal{I}_j$, for $j \in [N]$. Moreover, let $F(E(\mathcal{I}_j)) = \{f_1, \ldots, f_n\}$.

Now, let G be a fixed graph such that $|V(G)| = n$ and G has a minimum vertex cover of size k which is known to us. One such graph (assuming $n > k$) can be a split graph $G = (C, I)$ such that $|C| = k$ and each vertex in I is connected to each vertex in C. Let $\mathcal{I}$ be the corresponding HS instance of G. Consider the CI $\overline{E}(\mathcal{I})$, and let $F(\overline{E}(\mathcal{I})) = \{f_{n+1}, \ldots, f_{2n}\}$.

Finally, we create the CI E as follows. See Figure 2 for an illustration. Let $F(E) = \{f_1, \ldots f_{2n}\} \cup \{d_0\}$, i.e., $F(E)$ contains each feature in $F(\overline{E}(\mathcal{I}))$, each feature in $F(E(\mathcal{I}_j))$ for $j \in [N]$, and a *dummy feature* d_0. Now, E will contain $2N$ *blocks* of examples $B_1, \ldots, B_{2N}$. For $i \in [2N]$, the examples in B_i will depend on whether i is odd or even as follows.

1. i is odd. In this case, for each positive/negative example e' in $\overline{E}(\mathcal{I})$, we add a positive/negative example e to B_i as follows. First, set $e(d_0) = i$. Second, for $j \in [n]$, set $e(f_j) = 0$. Finally, for $n + 1 \leq j \leq 2n$, set $e(f_j) = e'(f_j)$. Observe that when restricted to the features in $\{f_{n+1}, \ldots, f_{2n}\}$, B_i is identical to $\overline{E}(\mathcal{I})$, and for every other feature of E, the examples in B_i have identical values. Hence, the examples in B_i can be classified using a DT of size (resp., depth) at most k, and cannot be classified using a DT of size (resp., depth) less than k.
2. i is even. Let $p = \frac{i}{2}$. In this case, for each positive/negative example e' of $E(\mathcal{I}_p)$, we add a positive/negative example e to B_i in the following manner. First, set $e(d_0) = i$. Second, for $j \in [n]$, set $e(f_j) = e'(f_j)$. Finally, for $n + 1 \leq j \leq 2n$, set $e(f_j) = 0$. Similarly to the previous case, observe that when restricted to the features in $\{f_1, \ldots, f_n\}$, B_i is identical to $E(\mathcal{I}_p)$, and for every other feature, the examples in B_i have identical values. Hence, examples in B_i can be classified using a DT of depth (resp., size) ℓ iff $E(\mathcal{I}_p)$ can be classified using a DT of depth (resp., size) k.

Finally, let $p = \log_2 N+1$ and $d = p+k = \log_2 N+k+1$. This completes our construction. Observe that in each block B_i, for $i \in [2N]$, there is exactly one example e such that for every vertex feature f_j, $j \in [2n]$, $e(f_j) = 0$. Moreover, e is a negative example if i is odd, and a positive example if i is even. We refer to such examples as *dummy examples*. All other examples are said to be *edge examples*. We have the following observations about our CI E.

Observation 7. *Let each* I_j*, for* $j \in [N]$*, be a Yes-instance of* VERTEX COVER*. Then, the examples of block* B_i*, for* $i \in [2N]$*, can be classified using a* DT T_i *of depth at most* k*. Moreover, when* i *is odd, the examples of* B_i *cannot be classified using a* DT *of depth less than* k*.*

Let T be a DT for E that classifies E. Then, a test node v of T is said to be a *dummy test node* if $f(v) = \{d_0\}$ and is said to be a *vertex test node*, otherwise. Moreover, similar to the reduction in W[1]-hardness proofs (Section), we assume that for each test node v with children l and r, $T_E(v)$ is non-uniform and $T_E(l) \neq \emptyset$ and $T_E(r) \neq \emptyset$. We discuss the effect of these test nodes below.

Dummy test node. Let v be a dummy test node of T, and let l and r be the left and right child of v, respectively. Then, $f(v) = d_0$ and $\lambda(v) \in [2N]$. Observe that for each example $e \in T_E(v)$, if $e(d_0) \leq \lambda(v)$, then $e \in T_E(l)$, else, $e \in T_E(r)$. Moreover, since the value of d_0 is the same for each example of a block B, all examples of B that are in $T_E(v)$ will end up in either $T_E(l)$ or $T_E(r)$, i.e. either $T_E(v) \cap B = T_E(l) \cap B$ or $T_E(v) \cap B = T_E(r) \cap B$. Hence, we have the following.

Observation 8. *Let* v_0 *be the root of a* DT T *for* E*. Moreover, let* v *be a dummy test node of* T *with* l *and* r *as the*

left and right child of v, respectively. Furthermore, let every node on the unique (v_0, v)-path be a dummy test node. Then, if $T_E(l)$ (resp., $T_E(r)$) contains some example of a block B, then $T_E(l)$ (resp., $T_E(r)$) contains every example of the block B.

Vertex test node. Let v be a vertex test node of T with l and r as the left and the right child of v, respectively. Then, $f(v) \in \{f_1, \ldots, f_{2n}\}$ and $\lambda(v) = 0$ (as otherwise, either $T_E(l) = \emptyset$ or $T_E(r) = \emptyset$). Moreover, if $f(v) \in \{f_1, \ldots, f_n\}$, then r is a negative leaf; else, r is a positive leaf. Hence, we have the following.

Observation 9. *Let v be a vertex test node in a* DT *T for E. Moreover, let l and r be the left and right child of v, respectively. Then, $\lambda(v) = 0$ and r is a negative (resp., positive) leaf if $f(v) \in \{f_1, \ldots, f_n\}$ (resp., $f(v) \in \{f_{n+1,\ldots,f_{2n}}\}$).*

Now, we have the following lemma, which proves one side of our reduction.

Lemma 7 ($*$)**.** *If each I_j, for $j \in [N]$, is a Yes-instance of* VERTEX COVER*, then (E, d) is a Yes-instance of* DTD.

In the following, we prove that if (E, d) is a Yes-instance of DTD, then I_j, for $j \in [N]$, is a Yes-instance of VERTEX COVER. Let v and v' be test nodes of DTs T and T', respectively (possibly, $T = T'$ and $v = v'$). Then, we say that $v \sim v'$ if $f(v) = f(v')$ and $\lambda(v) = \lambda(v')$. We have the following straightforward observation.

Observation 10. *Let u and v (resp., u' and v') be the nodes of a* DT *T (resp., T') such that*

1. *u (resp., u') is an ancestor of v (resp., v') in T (resp., T'),*
2. *$T'_E(u') \subseteq T_E(u)$, and*
3. *for each vertex x on the unique (u, v)-path in T, there is a vertex y on the unique (u', v')-path in T' such that $x \sim y$.*

Then, $T'_E(v') \subseteq T_E(v)$.

Next, we have the following crucial lemma.

Lemma 8 ($*$)**.** *Let T' be a* DT *that classifies E. Then, there is a* DT *T that classifies E, $dep(T) \leq dep(T')$, and none of the dummy test nodes in T' has a vertex test node as an ancestor.*

Next, we have the following lemma.

Lemma 9. *If (E, d) is a Yes-instance of* DTD*, then I_j, for $j \in [N]$, is a Yes-instance of* VERTEX COVER.

Proof. Let T be a DT with root v_0 such that $dep(T) \leq d$ and T classifies E. Due to Lemma 8, we can assume that none of the dummy test nodes has a vertex test node as its ancestor. Let T' be a subtree of T such that each node in T' except v_0 has a dummy test node as its parent. We note that each leaf of T' can be either a vertex test node in T or a leaf node in T. Also, note that for each node v of T', $T_E(v) = T'_E(v)$. We will now discuss the properties of T'.

Claim 1 ($*$)**.** *For each leaf l of T', $T'_E(l) = B_i$, for some $i \in [2N]$. Moreover, for each $i \in [2N]$, there is some leaf l of T' such that $T'_E(l) = B_i$.*

Next, we claim that T' is a complete binary tree of depth $p = \log_2(N) + 1$.

Claim 2 ($*$)**.** *T' is a complete binary tree and $dep(T') = p$.*

Finally, we have the following claim.

Claim 3 ($*$)**.** *Each $I_j = (G_j, k)$, $j \in [N]$, is a Yes-instance of* VERTEX COVER.

Therefore, due to Claim 3, we have that each $I_j = (G_j, k)$, $j \in [N]$, is a Yes-instance of VERTEX COVER. This completes our proof. □

Finally, our construction and Lemmas 7 and 9 imply that DTD is AND-compositional. Moreover, since $\delta_{max} \leq 3$ for E, $|F| = 2|V(G_i)| + 1$, and $d = \log_2 N + k$, we have the following theorem as a consequence.

Theorem 5. DTD *parameterized by $d + |F|$ is AND-compositional. Hence,* DTD *parameterized by $d + |F|$ does not admit a polynomial compression even when $\delta_{max} \leq 3$, unless NP $\subseteq$ coNP/poly.*

Conclusion

We considered a generalization of DECISION TREE LEARNING where the constructed DT is allowed to disagree with the input CI on a "*few*" examples. We considered DTDO and DTSO which are generalizations of DTD and DTS, respectively, from both parameterized and kernelization perspectives. We remark that our results can be generalized to a setting where the "outliers" are only permitted to be coming from some "noisy" part of the data by simply making $t + 1$ copies of each example not eligible to be an outlier (if we allow E to be a multiset), and otherwise, if we need E to be a set, the FPT-algorithm can be adjusted to restrict outliers to be only from the noisy part of the data. We established that DTD parameterized by $d + |F|$ is unlikely to admit a polynomial kernel even when δ_{max} is a fixed constant. To this end, we use a DT of depth $\log N + 1$ to filter out blocks of examples; each leaf of this "filter tree" gets a unique block. The size of this filter tree is $2N$, which is not allowed for a parameter in the AND-composition (the parameter cannot have linear dependency on the number of instances). Hence, one specific interesting open question is whether DTS parameterized by $s + |F|$ admits a polynomial kernel when δ_{max} is a fixed constant?

Komusiewicz et al. (Komusiewicz et al. 2023) recently considered the parameterized complexity of two generalizations of DTS where the task is to design ℓ DTs such that for each example, the majority of the trees agree with it. For $\ell = 1$ both these generalizations model DTS. They provided FPT algorithms parameterized by $s + \delta_{max} + \ell + D_{max}$ and asked as an open question if these problems admit a polynomial kernel parameterized by the same parameters. Notice that Observation 1 answers this question negatively since the incompressibility is implied for these problems even by fixing $\ell = 1$ and $D_{max} = 2$. Moreover, note that since the trivial kernel in Theorem 3 is obtained by establishing an upper bound on the size of the CI, the kernelization result applies to their generalizations as well.

Acknowledgements

Both authors of this work were supported by the European Research Council (ERC) grant titled PARAPATH.

References

Avellaneda, F. 2020. Efficient inference of optimal decision trees. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 3195–3202.

Bäckström, C.; Chen, Y.; Jonsson, P.; Ordyniak, S.; and Szeider, S. 2012. The complexity of planning revisited—a parameterized analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, 1735–1741.

Bessiere, C.; Hebrard, E.; Hnich, B.; Kiziltan, Z.; Quimper, C. G.; and Walsh, T. 2008. The parameterized complexity of global constraints. In *AAAI Conference on Artificial Intelligence*, 235–240. AAAI Press.

Bessiere, C.; Hebrard, E.; and O'Sullivan, B. 2009. Minimising decision tree size as combinatorial optimisation. In *International Conference on Principles and Practice of Constraint Programming*, 173–187. Springer.

Bredereck, R.; Chen, J.; Niedermeier, R.; and Walsh, T. 2017. Parliamentary voting procedures: Agenda control, manipulation, and uncertainty. *Journal of Artificial Intelligence Research*, 59: 133–173.

Cygan, M.; Fomin, F. V.; Kowalik, L.; Lokshtanov, D.; Marx, D.; Pilipczuk, M.; Pilipczuk, M.; and Saurabh, S. 2015. *Parameterized Algorithms*. Springer Publishing Company, Incorporated, 1st edition. ISBN 3319212745.

Darwiche, A.; and Hirth, A. 2020. On the Reasons Behind Decisions. In *ECAI 2020*, 712–720. IOS Press.

Diestel, R. 2006. *Graph Theory*. Springer.

Dom, M.; Lokshtanov, D.; and Saurabh, S. 2009. Incompressibility through Colors and IDs. In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming*, ICALP '09, 378–389. Berlin, Heidelberg: Springer-Verlag. ISBN 9783642029264.

Doshi-Velez, F.; and Kim, B. 2017. A roadmap for a rigorous science of interpretability. *arXiv preprint arXiv:1702.08608*, 2(1).

Eiben, E.; Ordyniak, S.; Paesani, G.; and Szeider, S. 2023. Learning small decision trees with large domain. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-23)*, volume toAppear.

Fellows, M. R. 2006. The Lost Continent of Polynomial Time: Preprocessing and Kernelization. IWPEC'06, 276–277. Berlin, Heidelberg: Springer-Verlag. ISBN 3540390987.

Fomin, F. V.; Lokshtanov, D.; Saurabh, S.; and Zehavi, M. 2019. *Kernelization: Theory of Parameterized Preprocessing*. Cambridge University Press.

Ganian, R.; Kanj, I.; Ordyniak, S.; and Szeider, S. 2018. Parameterized algorithms for the matrix completion problem. In *International Conference on Machine Learning*, 1656–1665. PMLR.

Gao, Y. 2009. Data reductions, fixed parameter tractability, and random weighted d-CNF satisfiability. *Artificial Intelligence*, 173(14): 1343–1366.

Gaspers, S.; Misra, N.; Ordyniak, S.; Szeider, S.; and Živnỳ, S. 2017. Backdoors into heterogeneous classes of SAT and CSP. *Journal of Computer and System Sciences*, 85: 38–56.

Goodman, B.; and Flaxman, S. 2017. European Union regulations on algorithmic decision-making and a "right to explanation". *AI magazine*, 38(3): 50–57.

Guo, J.; and Niedermeier, R. 2007. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(1): 31–45.

Guo, J.; Niedermeier, R.; and Wernicke, S. 2007. Parameterized complexity of vertex cover variants. *Theory of Computing Systems*, 41: 501–520.

Ibaraki, T.; Crama, Y.; and Hammer, P. L. 2011. Partially defined Boolean functions. *Boolean Functions-Theory, Algorithms, and Applications. Eds: Crama Y., Hammer PL Cambridge Press University*.

Kobourov, S. G.; Löffler, M.; Montecchiani, F.; Pilipczuk, M.; Rutter, I.; Seidel, R.; Sorge, M.; and Wulms, J. 2023. The influence of dimensions on the complexity of computing decision trees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 8343–8350.

Komusiewicz, C.; Kunz, P.; Sommer, F.; and Sorge, M. 2023. On Computing Optimal Tree Ensembles. In *Proceedings of the International Conference on Machine Learning (ICML-23)*, volume toAppear.

Larose, D. T.; and Larose, C. D. 2014. *Discovering knowledge in data: an introduction to data mining*, volume 4. John Wiley & Sons.

Laurent, H.; and Rivest, R. L. 1976. Constructing optimal binary decision trees is NP-complete. *Information processing letters*, 5(1): 15–17.

Lipton, Z. C. 2018. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3): 31–57.

Monroe, D. 2018. AI, explain yourself. *Communications of the ACM*, 61(11): 11–13.

Murthy, S. K. 1998. Automatic construction of decision trees from data: A multi-disciplinary survey. *Data mining and knowledge discovery*, 2: 345–389.

Narodytska, N.; Ignatiev, A.; Pereira, F.; Marques-Silva, J.; and Ras, I. 2018. Learning Optimal Decision Trees with SAT. In *Ijcai*, 1362–1368.

Niedermeier, R.; and Rossmanith, P. 2000. A general method to speed up fixed-parameter-tractable algorithms. *Information Processing Letters*, 73(3-4): 125–129.

Ordyniak, S.; and Szeider, S. 2021. Parameterized complexity of small decision tree learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 6454–6462.

Quinlan, J. R. 1986. Induction of decision trees. *Machine learning*, 1: 81–106.

Schidler, A.; and Szeider, S. 2021. SAT-based decision tree learning for large data sets. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 3904–3912.

Steinberg, D.; and Colla, P. 2009. CART: classification and regression trees. *The top ten algorithms in data mining*, 9: 179.

Weihe, K. 1998. Covering trains by stations or the power of data reduction. *Proceedings of Algorithms and Experiments, ALEX*, 1–8.

Wu, X.; Kumar, V.; Ross Quinlan, J.; Ghosh, J.; Yang, Q.; Motoda, H.; McLachlan, G. J.; Ng, A.; Liu, B.; Yu, P. S.; et al. 2008. Top 10 algorithms in data mining. *Knowledge and information systems*, 14: 1–37.