A query-optimal algorithm for finding counterfactuals

Guy Blanc [*1] **Caleb Koch** [*1] **Jane Lange** [*2] **Li-Yang Tan** [*1]

Abstract

We design an algorithm for finding counterfactuals with strong theoretical guarantees on its performance. For any monotone model $f : X^d \to \{0,1\}$ and instance $x^\star$, our algorithm makes

$$S(f)^{O(\Delta_f(x^\star))} \cdot \log d$$

queries to f and returns an *optimal* counterfactual for $x^\star$: a nearest instance x' to $x^\star$ for which $f(x') \neq f(x^\star)$. Here $S(f)$ is the sensitivity of f, a discrete analogue of the Lipschitz constant, and $\Delta_f(x^\star)$ is the distance from $x^\star$ to its nearest counterfactuals. The previous best known query complexity was $d^{O(\Delta_f(x^\star))}$, achievable by brute-force local search. We further prove a lower bound of $S(f)^{\Omega(\Delta_f(x^\star))} + \Omega(\log d)$ on the query complexity of any algorithm, thereby showing that the guarantees of our algorithm are essentially optimal.

1. Introduction

Counterfactual reasoning is at the very heart of causal inference (Pearl, 2009a;b; Morgan & Winship, 2015). Counterfactuals are answers to "what would have happened if" questions: if a person has been denied a loan, would their application have been approved if their annual income were \$20k higher? In explainable ML, there is a growing interest in the use of *counterfactual explanations* (Wachter et al., 2017) to understand the predictions of black box models: given a model f and an instance $x^\star$, we would like to efficiently determine how a few features of $x^\star$ can be changed to obtain a similar instance x' for which $f(x') \neq f(x^\star)$.

It is natural to seek *sparse* counterfactuals, meaning that x' differs from $x^\star$ in as few features as possible. Ideally, we would like counterfactuals that are *optimally* sparse:

Definition 1.1 (Sparsity and optimality)**.** For a model $f : X^d \to \{0,1\}$ and two instances $x^\star, x'$ such that $f(x^\star) \neq f(x')$, the *sparsity* of x' as a counterfactual for $x^\star$ is the number of features on which they differ: the quantity $|\Delta(x^\star, x')|$, where

$$\Delta(x^\star, x') \coloneqq \{i \in [d] \colon x^\star_i \neq x'_i\}.$$

The *counterfactual complexity* of $x^\star$ with respect to f is the quantity:

$$\Delta_f(x^\star) \coloneqq \min_{x' \in X^d} \{|\Delta(x^\star, x')| \colon f(x^\star) \neq f(x')\},$$

and we say that x' is an *optimal* counterfactual for $x^\star$ if $|\Delta(x^\star, x')| = \Delta_f(x^\star)$.

Another desideratum that has received significant attention, motivated by the need for actionable recourse (Ustun et al., 2019), is that of *diversity* in counterfactual explanations (Wachter et al., 2017; Russell, 2019; Mothilal et al., 2020; Karimi et al., 2020): a wide range of counterfactuals instead of just a single one.

1.1. Our contributions

Our first result is an efficient algorithm for finding *all optimal* counterfactuals for any monotone f in the setting of binary features:

Theorem 1.2. *Given queries to a monotone model $f : \{0,1\}^d \to \{0,1\}$ with sensitivity $S(f)$ and an instance $x^\star$, our algorithm makes*

$$S(f)^{O(\Delta_f(x^\star))} \cdot \log d$$

queries to f and w.h.p.[1] *returns all optimal counterfactuals for $x^\star$.*

We will define and discuss sensitivity in the body of the paper (see Section 1.4), mentioning for now that it is a well-studied discrete analogue of the Lipschitz constant, and this quantity is often much smaller than d.

[*]Equal contribution [1]Department of Computer Science, Stanford University [2]Department of Computer Science, Massachusetts Institute of Technology. Correspondence to: Guy Blanc <gblanc@stanford.edu>, Caleb Koch <ckoch@stanford.edu>, Jane Lange <jlange@mit.edu>, Li-Yang Tan <liyang@cs.stanford.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

[1]Throughout we write "with high probability" or w.h.p. to indicate probability at least $1 - 1/\mathrm{poly}(d)$.

Next, we consider the setting of general feature spaces. In this setting it is infeasible to return *all* optimal counterfactuals due to the sheer number of them. We are nevertheless able to efficiently return the collection of all *subsets of features* induced by these optimal counterfactuals:

Theorem 1.3. *Given queries to a monotone model $f : X^d \to \{0,1\}$ with sensitivity $S(f)$ and an instance $x^\star$, our algorithm makes*

$$S(f)^{O(\Delta_f(x^\star))} \cdot \log d$$

queries to f and w.h.p. returns the collection $\mathcal{C}_f(x^\star) = \{\Delta(x^\star, x') \colon x' \text{ is an optimal counterfactual for } x^\star\}$.

Theorems 1.2 and 1.3 give the first algorithms with query complexity that evades the curse of dimensionality, and indeed, strongly so. We contrast our query complexity to that of ball search, a simple and natural algorithm for finding counterfactuals: first query f on $x^\star$ and all instances that differ from $x^\star$ by a single feature. (By the monotonicity of f, for any feature, it suffices to query f on the instance that differs maximally from $x^\star$ on that feature.) Next, query f on the instances that differ by two features, and so on, until a counterfactual is found. This algorithm has query complexity $d^{O(\Delta_f(x^\star))}$, an exponentially worse dependence on d than ours. Prior to our work, this was the best known query complexity even just to return a *single* optimal counterfactual.

Lower bounds. We complement Theorems 1.2 and 1.3 with a couple of lower bounds. All our lower bounds hold even in the setting of binary features ($X = \{0,1\}$), and against randomized algorithms that are only required to successfully return an optimal counterfactual with a small probability (say 0.01). We first show that the query complexity of our algorithm is essentially optimal:

Theorem 1.4 (Optimality of Theorems 1.2 and 1.3, see Theorem C.1 for the formal version)**.** *For any algorithm $\mathcal{A}$ and $S \in \mathbb{N}$, there is a monotone model $f : \{0,1\}^d \to \{0,1\}$ with $S(f) = S$ and an instance $x^\star$ such that $\mathcal{A}$ must make*

$$S^{\Omega(\Delta_f(x^\star))} + \Omega(\log d)$$

many queries to f, even just to find a single optimal counterfactual for $x^\star$.

We also establish the inherent limitations of *local search* algorithms, a broad and natural class of algorithms for finding counterfactuals. A local search algorithm is any algorithm whose first query is $x^\star$, and whose every subsequent query differs from a previous one by exactly one feature. For example, ball search is a local search algorithm. We show that no local search algorithm can achieve the query complexity that our algorithm does, even for low-sensitivity monotone functions:

Theorem 1.5 (Lower bound for local search algorithms, see Theorem C.5 for the formal version)**.** *For any local search algorithm $\mathcal{A}$ and $d \in \mathbb{N}$, there is a monotone model $f : \{0,1\}^d \to \{0,1\}$ with $S(f) = 1$ and an instance $x^\star$ with $\Delta_f(x^\star) = 1$, such that $\mathcal{A}$ must make $\Omega(d)$ many queries to f to find a single optimal counterfactual for $x^\star$.*

1.2. Overview of our algorithm and techniques

Theorems 1.2 and 1.3 circumvent the lower bound of Theorem 1.5 via a novel approach that is *not* based on local search. The crux of our approach is a new algorithm for understanding monotone black box models that we believe will see further utility beyond counterfactual explanations: using only queries to a monotone black box model f, this algorithm allows us to navigate a decision tree representation T of f without actually building T in full. Crucially, since we strive to handle arbitrarily complex models f, this tree T may be too large to build efficiently in its entirety.

With this algorithm, given any instance $x^\star$, we will build only the "necessary part" of T to find all optimal counterfactuals for $x^\star$. We show that just a tiny portion of T suffices for this purpose: it suffices to build only the root-to-leaf path ρ in T that $x^\star$ follows and the paths that are "close to" ρ in T in a sense that we will make precise (see the beginning of Section 3 for more details on this notion of " path distance"). Writing h to denote the depth of T, we prove that we only have to construct $h^{O(\Delta_f(x^\star))}$ many nodes in T, which is only an exponentially small part of T. Next, by bringing together classical (Nisan, 1989) and recently-develop results (Blanc et al., 2022) about the structure of monotone functions, we are able to bound the depth of T and in terms of f's sensitivity.

To summarize, our overall approach allows us to enjoy the benefits of having access to a highly interpretable representation of a black box model f—a decision tree representation—without paying the price associated with T being intractably large for most complex models of interest.

1.3. Related work

Our work is theoretical in nature: we give an algorithm with strong bounds on its query complexity and all the counterfactuals that it returns are guaranteed to be optimal. We view our main contribution as proving the curse of dimensionality can be strongly evaded for a broad and natural class of models—all monotone models.

There is a substantial body of empirical work on counterfactual explanations. The algorithms in these works either do not guarantee optimal counterfactuals or do not come with a formal analysis of their query complexity. Many of these works rely on classic optimization tools: the algorithms of (Wachter et al., 2017; Mothilal et al., 2020) are based on

gradient descent and hence are restricted to differentiable models, whereas the algorithms of (Russell, 2019; Ustun et al., 2019) use mixed integer programming and are limited to linear models. (Karimi et al., 2020) encode the problem of finding optimal counterfactuals as a satisfiability problem, which they then solve using SMT solvers.

The importance of diversity in counterfactual explanations has been highlighted in several works (Wachter et al., 2017; Russell, 2019; Mothilal et al., 2020; Karimi et al., 2020). Significant motivation comes from the need for actionable recourse (Ustun et al., 2019): a large set of counterfactuals is more likely to contain one that is actionable (e.g. in the context of a loan denial, an applicant can plausibly work towards earning a higher income but cannot change their history of defaults).

Regarding our techniques, they are in the spirit of a recent line of theoretical work on *implicit* learning algorithms: algorithms that are able to access their hypotheses efficiently without constructing them in full (Kong & Valiant, 2018; Blum & Hu, 2018; Kong et al., 2020; Backurs et al., 2020; Blanc et al., 2021). In particular, Blanc, Lange, and Tan (Blanc et al., 2021) show how, given queries to a model f, one can efficiently navigate an implicit decision tree hypothesis T that is ε-close to f under the uniform distribution. The presence of errors in the hypothesis, and the fact that these errors are measured with respect to the uniform distribution, are significant limitations of their result. Our work shows how these limitations can be overcome, in a strong sense, in the case of monotone models f: our implicit decision tree hypothesis T computes f *exactly*, which is crucial for our application to finding counterfactuals. Another limitation of (Blanc et al., 2021)'s analysis is that it only applies to the setting of binary features; we do not need this assumption.

Finally, we mention that counterfactual explanations are part of a broader landscape of local explanations (Strumbelj & Kononenko, 2010; Baehrens et al., 2010; Simonyan et al., 2014; Ribeiro et al., 2016; Koh & Liang, 2017; Lundberg & Lee, 2017; Ribeiro et al., 2018), which seek to explain a model's prediction for specific inputs. Global explanations, on the other hand, approximate the entire model with a simple and interpretable one (Craven & Shavlik, 1995; Breiman & Shang, 1996; Van Assche & Blockeel, 2007; Zhou & Hooker, 2016; Vandewiele et al., 2017; Bastani et al., 2017; Vidal & Schiffer, 2020; Lakkaraju et al., 2019; 2020).

1.4. Preliminaries

We rely on a few standard notions from the study of Boolean functions.

Definition 1.6 (Sensitivity)**.** For a function $f : X^d \to \{0,1\}$ and an instance $x \in X^d$, the *sensitivity of f at x* is the quantity

$$S_f(x) = |\{i \in [d] \colon \exists\, a \in X \text{ s.t. } f(x) \neq f(x_{i \leftarrow a})\}|,$$

where $i \leftarrow a$ denotes x with its i-th feature set to a. That is, $S_f(x)$ is the number of features i of x for which there is a way of changing x_i that flips f's value on x.

The *sensitivity of f* is the quantity $S(f) = \max_{x \in X^d} \{S_f(x)\}$.

The sensitivity of a function can be viewed as a discrete analogue of the Lipschitz constant, with low-sensitivity discrete functions being considered smooth. To see this analogy, we observe that for all $x \in X^d$ and $\delta \in (0,1)$,

$$\mathop{\mathbb{E}}_{\substack{\boldsymbol{y} \in X^d \\ \Delta(x,\boldsymbol{y}) = \delta d}} \left[|f(x) - f(\boldsymbol{y})|\right] \leq \delta \cdot S(f).$$

See (Gopalan et al., 2016) for more on this perspective. Introduced by Cook, Dwork, and Reischuk (Cook et al., 1986), the sensitivity of discrete functions is the subject of intensive study in theoretical computer science; it is, for example, the key notion in the recent breakthrough *Sensitivity Theorem* of Huang (Huang, 2019) (see also (Knuth, 2019)), resolving a longstanding conjecture of Nisan and Szegedy (Nisan & Szegedy, 1994).

Monotonicity. For a model $f : X^d \to \{0,1\}$, we assume an ordering $\leq$ on the elements of X which we lift coordinatewise to a partial ordering on X^d. That is, for $x, y \in X^d$, $x \leq y$ if and only if $x_i \leq y_i$ for all $i \in [d]$. We say f is *monotone* if it is monotone with respect to such an ordering. We'll also assume that X has a maximum and minimum element.

Structure of the rest of this paper. We first prove Theorem 1.2, which focuses on functions with Boolean features. To do so, we will recall the notion of an implicit decision tree from (Blanc et al., 2021) in Section 2. Then, in Section 3, we prove Theorem 1.2 assuming a sufficiently good implicit decision tree exists. In Section 4, we complete the proof of Theorem 1.2 by showing how to build that implicit decision tree.

Given Theorem 1.2, we show how to extend our algorithm to the setting of arbitrary features by reducing to the Boolean setting and prove Theorem 1.3 in Section 5. Finally, we prove our lower bounds, Theorems 1.4 and 1.5, in Appendix C.

2. Implicit decision trees for monotone models

Our algorithm for finding counterfactuals will rely on having access to a decision tree T which exactly represents the model $f : \{0,1\}^d \to \{0,1\}$. At a high level, for an instance $x^\star$, the algorithm follows a unique root-to-leaf path ρ in T

searches all root-to-leaf paths in T that are "close to" ρ in a sense that make precise in the next section. In general, the size of T will be exponentially larger than the search space of our counterfactual finding algorithm, and hence it is advantageous to avoid computing the entire tree. Instead, we maintain an *implicit* copy of the tree T. This copy allows us to compute only the parts of the tree that we need to find optimal counterfactuals for $x^\star$.

Restrictions. An implicit decision tree T, a notion introduced in (Blanc et al., 2021), is an algorithm which given query access to f can navigate T without building it in full. A node in T is specified by a *restriction* which is a partial function $\rho : [d] \rightharpoonup \{0,1\}$ indicating which (if any) features are fixed to specific values. These features and their values correspond to those queried along a path in the tree. We write $\mathrm{Dom}(\rho) \subseteq [d]$ to denote the domain of the restriction. The size of a restriction $|\rho|$ is $|\mathrm{Dom}(\rho)|$, the number of values on which it is defined. We write $f_\rho : \{0,1\}^{d-|\mathrm{Dom}(\rho)|} \to \{0,1\}$ to denote the restriction of f to ρ. That is, $f_\rho(x)$ is f evaluated on the instance $x' \in \{0,1\}^d$ which is formed by inserting features specified by ρ into the instance x. For example, $\rho = \{1 \mapsto 1, 4 \mapsto 1\}$ is the restriction where the 1st and 4th features are fixed to 1. If $f : \{0,1\}^4 \to \{0,1\}$, then $f_\rho : \{0,1\}^2 \to \{0,1\}$ and e.g. $f_\rho(00) = f(1001)$ under this restriction.

Definition 2.1 (Implicit decision tree)**.** An implicit decision tree (IDT) T for $f : \{0,1\}^d \to \{0,1\}$ is an algorithm which has query access to f and supports the following operations. For a restriction $\rho : [d] \rightharpoonup \{0,1\}$ corresponding to features queried along a root-to-ρ path in T and auxiliary information $A \subseteq [d]$:

1. $\textsc{IsLeaf}_f(\rho, A)$ indicates whether ρ is a leaf in T.
2. $\textsc{Query}_f(\rho, A)$, for a non-leaf ρ, returns $i \in [d]$ corresponding to the index queried at ρ in T and $A' \subseteq [d]$, updated auxiliary information.

If each operation uses at most q queries to f with high probability then the algorithm is a q-query implicit decision tree for f.[2]

The role of auxiliary information. Our algorithm for finding counterfactuals updates the auxiliary information after each query and we assume that the auxiliary information being passed to the algorithm originates from the previous query (the auxiliary information at the root can be arbitrary). Internally, the auxiliary information will be a subset of features and will allow us to precompute parts of the tree instead of node by node. This way, most calls to $\textsc{Query}(\rho, A)$ will simply return the appropriate feature index from A until all such features in A are exhausted after which a new A is computed from scratch. The example below illustrates the reuse of auxiliary information between IDT operations. For a more detailed illustration of an IDT and the precomputation of parts of the tree using auxiliary information see Figure 4.

Example. Figure 3 in Appendix D illustrates how the IDT operations can be used to walk down a decision tree T representing a function f. In this case, we make three calls to $\textsc{Query}_f(\cdot,\cdot)$ corresponding to the depth of this particular root-to-leaf path. In general, if a q-query IDT has depth at most k, then the root-to-leaf path corresponding to an instance $x^\star$ can be constructed using $O(k)$ calls to the IDT operations and therefore $O(kq)$ queries to f.

Our key technical lemma. For our intended application, it will be important that the IDT *exactly* represents f. The work of (Blanc et al., 2021) design a relaxed variant of IDTs that only *approximates* models f with respect to the uniform distribution over instances. Both the presence of errors and the uniform-distribution assumption are not ideal, and can be seen to be inherent to the proof technique of (Blanc et al., 2021). In this work we use recently developed structural results for monotone models (Blanc et al., 2022) to overcome these limitations. Our key technical lemma is the following.

Lemma 2.2 (Exact IDTs for monotone models)**.** *Let $f : \{0,1\}^d \to \{0,1\}$ be monotone. Then there is an $S(f)^{O(1)} \cdot \log d$-query IDT for f with depth at most $S(f)^{O(1)}$.*

In the next section we prove Theorem 1.2 assuming Lemma 2.2. We then prove Lemma 2.2 in Section 4.

3. Using IDTs to find optimal counterfactuals: Proof of Theorem 1.2

Suppose we may access an implicit decision tree T for f as described in Definition 2.1. We present an algorithm for finding counterfactuals using these operations and analyze its query complexity in terms of the depth h of T and the query complexity of the IDT—Lemma 2.2 provides strong bounds on both quantities in terms of the sensitivity of f.

Intuition. Given an instance $x^\star$, it follows a unique root-to-leaf path ρ in T. We begin with a simple observation: if there is a counterfactual x' of distance 1 from $x^\star$, it must belong to the set I_1 of all instances that differ from $x^\star$ on a single feature *queried along* ρ. The size of I_1 is the length of ρ, which is at most the depth h of T. Building on this observation, if x'' is a counterfactual of distance 2 from $x^\star$, there must be an instance $x' \in I_1$ such that x'' differs from

[2] Note that the term "query" here is being used in two separate ways. A variable being "queried" on a path simply means that variable appears as an internal node on that path. The IDT itself has "query access" to f meaning it may query the value of $f(x)$ for various x in its implementation.

x' on a single feature queried on the path π that x' follows in T. This reasoning leads to a bound of $h^{O(\Delta_f(x^\star))}$ on the size of our search space—all paths in T of "path distance" $\Delta_f(x^\star)$ from ρ—which stands in contrast to $d^{O(\Delta_f(x^\star))}$, the size of the search space of ball search.

3.1. A helper algorithm that finds minimal counterfactuals

Our algorithm for finding optimal countefactuals will call on a subroutine, FINDMINIMAL (Figure 1), for finding *minimal* counterfactuals:

Definition 3.1 (Minimal counterfactual)**.** For a model $f : \{0,1\}^d \to \{0,1\}$ and an instance $x^\star$, a *minimal counterfactual for $x^\star$ among $I \subseteq \{0,1\}^d$* is an instance $x' \in I$ such that:

1. $f(x') \neq f(x^\star)$, and
2. Let $V \subseteq [d]$ denote the features that $x^\star$ and x' differ on. Then $f(x'') = f(x^\star)$ for all instances $x'' \in I$ that differ from $x^\star$ only on the features in a strict subset $U \subset V$.

We observe that optimal counterfactuals are minimal, but minimal counterfactuals may not be optimal.

Our pseudocode for FINDMINIMAL uses the following notation. Recall that a restriction is a partial function $\rho : [d] \rightharpoonup \{0,1\}$, which we can equivalently represent as a string $\rho \in \{0,1,*\}^d$. We use the notation $\rho_{i \leftarrow b}$ to denote the restriction ρ with ρ_i overwritten with b. For $x^\star \in \{0,1\}^d$, we define $x^\star$ overwritten by ρ, denoted $x^\star_\rho \in \{0,1\}^d$, to be the following hybrid input: for all $i \in [d]$,

$$(x^\star_\rho)_i = \begin{cases} x^\star_i & \text{if } \rho_i = * \\ \rho_i & \text{if } \rho_i \neq *. \end{cases}$$

3.1.1. ANALYSIS OF FINDMINIMAL

Lemma 3.2 (Correctness of FINDMINIMAL)**.** FINDMINIMAL$_f(\rho, x^\star, k, A)$ *returns all minimal counterfactuals x' for $x^\star$ among those that are consistent with ρ and satisfy $|\Delta_f(x', x^\star_\rho)| \leq k$.*

Proof. We begin by noting all instances that are consistent with ρ differ from $x^\star$ on at least the features on which $x^\star$ and $x^\star_\rho$ differ. Therefore, if $f(x^\star_\rho) \neq f(x^\star)$, FINDMINIMAL correctly returns $\{x^\star_\rho\}$ in Line 1.

We therefore assume that $f(x^\star_\rho) = f(x^\star)$ and proceed by induction on the height of the subtree of T rooted at ρ. If ρ is a leaf, then f takes the same value on all instances x' that are consistent with ρ, and hence $f(x') = f(x^\star_\rho) = f(x^\star)$. In this case FINDMINIMAL correctly returns $\varnothing$ in Line 2.

For the inductive step, suppose ρ is an internal node. If $k = 0$, the only instance that is of distance 0 from $x^\star_\rho$ is $x^\star_\rho$ itself, so FINDMINIMAL again correctly returns $\varnothing$ in Line 2. Otherwise, the search space of all instances x' that are consistent with ρ and satisfy $\Delta_f(x', x^\star_\rho) \leq k$ can be partitioned into:

- Those that are consistent with $\rho_{i \leftarrow x^\star_i}$ and satisfy $|\Delta_f(x', x^\star_{\rho_{i \leftarrow x^\star_i}})| \leq k$
- Those that are consistent with $\rho_{i \leftarrow \overline{x}^\star_i}$ and satisfy $|\Delta_f(x', x^\star_{\rho_{i \leftarrow \overline{x}^\star_i}})| \leq k - 1$,

where i is the feature that is queried at ρ (Line 3a). FINDMINIMAL recurses on these two sets in Line 3b, and correctness follows by our induction hypothesis. □

Remark 3.3 (Correctness of auxiliary information). Our proof of Lemma 3.2 relies on the correctness of the IDT operations, which only hold if our calls to them use the "correct" auxiliary information A. This will be the case given the way we use FINDMINIMAL in our overall algorithm FINDOPTIMAL. FINDOPTIMAL calls FINDMINIMAL with ρ being the empty restriction and $A = \varnothing$. In its recursive execution, FINDMINIMAL with auxiliary information A recursively calls itself with A' where A' is the auxiliary information returned by QUERY(ρ, A) in Line 3a: the correctness of A' follows from the correctness of A and the QUERY operation.

Lemma 3.4 (Query complexity of FINDMINIMAL)**.** FINDMINIMAL$_f(\rho, x, k, A)$ *makes $h^{O(k)}$ calls to the IDT operations and $h^{O(k)}$ additional queries to f, where h is the height of the subtree of T rooted at ρ.*

Proof. We claim that both quantities are bounded by $Q(h,k) \leq 2(h+1)^k$.The proof is by induction on h. When $h = 0$ (i.e. ρ is a leaf in T), FINDMINIMAL queries f on $x^\star$ and $x^\star_\rho$, makes one call to ISLEAF, and terminates. When $h \geq 1$, there are at most two queries to f (on $x^\star$ and $x^\star_\rho$), two calls to IDT operations (ISLEAF and QUERY), in addition to those made recursively. We therefore have the following recursive relation:

$$Q(h,k) \leq 2 + Q(h-1,k) + Q(h-1,k-1).$$

Applying the induction hypothesis, we can bound $Q(h,k)$ by

$$Q(h,k) \leq 2 + 2h^k + 2h^{k-1} \leq 2(h+1)^k. \qquad \square$$

3.2. Our algorithm for finding optimal counterfactuals

With the helper algorithm FINDMINIMAL in hand, our overall algorithm for finding optimal counterfactuals is simple:

Theorem 1.2 is a straightforward consequence of Lemmas 2.2, 3.2 and 3.4:

FINDMINIMAL$_f(\rho, x^\star, k, A)$:

Given: Access to f via queries and IDT operations for a tree T that represents f. Restriction ρ corresponding to a path in T, instance $x^\star$, distance bound k, auxiliary information A.

Output: All minimal counterfactuals x' for $x^\star$ among those that are consistent with ρ and satisfy $|\Delta_f(x', x^\star_\rho)| \le k$.

1. If $f(x^\star) \neq f(x^\star_\rho)$, return $\{x^\star_\rho\}$.
2. Else if ISLEAF(ρ, A) or $k = 0$, return $\varnothing$.
3. Else:
   (a) Let $(i, A') \leftarrow$ QUERY(ρ, A).
   (b) Return FINDMINIMAL$_f(\rho_{i \leftarrow x^\star_i}, x^\star, k, A') \cup$ FINDMINIMAL$_f(\rho_{i \leftarrow \overline{x}^\star_i}, x^\star, k-1, A')$.

Figure 1: Helper algorithm FINDMINIMAL

FINDOPTIMAL$_f(x^\star)$:

Given: Instance $x^\star$ and queries to f.

Output: The set $\mathcal{C}$ of all optimal counterfactuls for $x^\star$.

1. Initialize $k \leftarrow 1, \mathcal{C} \leftarrow \varnothing$.
2. While $\mathcal{C} \neq \varnothing$:
   (a) Let $\mathcal{C} \leftarrow$ FINDMINIMAL$_f(*^d, x^\star, k, \varnothing)$.
   (b) $k \leftarrow k + 1$.
3. Return $\mathcal{C}$.

Figure 2: Algorithm for finding optimal counterfactuals, using FINDMINIMAL as its main subroutine.

Theorem 3.5 (Formal version of Theorem 1.2)**.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a monotone function. Then* FINDOPTIMAL$_f(x^\star)$*, using the IDT operations given in Lemma 2.2, returns all optimal counterfactuals for $x^\star$ and makes*

$$S(f)^{O(\Delta_f(x^\star))} \cdot \log d$$

queries to f.

Proof. The correctness of FINDOPTIMAL follows from that of FINDMINIMAL: FINDOPTIMAL calls FINDMINIMAL$_f(*^d, x^\star, k, \varnothing)$ for $k = 1, 2, 3, \ldots$ until a call returns a nonempty set. This happens exactly when k reaches $\Delta_f(x^\star)$.

It remains to analyze the query complexity of FINDOPTIMAL. By Lemma 2.2, the height of T is at most $S(f)^{O(1)}$. Therefore by Lemma 3.4 we have that FINDMINIMAL$_f(*^d, x^\star, k, \emptyset)$ uses at most $S(f)^{O(k)}$ IDT operations and additional queries to f. By Lemma 2.2, each IDT operation can be supported with $S(f)^{O(1)} \cdot \log d$ queries to f. The overall query complexity of FINDOPTIMAL is therefore upper bounded by

$$\sum_{k=1}^{\Delta_f(x^\star)} S(f)^{O(k)} \cdot S(f)^{O(1)} \cdot \log d \le S(f)^{O(\Delta_f(x))} \cdot \log d.$$

□

4. Building efficient implicit decision trees: Proof of Lemma 2.2

In this section, we prove Lemma 2.2. Our implementation of IDTs will be based on known algorithms for finding *certificates*. These certificates will be stored in the auxiliary information of the IDT and will be used to select which features to query in the decision tree.

4.1. Certificate complexity and query-efficient certificate finding

At a high level, a certificate of f on $x^\star \in \{0,1\}^d$ is a set of features $W \subseteq [d]$ that "witness" f's value on $x^\star$ in the sense that $f(y) = f(x^\star)$ for all $y \in \{0,1\}^d$ that agree with $x^\star$ on the coordinates in W.

Definition 4.1 (Certificate complexity)**.** For a model $f : \{0,1\}^d \to \{0,1\}$ and an input $x^\star$, the *complexity of certifying f's value on $x^\star$*, $C(f, x^\star)$, is the quantity:

$$\min_{W \subseteq [d]} \left\{ |W| \colon f(y) = f(x^\star) \text{ for all } y \text{ s.t. } y_W = x^\star_W \right\}.$$

The *certificate complexity of f* is the quantity $C(f) := \max_{x \in \{0,1\}^d} \{C(f, x)\}$.

Example. Because the set $W = [d]$ is always a certificate of f on $x^\star$, we typically want *small* certificates satisfying $|W| \leq C(f)$. For a specific example, consider the function $f : \{0,1\}^5 \to \{0,1\}$ defined by the decision tree in Figure 3. Then $W = \{1,3,5\}$ is a certificate of f on $x^\star = 10000$ since f outputs 0 on any input y satisfying $y_1 = 1, y_3 = 0$, and $y_5 = 0$. Indeed, for an arbitrary instance $x^\star$ the indices queried along the root-to-leaf path for $x^\star$ constitute a certificate of f on $x^\star$.

In general, we say $W \subseteq [d]$ is a certificate of f if there exists some $x \in \{0,1\}^d$ such that W is a certificate of f on x. Our algorithm relies on a query-efficient procedure $\textsc{Cert}(f)$ which takes a monotone f and returns the features in small certificate of f. The features in this certificate will be exactly the features we query in our IDT for f. "Small certificate" here means having size at most $C(f)^{O(1)}$. For monotone functions, this size bound is equivalently $S(f)^{O(1)}$ using the fact that certificate complexity equals sensitivity for monotone models f:

Fact 4.2 ((Nisan, 1989)). *For a monotone model $f : \{0,1\}^d \to \{0,1\}$, $S(f) = C(f)$.*

The authors of (Blanc et al., 2022) recently developed a query-efficient implementation of $\textsc{Cert}(\cdot)$ for monotone f. We restate their result here in terms of $S(f)$ using Fact 4.2.

Theorem 4.3 (Theorem 5 of (Blanc et al., 2022)). *Let $f : \{0,1\}^d \to \{0,1\}$ be monotone. Then there is an algorithm that computes a certificate of f of size $S(f)^{O(1)}$ with high probability using $S(f)^{O(1)} \cdot \log d$ queries to f.*

4.2. Overall structure of T

Figure 3 depicts the overall structure of an IDT. Conceptually, the tree can be divided into subtree "blocks". For a subtree rooted at ρ, this block is the complete binary tree that exhaustively queries all features in $\textsc{Cert}(f_\rho)$, a small certificate for f_ρ. One call to $\textsc{Cert}(f_\rho)$ yields a set of features to query in the tree and so we store this set as auxiliary information until we need to call $\textsc{Cert}(\cdot)$ again. The overall depth of the tree is the number of consecutive blocks in a root-to-leaf path times the depth of each block. The depth of each block is the size of the certificate returned by $\textsc{Cert}(\cdot)$ and so Theorem 4.3 ensures that this depth is small. In the next section, we give the implementation details of the IDT operations which yield this IDT structure. Then, we show how to bound the number of consecutive blocks which provides a bound on the overall depth of the IDT.

4.3. Query-efficient implementation of IDT operations for navigating T

$\textsc{Query}_f(\cdot,\cdot)$. We assume there is some ordering on features $[d]$ and hence on elements of a subset $A \subseteq [d]$. Consider the following implementation of $\textsc{Query}_f$.

$\textsc{Query}_f(\rho, A)$:

1. if $A \subseteq \text{Dom}(\rho)$ then let $A' = \textsc{Cert}(f_\rho)$ and return the next feature of A';
2. otherwise, return the next feature in $A \setminus \text{Dom}(\rho)$ and set $A' = A$.

This procedure stores a list of features in the auxiliary information, A, which is then queried iteratively until all features have been exhausted after which a new set of features is computed using $\textsc{Cert}$. The set A specifies a complete binary tree of depth $|A|$ – the tree which exhaustively queries all of the features in A. In this way, the entire IDT is a collection of complete subtree blocks each of which originates from a certificate output by $\textsc{Cert}$. Figure 4 illustrates the computation path of a particular input through an IDT using auxiliary information to store subtree blocks.

$\textsc{IsLeaf}_f(\cdot,\cdot)$. When f is monotone, one can check whether it's the constant function by querying f on the all 0s input and the all 1s input. Therefore, to check whether ρ is a leaf, we can query f_ρ on these two inputs and output Yes if and only if f_ρ is determined to be the constant function.

4.4. Bounding the depth of T

We also use the following lemma (implicit in Theorem 6 of (Blanc et al., 2022)) which states that f quickly becomes the constant function after being repeatedly restricted by a certificate. Alternatively, consecutive blocks in an IDT quickly terminate in a leaf node. More specifically, suppose $A = \textsc{Cert}(f)$ is a certificate for f. So, there is some restriction $\rho : [d] \to \{0,1\}$ with $\text{Dom}(\rho) = A$ such that f_ρ is a constant function. Equivalently, there is some way of setting each feature in A to a value in $\{0,1\}$ such that f_ρ is the constant function. In general, for an *arbitrary* restriction $\rho : [d] \to \{0,1\}$ with $\text{Dom}(\rho) = A$ (an arbitrary setting of each feature in A to $\{0,1\}$), f_ρ may be nonconstant. However, one can show f_ρ is "closer" to a constant than f in the sense that f_ρ has smaller certificate complexity. The following lemma captures this behavior and states more specifically that f becomes constant after $O(S(f))$ many such restrictions. For more details, see the proof of the lemma in Appendix A.

Lemma 4.4 (Implicit in Theorem 6 of (Blanc et al., 2022)). *Let $f : \{0,1\}^d \to \{0,1\}$ be monotone. Let $\rho_1, \ldots, \rho_\ell$ be a series of restrictions and $A_1, \ldots, A_\ell \subseteq [d]$ be disjoint such that for all $i \in [\ell]$ $\text{Dom}(\rho_i) = A_1 \cup \cdots \cup A_i$ and each A_{i+1} is a certificate for f_{ρ_i}. Then $\ell = 2 \cdot S(f)$ suffices to ensure that f_{ρ_ℓ} is constant.*

4.5. Putting it all together: Proof of Lemma 2.2

We prove that our implementations of $\textsc{Query}_f$ and $\textsc{IsLeaf}_f$ suffice to prove the lemma statement. First, we show that the implementations require at most $S(f)^{O(1)} \cdot \log d$ queries to f, then we show the depth of the IDT is at most $S(f)^{O(1)}$.

The procedure $\textsc{IsLeaf}$ uses $O(1)$ queries to f. A call to $\textsc{Query}(\rho, A)$ returns either a feature in A or queries $\textsc{Cert}(f)$. Theorem 4.3 shows that $\textsc{Cert}(f)$ requires at most $S(f)^{O(1)} \cdot \log d$ to f and hence $\textsc{Query}(\rho, A)$ requires at most $S(f)^{O(1)} \cdot \log d$ queries to f.

It remains to show that the depth of the IDT is at most $S(f)^{O(1)}$. Let ρ be an arbitrary leaf of the IDT. By construction, $\text{Dom}(\rho)$ can be partitioned into blocks of features $A_1, \ldots, A_\ell$ where each A_i is a certificate of f restricted on the features in $A_1 \cup \cdots \cup A_{i-1}$. Lemma 4.4 ensures that f becomes constant after being restricted by $\ell = S(f)^{O(1)}$ many such certificates. □

5. Functions over general feature spaces: Proof of Theorem 1.3

For a monotone model $f : X^d \to \{0,1\}$ and an instance $x^\star \in X^d$, we reduce the problem of computing $\mathcal{C}_f(x^\star)$ to the problem of computing $\mathcal{C}_{f_{x^\star}}(y)$ where $f_{x^\star} : \{0,1\}^d \to \{0,1\}$ is a Boolean model defined in terms of f and $x^\star$ and $y \in \{0,1\}^d$ is a specific Boolean input to $f_{x^\star}$.

The reduction. We assume $f : X^d \to \{0,1\}$ is monotone with respect to some total ordering on X. Write $\overline{X}$ for the top element of X and $\underline{X}$ for the bottom element of X. Assuming f is nonconstant, we then have $f(\overline{X}^d) = 1$ and $f(\underline{X}^d) = 0$. For $x \in X^d$ and $y \in \{0,1\}^d$, we define $x \uparrow y, x \downarrow y \in X^d$ as instances satisfying

$$(x \uparrow y)_i = \begin{cases} \overline{X} & y_i = 1 \\ x_i & y_i = 0 \end{cases} \qquad (x \downarrow y)_i = \begin{cases} x_i & y_i = 1 \\ \underline{X} & y_i = 0 \end{cases}$$

for all $i \in [d]$. That is, $x \uparrow y$ is the instance formed by "snapping" a subset of features, specified by y, to their largest possible value $\overline{X}$. In $x \downarrow y$, features are snapped in the opposite direction. Note that relative to the ordering $\leq$ on X, we have $x \downarrow y \leq x \leq x \uparrow y$ for all $y \in \{0,1\}^d$. Given a monotone $f : X^d \to \{0,1\}$, we write $x \uparrow\downarrow_f y$ to denote snapping x in a direction specified by $f(x)$:

$$x \uparrow\downarrow_f y = \begin{cases} x \downarrow y & f(x) = 1 \\ x \uparrow y & f(x) = 0. \end{cases}$$

We will just write $x \uparrow\downarrow y$ when f is known from context. We then define the Boolean function $f_x : \{0,1\}^d \to \{0,1\}$ as

$$f_x(y) = f(x \uparrow\downarrow y).$$

Note that f_x is monotone since $y \leq y'$ implies $x \uparrow\downarrow y \leq x \uparrow\downarrow y'$.

Properties of f_x. We establish two important properties of the reduction f_x. The first property states that the set $\mathcal{C}_f(x) = \{\Delta(x', x) : x' \in X^d$ is an optimal counterfactual for x with respect to $f\}$ is equal to the set $\mathcal{C}_{f_x}(f(x)^d)$ where $f(x)^d \in \{0,1\}^d$ is the bit $f(x)$ repeated d times. The second property states that the sensitivity of f_x is at most the sensitivity of f. For a proof, see Appendix B.

Lemma 5.1. *Let $f : X^d \to \{0,1\}$ be monotone. Then for all $x \in X^d$,*

1. $\mathcal{C}_f(x) = \mathcal{C}_{f_x}(f(x)^d)$
2. $S(f_x) \leq S(f)$.

5.1. Proof of Theorem 1.3

Let $f : X^d \to \{0,1\}$ be monotone and $x^\star \in X^d$. Assume without loss of generality $f(x^\star) = 1$. Using Theorem 3.5, we compute and return the set $\mathcal{C}_{f_{x^\star}}(1^d)$ which queries f at most $S(f_{x^\star})^{O(\Delta_{f_{x^\star}}(1^d))}$ times. Lemma 5.1 states that $\mathcal{C}_{f_{x^\star}}(1^d) = \mathcal{C}_f(x^\star)$ which ensures we have returned the desired set. Moreover, since these two sets are equal we have that $\Delta_{f_{x^\star}}(1^d) = \Delta_f(x^\star)$. We can then write the query bound as

$$\begin{aligned} S(f_{x^\star})^{O(\Delta_{f_{x^\star}}(1^d))} \log d &= S(f_{x^\star})^{O(\Delta_f(x^\star))} \log d \\ &\leq S(f)^{O(\Delta_f(x^\star))} \log d \\ &\qquad \text{(Lemma 5.1)} \end{aligned}$$

which completes the proof.

6. Conclusion

We have given a query-optimal algorithm for finding counterfactuals for monotone models. Our algorithm achieves a logarithmic dependence on the dimension d, which is exponentially smaller than the previous best known query complexity of $d^{O(\Delta_f(x^\star))}$, achievable by brute-force local search.

There are several concrete avenues for future work. The key enabling ingredient in our overall algorithm is our implicit decision tree algorithm for monotone models. This gives us a way to efficiently navigate an interpretable representation of a black box model f — a decision tree representation — even in the case of complex models for which this representation is intractably large. This is a potentially versatile technique for understanding black box models — can we use it to efficiently compute other types of explanations? The query complexity of our algorithm scales with the sensitivity of f, a discrete analogue of the Lipschitz constant. Can our lower bounds be evaded by running our algorithm,

or variants of it, for *smoothed* versions $\tilde{f}$ of f, obtained from f by adding a small amount of noise? Finally, while we have focused on sparsity as our notion of distance in this work, it would be interesting to extend our techniques to accommodate other distance functions.

Acknowledgments

We thank the ICML reviewers for their useful comments and feedback.

Guy, Caleb, and Li-Yang are supported by NSF CAREER Award 1942123. Caleb is also supported by an NDSEG fellowship. Jane is supported by NSF Award CCF-2006664.

References

Backurs, A., Blum, A., and Gupta, N. Active local learning. In *Proceedings of the 33rd Conference On Learning Theory (COLT)*, pp. 363–390. Proceedings of Machine Learning Research, 2020.

Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., and Müller, K.-R. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(61):1803–1831, 2010.

Bastani, O., Kim, C., and Bastani, H. Interpretability via model extraction. In *Proceedings of the 4th Workshop on Fairness, Accountability, and Transparency in Machine Learning (FAT/ML)*, 2017.

Blanc, G., Lange, J., and Tan, L. Provably efficient, succinct, and precise explanations. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021 (NeurIPS)*, 2021.

Blanc, G., Koch, C., Lange, J., and Tan, L.-Y. The Query Complexity of Certification. *ArXiv*, abs/2201.07736, 2022.

Blum, A. and Hu, L. Active tolerant testing. In *Proceedings of the 31st Conference On Learning Theory (COLT)*, volume 75, pp. 474–497, 2018.

Breiman, L. and Shang, N. Born again trees. Technical report, University of California, Berkeley, 1996.

Cook, S., Dwork, C., and Reischuk, R. Upper and lower time bounds for parallel random access machines without simultaneous writes. *SIAM Journal on Computing*, 15(1): 87–97, 1986.

Craven, M. and Shavlik, J. Extracting tree-structured representations of trained networks. *Proceedings of the 8th Conference on Advances in Neural Information Processing Systems (NeurIPS)*, 8:24–30, 1995.

Gopalan, P., Nisan, N., Servedio, R. A., Talwar, K., and Wigderson, A. Smooth boolean functions are easy: Efficient algorithms for low-sensitivity functions. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science (ITCS)*, pp. 59–70, 2016.

Huang, H. Induced subgraphs of hypercubes and a proof of the sensitivity conjecture. *Annals of Mathematics*, 190 (3):949–955, 2019.

Karimi, A.-H., Barthe, G., Balle, B., and Valera, I. Model-agnostic counterfactual explanations for consequential decisions. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 895–905. PMLR, 2020.

Knuth, D. A computational proof of huang's degree theorem, 2019. Available at https://www.cs.stanford.edu/~knuth/papers/huang.pdf.

Koh, P. W. and Liang, P. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 1885–1894, 2017.

Kong, W. and Valiant, G. Estimating learnability in the sublinear data regime. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems (NeurIPS)*, pp. 5460–5469, 2018.

Kong, W., Valiant, G., and Brunskill, E. Sublinear optimal policy value estimation in contextual bandits. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.

Lakkaraju, H., Kamar, E., Caruana, R., and Leskovec, J. Faithful and customizable explanations of black box models. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society (AIES)*, pp. 131–138, 2019.

Lakkaraju, H., Arsov, N., and Bastani, O. Robust and stable black box explanations. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119, pp. 5628–5638, 2020.

Lundberg, S. M. and Lee, S.-I. A unified approach to interpreting model predictions. In *Proceedings of the 31st Annunal Conference on Advances in Neural Information Processing Systems (NeurIPS)*, pp. 4765–4774, 2017.

Morgan, S. L. and Winship, C. *Counterfactuals and causal inference*. Cambridge University Press, 2015.

Mothilal, R. K., Sharma, A., and Tan, C. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pp. 607–617, 2020.

Nisan, N. CREW PRAMs and decision trees. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC)*, pp. 327–335, 1989.

Nisan, N. and Szegedy, M. On the degree of boolean functions as real polynomials. *Computational complexity*, 4 (4):301–313, 1994.

Pearl, J. Causal inference in statistics: An overview. *Statistics surveys*, 3:96–146, 2009a.

Pearl, J. *Causality*. Cambridge university press, 2009b.

Ribeiro, M. T., Singh, S., and Guestrin, C. "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 1135–1144, 2016.

Ribeiro, M. T., Singh, S., and Guestrin, C. Anchors: High-precision model-agnostic explanations. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI)*, pp. 1527–1535, 2018.

Russell, C. Efficient search for diverse coherent explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pp. 20–28, 2019.

Simonyan, K., Vedaldi, A., and Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop at International Conference on Learning Representations (ICLR)*, 2014.

Strumbelj, E. and Kononenko, I. An efficient explanation of individual classifications using game theory. *Journal of Machine Learning Research*, 11:1–18, March 2010. ISSN 1532-4435.

Ustun, B., Spangher, A., and Liu, Y. Actionable recourse in linear classification. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pp. 10–19, 2019.

Van Assche, A. and Blockeel, H. Seeing the forest through the trees: Learning a comprehensible model from an ensemble. In *European Conference on Machine Learning (ECML)*, pp. 418–429, 2007.

Vandewiele, G., Lannoye, K., Janssens, O., Ongenae, F., De Turck, F., and Van Hoecke, S. A genetic algorithm for interpretable model extraction from decision tree ensembles. In *Trends and Applications in Knowledge Discovery and Data Mining*, pp. 104–115, 2017.

Vidal, T. and Schiffer, M. Born-again tree ensembles. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pp. 9743–9753, 2020.

Wachter, S., Mittelstadt, B., and Russell, C. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.*, 31:841, 2017.

Yao, A. C. C. Probabilistic computations: toward a unified measure of complexity. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 222–227, 1977.

Zhou, Y. and Hooker, G. Interpreting models via single tree approximation, 2016.

A. Certificate-based restrictions

In this section, we prove Lemma 4.4. We say a certificate $S \subseteq [n]$ of f on $x \in \{0,1\}^d$ is a 0-certificate if $f(x) = 0$ and likewise S is a 1-certificate if $f(x) = 1$. We write $C_0(f) = \max_{x \in f^{-1}(0)}\{C(f,x)\}$ to denote the 0-certificate complexity of f and $C_1(f) = \max_{x \in f^{-1}(1)}\{C(f,x)\}$ for the 1-certificate complexity of f. Our proof will reference the fact that the intersection of a 1-certificate with a 0-certificate is necessarily nonempty (since otherwise there would be some instance $x \in \{0,1\}^d$ having both a 0-certificate and a 1-certificate).

Fact A.1. *Let S_0 be a 0-certificate of $f : \{0,1\}^d \to \{0,1\}$ and let S_1 be a 1-certificate of f. Then $S_0 \cap S_1 \neq \varnothing$.*

Proof of Lemma 4.4. Let ρ be a restriction of $f : \{0,1\}^d \to \{0,1\}$ such that $\mathrm{Dom}(\rho) = \textsc{Cert}(f)$. We'll show that $C_0(f_\rho) + C_1(f_\rho) \leq C_0(f) + C_1(f) - 1$. The result then follows by induction. Suppose without loss of generality that $\mathrm{Dom}(\rho)$ is a 1-certificate of f. Then we claim $C_0(f_\rho) \leq C_0(f) - 1$. Consider any $x \in f_\rho^{-1}(0)$. Consider the string $x' \in \{0,1\}^d$ formed by inserting ρ into the string x so that $f(x') = f_\rho(x)$. Let S_0 be a 0-certificate of f on x' with $|S_0| \leq C_0(f)$. Then $S_0 \setminus \mathrm{Dom}(\rho)$ is a 0-certificate of f_ρ on x. Moreover, $S_0 \cap \mathrm{Dom}(\rho) \neq \varnothing$ by Fact A.1 and so $|S_0 \setminus \mathrm{Dom}(\rho) \leq |S_0| - 1 \leq C_0(f) - 1$. Since x is any arbitrary 0-input to f_ρ, we have that $C_0(f_\rho) \leq C_0(f) - 1$ as desired.

It follows each such restriction ρ decreases $C_0(f) + C_1(f)$ by at least 1. Thus, after at most $2C(f) \geq C_0(f) + C_1(f)$ restrictions f becomes the constant function. □

B. Proof of Lemma 5.1

For (1), fix an input $x \in X^d$. Suppose without loss of generality that $f(x) = 1$ (the arguments for $f(x) = 0$ are symmetric) so that $f_x(1^d) = f(x \downarrow 1^d) = f(x) = 1$. We will show that if $y \in \{0,1\}^d$ is a counterfactual for 1^d then there exists a counterfactual $x' \in X^d$ for x satisfying $\Delta(x,x') = \Delta(y,1^d)$ and likewise if there is a counterfactual x' for x then there is a counterfactual y for 1^d satisfying the same $\Delta(x,x') = \Delta(y,1^d)$. It then follows that $\Delta_f(x) = \Delta_{f_x(1^d)}$ and also $\mathcal{C}_f(x) = \mathcal{C}_{f_x}(1^d)$. Let $y \in \{0,1\}^d$ be a counterfactual for 1^d. Then,

$$f_x(y) = f(x \downarrow y) = 0$$

which shows that $x \downarrow y \in X^d$ is a counterfactual for x. Moreover, by definition,

$$\Delta(x \downarrow y, x) = \Delta(y, 1^d).$$

On the other hand, suppose $x' \in X^d$ is a counterfactual for x. Let $y \in \{0,1\}^d$ be the instance defined by

$$y_i = \begin{cases} 0 & i \in \Delta(x,x') \\ 1 & i \notin \Delta(x,x') \end{cases}$$

for all $i \in [d]$. Then $x \downarrow y \leq x'$ which implies

$$f_x(y) = f(x \downarrow y) \leq f(x') = 0$$

by monotonicity. Hence, y is a counterfactual for 1^d and again we have $\Delta(x \downarrow y, x) = \Delta(y, 1^d)$.

For (2), let $y \in \{0,1\}^d$ be an instance satisfying $S_{f_x}(y) = S(f_x)$. Then $f_x(y) = f(x \uparrow\downarrow y)$. In particular,

$$S(f_x) = S_{f_x}(y) = S_f(x \uparrow\downarrow y) \leq S(f)$$

as desired. □

C. Lower bounds

C.1. Optimality of Theorem 3.5

We prove the following (slightly more general and formal) version of Theorem 1.4.

Theorem C.1. *Fix any constant $c < 1$ and success probability $\varepsilon > 0$. For any algorithm $\mathcal{A}$, and $S, d, \Delta \in \mathbb{N}$ such that $S \leq d$ and $\Delta \leq S^c$, there is a monotone model $f : \{0,1\}^d \to \{0,1\}$ with $S(f) \leq S$ and instance $x^\star$ with $\Delta_f(x^\star) = \Delta$ such that $\mathcal{A}$ must make*

$$q = S^{\Omega(\Delta)} + \Omega(\log d)$$

queries to find a single optimal counterfactual for $x^\star$ with success probability ε.

We'll allow the algorithm $\mathcal{A}$ to use randomness. In order to give lower bounds against randomized algorithms, we'll use the easy direction of Yao's Lemma.

Lemma C.2 ((Yao, 1977))**.** *For any $q \in \mathbb{N}$, let $\mathcal{R}_q$ and $\mathcal{D}_q$ be the set of all q-query randomized and deterministic algorithms respectively, and let I be the set all of possible pairs $f : \{0,1\}^n \to \{0,1\}$ and $x^\star \in \{0,1\}^n$ (i.e. instances of the counterfactual problem).*

For any distribution μ supported on I,

$$\begin{aligned}&\min_{R \in \mathcal{R}_q} \max_{(f,x^\star) \in I} [\mathrm{error}_R(f, x^\star)] \\ &\quad \geq \min_{D \in \mathcal{D}_q} \mathop{\mathbb{E}}_{(\boldsymbol{f}, \boldsymbol{x}^\star) \sim \mu} [\mathrm{error}_D(\boldsymbol{f}, \boldsymbol{x}^\star)]\end{aligned}$$

where $\mathrm{error}_R(f, x^\star)$ is the probability that R does not return an optimal counterfactual for $x^\star$, and $\mathrm{error}_D(f, x^\star) = \mathbb{1}[D$ does not return an optimal counterfactual for $x^\star]$.

First, we show that $s^{\Omega(\Delta)}$ queries are necessary.

Lemma C.3. *For any $\Delta, d, S, q \in \mathbb{N}$ where $S \leq d$, and $\mathcal{A}$ a q-query randomized algorithm, there exists some monotone function $f : \{0,1\}^d \to \{0,1\}$ with $S(f) \leq S$ and input $x^\star \in \{0,1\}$ satisfying $\Delta_f(x^\star) = \Delta$ on which $\mathcal{A}$ successfully returns an optimal counterfactual for $x^\star$ with probability at most $\frac{q+1}{\binom{S}{\Delta}}$.*

Proof. We will apply Yao's Lemma: To apply it, we need to define a distribution over input instances $(\boldsymbol{f}, \boldsymbol{x}^\star)$. That distribution is simple. With probability 1, $\boldsymbol{x}^\star = (0, \ldots, 0)$. Then, we sample a uniformly random $\boldsymbol{z}$ from the $\binom{S}{\Delta}$ elements of $\{0,1\}^S$ with Hamming weight equal to Δ and set $\boldsymbol{f} : \{0,1\}^d \to \{0,1\}$ as

$$\boldsymbol{f}(x) \coloneqq \begin{cases} 1 \mid \geq \Delta + 1 \text{ of the first } s \text{ coordinates of } x \text{ are } 1 \\ 1 \mid \text{the first } s \text{ coordinates of } x \text{ equal } \boldsymbol{z} \\ 0 \mid \text{otherwise.} \end{cases}$$

First, we note that it is always true that $S(\boldsymbol{f}) \leq S$. This is because $\boldsymbol{f}(x)$ only depends on the first S-coordinates of x, so flipping any of the other $d - S$ coordinates cannot change $\boldsymbol{f}$'s output. Second, we note that the input $\boldsymbol{x}'$ being $\boldsymbol{z}$ concatenated with $d - S$ many 0s satisfies $\boldsymbol{f}(x^\star) \neq \boldsymbol{f}(\boldsymbol{x}')$ and $|x^\star - \boldsymbol{x}'| = \Delta$. Therefore, $\Delta_{\boldsymbol{f}}(x^\star) \leq \Delta$. Furthermore, $\boldsymbol{x}'$ is the *only* counterfactual for $x^\star$ at distance $\leq \Delta$, and so the algorithm succeeds if and only if it outputs $\boldsymbol{x}'$.

Consider an arbitrary q-query *deterministic* algorithm $\mathcal{A}'$. By Yao's Lemma, it is sufficient to prove that the probability $\mathcal{A}'$ outputs $\boldsymbol{x}'$ given the random instance $(\boldsymbol{f}, \boldsymbol{x}^\star)$ is at most $\frac{q+1}{\binom{S}{\Delta}}$. As $\mathcal{A}'$ is deterministic, the queries it makes and answer it outputs are a deterministic function of the answers to its previous queries. Let $g : \{0,1\}^d \to \{0,1\}$ be defined as

$$g(x) \coloneqq \begin{cases} 1 \mid \geq \Delta + 1 \text{ of the first } S \text{ coordinates of } x \text{ are } 1 \\ 0 \mid \text{otherwise.} \end{cases}$$

We will consider *one* possible execution path of $\mathcal{A}'$, when $\boldsymbol{f}$ and g give the same answer on each query $\mathcal{A}'$ makes. Specifically, the execution path of $\mathcal{A}'$ when every query it makes is answered consistently with g. In that case, $\mathcal{A}'$ will make

(the deterministic) queries $x^{(1)}, \ldots, x^{(q)}$. We compute

$$\begin{aligned}
\Pr[\text{Any query inconsistent with } g] &\leq \sum_{j=1}^{q} \Pr[\boldsymbol{f}(x^{(j)}) \neq g(x^{(j)})] && \text{(Union bound)} \\
&= \sum_{j=1}^{q} \Pr[\text{The first } S \text{ coordinates of } x^{(j)} \text{ equal } \boldsymbol{z}] \\
&\leq \frac{q}{\binom{S}{\Delta}}.
\end{aligned}$$

Lastly, if $\mathcal{A}'$ follows the single execution path corresponding to every query it makes being answered consistently with g, it will output some answer $x^{(g)}$. This answer is correct if and only if it equals $\boldsymbol{x}'$, which occurs with probability at most $\frac{1}{\binom{S}{\Delta}}$ for any choice of $x^{(g)}$. Therefore,

$$\begin{aligned}
&\Pr[\mathcal{A}' \text{ returns an optimal counterfactual}] \\
&\quad = \Pr[\mathcal{A}' \text{ returns an optimal counterfactual}, \text{All queries consistent with } g] \\
&\qquad + \Pr[\mathcal{A}' \text{ returns an optimal counterfactual}, \text{Any query inconsistent with } g] \\
&\quad \leq \Pr[x^{(g)} = \boldsymbol{z}] + \Pr[\text{Any query inconsistent with } g] \\
&\quad \leq \frac{1}{\binom{S}{\Delta}} + \frac{q}{\binom{S}{\Delta}} = \frac{q+1}{\binom{S}{\Delta}}
\end{aligned}$$

The desired result follows from Yao's Lemma.

□

Next we show that $\log d$ queries are necessary.

Lemma C.4. *For any $d, q \in \mathbb{N}$ and $\mathcal{A}$ a q-query randomized algorithm, there exists a monotone model $f : \{0,1\}^d \to \{0,1\}$ satisfying $S(f) = 1$ and input $x^\star$ where $\Delta_f(x^\star) = 1$ on which $\mathcal{A}$ successful returns an optimal counterfactual for $x^\star$ with probability at most $\frac{2^q}{d}$.*

Proof. Once again, we will pick a distribution over input instances $(\boldsymbol{f}, \boldsymbol{x}^\star)$ and apply Yao's Lemma. With probability 1, we set $\boldsymbol{x}^\star = (0, \ldots, 0)$. Then, we sample a uniformly random $\boldsymbol{i} \in [d]$ and set

$$\boldsymbol{f}(x) = x_{\boldsymbol{i}}$$

First, we note that $S(\boldsymbol{f}) = 1$, as the f is only sensitive on the $\boldsymbol{i}^{\text{th}}$ coordinate. Furthermore, the input that is 1 on the $\boldsymbol{i}^{\text{th}}$ coordinate and 0 everywhere else is a counterfactual for $\boldsymbol{x}^\star$ of distance 1. This is the only optimal counterfactual.

All that remains is to argue that any q-query deterministic algorithm fails to return an optimal counterfactual for $\boldsymbol{x}^\star$. A q-query deterministic algorithm can only output 2^q different answers. However, based on the choice of $\boldsymbol{i}$, there are d unique choices for the optimal counterfactual. Therefore, the probability the algorithm can output the optimal counterfactual is at most $\frac{2^q}{d}$. □

Proof of Theorem C.1. It's sufficient to show that

$$q = \max(S^{\Omega(\Delta)}, \Omega(\log d))$$

where we treat ε and c as constants. By Lemma C.3, we have that

$$\begin{aligned}
q &\geq \varepsilon \cdot \binom{S}{\Delta} - 1 \\
&\geq \varepsilon \cdot \left(\frac{S}{\Delta}\right)^{\Delta} - 1 \\
&\geq \varepsilon \cdot \left(S^{1-c}\right)^{\Delta} - 1 && (\Delta \leq S^c) \\
&= S^{\Omega(\Delta)}.
\end{aligned}$$

Then, by Lemma C.4, $\frac{2^q}{d} \geq \varepsilon$, and so

$$q \geq \log(\varepsilon d) = \Omega(\log d).$$

□

C.2. Lower bound against local algorithms

Recall that an algorithm for finding a counterfactual for $x^\star$ is *local* if its first query is $x^\star$ and every subsequent query differs from a previous query by exactly one feature.

Theorem C.5. *For any local search algorithm $\mathcal{A}$, $d \in \mathbb{N}$, and $\varepsilon > 0$, there is a monotone model $f : \{0,1\}^d \to \{0,1\}$ with $S(f) = 1$ and an instance $x^\star$ with $\Delta_f(x^\star)$ such that $\mathcal{A}$ must make $q \geq \varepsilon d$ queries in order to find a single optimal counterfactual for $x^\star$.*

Proof. As in the proof of Lemma C.4, we apply Yao's Lemma with the distribution with $\boldsymbol{x}^\star = (0, \ldots, 0)$ with probability 1 and

$$\boldsymbol{f}(x) = x_{\boldsymbol{i}}$$

where $\boldsymbol{i} \in [d]$ is chosen uniformly at random. Once again, $S(\boldsymbol{f}) = 1$ as f is only sensitive on the i^{th} and there is a counterfactual from $\boldsymbol{x}^\star$ of distance 1.

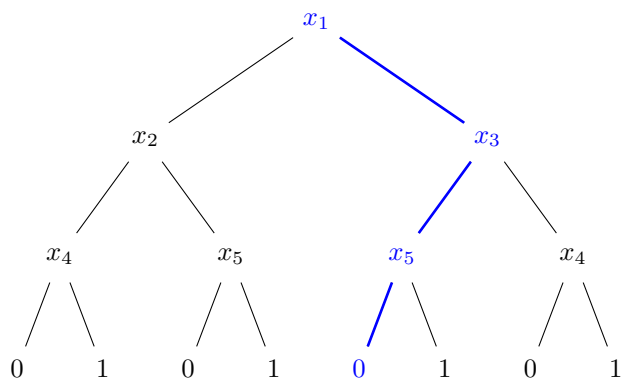
Since we are applying Yao's Lemma, it is sufficient to reason about deterministic local algorithms. $\mathcal{A}'$ be a q-query local and deterministic algorithm. We will consider one execution path of $\mathcal{A}'$, namely that in which all its queries return 0. In that case, it will make the deterministic queries $x^{(1)}, \ldots, x^{(q)}$. As $\mathcal{A}'$ is local, the queries $x^{(1)}, \ldots, x^{(q)}$ must all be adjacent, and since $x^{(1)} = (0, \ldots, 0)$, the number of coordinates $j \in [d]$ on which $x_j^{(t)} = 1$ for some $t \in [q]$ is at most $q - 1$. The probability that $\boldsymbol{f}(x^{(t)}) = 1$ for some $t \in [q]$ is the probability that $\boldsymbol{f}(x^{(t)})_{\boldsymbol{i}} = 1$ for some $t \in [q]$ which is at most $\frac{q-1}{d}$.

Therefore $\mathcal{A}'$ follows a single execution path with probability at least $1 - \frac{q-1}{d}$. At the end of that execution path, it outputs a single "guess" counterfactual y. However, y is an optimal counterfactual for $\boldsymbol{x}^\star$ only if $y_{\boldsymbol{i}} = 1$ and $y_j = 0$ for all $j \neq \boldsymbol{i}$, which can occur with probability at most $\frac{1}{d}$ for any single y. Therefore, the probability $\mathcal{A}'$ outputs a correct counterfactual is at most

$$\Pr[y \text{ is an optimal counterfactual for } \boldsymbol{x}^\star] + \Pr[\mathcal{A}' \text{ does not output } y] \leq \frac{1}{d} + \frac{q-1}{d} = \frac{q}{d}.$$

By applying Yao's Lemma, we conclude that any randomized q-query local algorithm succeeds in finding an optimal counterfactual with probability at most $\frac{q}{d}$. Therefore, to succeed with probability ε, we must have $q \geq \varepsilon d$. □

D. Figures



1. $\textsc{IsLeaf}_f(\{\}, \{\}) = \text{No}$
2. $\textsc{Query}_f(\{\}, \{\}) = (1, A_1)$
3. $\textsc{IsLeaf}_f(\{1 \mapsto 1\}, A_1) = \text{No}$
4. $\textsc{Query}_f(\{1 \mapsto 1\}, A_1) = (3, A_2)$
5. $\textsc{IsLeaf}_f(\{1 \mapsto 1, 3 \mapsto 0\}, A_2) = \text{No}$
6. $\textsc{Query}_f(\{1 \mapsto 1, 3 \mapsto 0\}, A_2) = (5, A_3)$
7. $\textsc{IsLeaf}_f(\{1 \mapsto 1, 3 \mapsto 0, 5 \mapsto 0\}, A_3) = \text{Yes}$

Figure 3: Walking down a tree T representing a function $f : \{0,1\}^5 \to \{0,1\}$ using the IDT operations. The input is $x^\star = 10000$ and its computation path through the tree is colored in blue. The walk starts at the root with the empty restriction $\{\}$ and ends at the leaf node represented by the restriction $\{1 \mapsto 1, 3 \mapsto 0, 5 \mapsto 0\}$. The decision tree leaf value here is 0 indicating that $T(x^\star) = f(x^\star) = 0$.

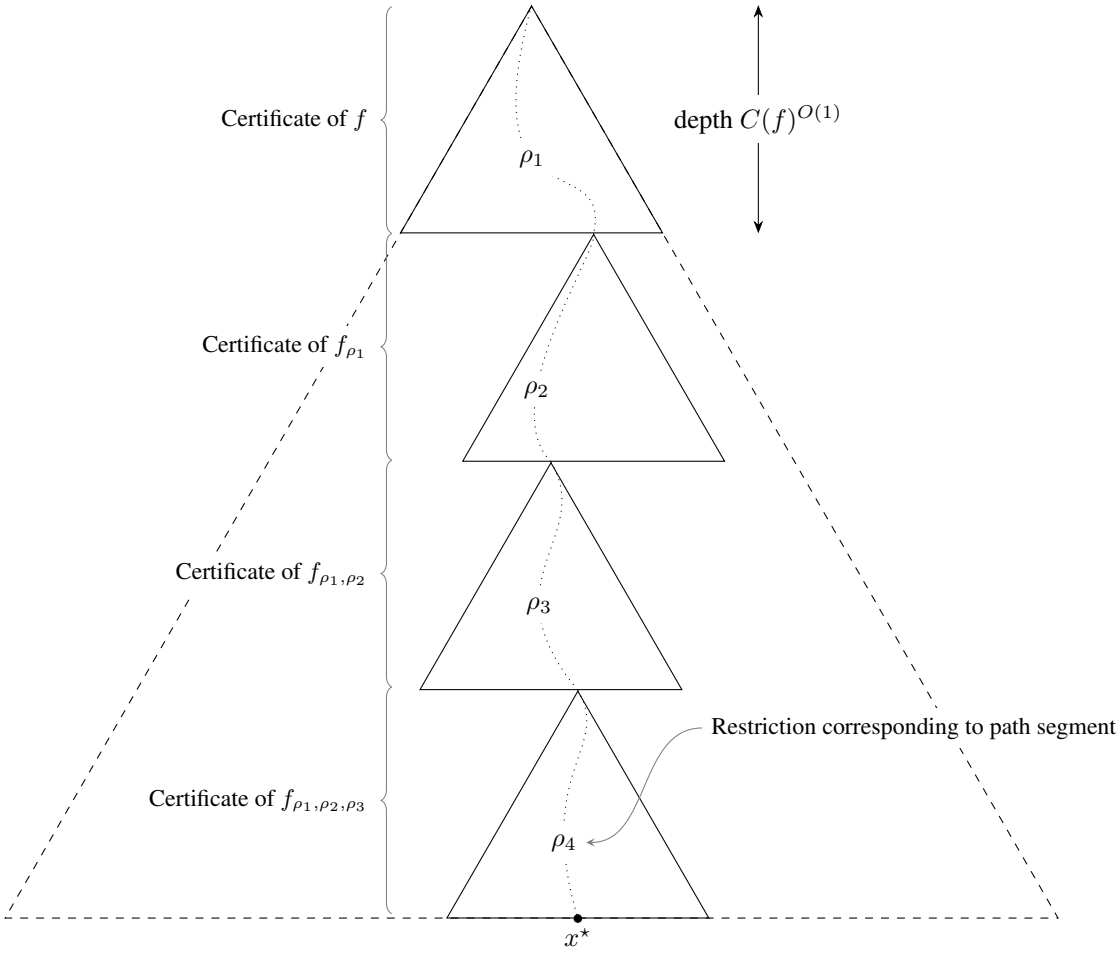


Figure 4: Tracing the root-to-leaf path of an input $x^\star \in \{0,1\}^d$ through an implicit decision tree (IDT) separated into blocks. The dashed triangle outline represents the entire tree. The IDT operations allow one to construct the dotted path without constructing the entire tree. Each triangular block corresponds to the subtree exhaustively querying all the features in a certificate of a restriction of f. The depth of each block is $C(f)^{O(1)}$, the number of features in a certificate of f. The total number of blocks intersecting any root-to-leaf path is at most $C(f)^{O(1)}$ and so the overall depth of the IDT is at most $C(f)^{O(1)}$.