Proceedings of Machine Learning Research vol 291:1–28, 2025 38th Annual Conference on Learning Theory

Faster Algorithms for Agnostically Learning Disjunctions and their Implications

Ilias Diakonikolas ILIAS@CS.WISC.EDU
University of Wisconsin-Madison

Daniel M. Kane DAKANE@CS.UCSD.EDU
University of California, San Diego

Lisheng Ren LREN29@WISC.EDU
University of Wisconsin-Madison

Editors: Nika Haghtalab and Ankur Moitra

Abstract

We study the algorithmic task of learning Boolean disjunctions in the distribution-free agnostic PAC model. The best known agnostic learner for the class of disjunctions over $\{0,1\}^n$ is the L_1-polynomial regression algorithm, achieving complexity $2^{\tilde{O}(n^{1/2})}$. This complexity bound is known to be nearly best possible within the class of Correlational Statistical Query (CSQ) algorithms. In this work, we develop an agnostic learner for this concept class with complexity $2^{\tilde{O}(n^{1/3})}$. Our algorithm can be implemented in the Statistical Query (SQ) model, providing the first separation between the SQ and CSQ models in distribution-free agnostic learning.

1. Introduction

A disjunction (resp. conjunction) over $\{0,1\}^n$ is an OR (resp. AND) of literals, where a literal is either a Boolean variable or its negation. While disjunctions are known to be efficiently learnable in Valiant's realizable PAC model Valiant (1984) (i.e., in the presence of clean/consistent labels), the learning task becomes substantially more challenging in the presence of partially corrupted labels. Here we study the task of learning disjunctions (or equivalently conjunctions) in the distribution-free agnostic PAC model Haussler (1992); Kearns et al. (1994). Agnostically learning disjunctions was one of the original problems studied by Kearns et al. (1994), and has since been highlighted in Avrim Blum's FOCS 2003 tutorial Blum (2003).

In the agnostic model, no assumptions are made about the labels, and the goal of the learner is to compute a hypothesis that is competitive with the best-fit function in the target class. For concreteness, we formally define the agnostic model in the Boolean setting below.

Definition 1 (Distribution-free Agnostic PAC learning) *Let $\mathcal{C}$ be a concept class of functions $f : \{0,1\}^n \to \{0,1\}$ and D be fixed but unknown distribution of $(\mathbf{x}, y)$ over $\{0,1\}^n \times \{0,1\}$. Given an error parameter $\epsilon \in (0,1)$ and sample access to D, the goal of an agnostic PAC learner $\mathcal{A}$ is to output a hypothesis $h : \{0,1\}^n \to \{0,1\}$ such that with high probability $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq \mathrm{OPT} + \epsilon$, where $\mathrm{OPT} \min_{f\in\mathcal{C}} \mathbf{Pr}_{(\mathbf{x},y)\sim D}[f(\mathbf{x}) \neq y]$. We say that $\mathcal{A}$ agnostically PAC learns $\mathcal{C}$ to error ϵ.*

Prior to this work, the fastest—and essentially only known non-trivial— algorithm for agnostically learning disjunctions was the L_1-polynomial regression algorithm Kalai et al. (2008a). As shown in that work, the L_1-regression algorithm agnostically learns disjunctions over $\{0,1\}^n$ up to excess error ϵ with sample and computational complexity bounded above by $2^{\tilde{O}(n^{1/2}\log(1/\epsilon))}$[1]. This complexity upper bound is tight, as a function of n, for the L_1-regression algorithm.

In terms of computational limitations, it is known that $2^{\Omega(n^{1/2})}$ is a complexity lower bound in various restricted models of computation, including Perceptron-based approaches Klivans and Sherstov (2010) and Correlational Statistical Query (CSQ) algorithms Gollakota et al. (2020). In the (more general) Statistical Query (SQ) model, to the best of our knowledge, the strongest known hardness result is a quasi-polynomial SQ lower bound Feldman (2009) that applies even under the uniform distribution. Finally, we note that Feldman et al. (2009) proved strong NP-hardness results for agnostically learning disjunctions with a halfspace hypothesis. This result does not rule out efficient improper learning.

The vast gap between known upper and lower bounds motivates further algorithmic investigation of this fundamental learning task. In this work, we give a new algorithm for agnostically learning disjunctions with substantially improved complexity. Specifically, we show the following:

Theorem 2 (Main Result) *There exists an algorithm that agnostically PAC learns the class of disjunctions over $\{0,1\}^n$ to error ϵ with sample and computational complexity $2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$.*

We give two algorithms that achieve the above guarantee (up to the $\tilde{O}()$ factor in the exponent). Our first algorithm is simpler and is slightly more efficient (with better logarithmic factors in the $\tilde{O}()$). Our second algorithm is implementable in the SQ model with complexity $2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$.

As a corollary, we obtain a super-polynomial separation between the CSQ and SQ models in the context of agnostic learning, answering an open problem posed in Gollakota et al. (2020). We elaborate on this connection in the proceeding discussion.

Discussion The CSQ model Bshouty and Feldman (2002) is a subset of the SQ model Kearns (1998), where the oracle access is of a special form (see Definition 8 and Appendix D). In the context of learning Boolean-valued functions, the two models are known to be equivalent in the distribution-specific setting (i.e., when the marginal distribution on feature vectors is known to the learner); see Bshouty and Feldman (2002). However, they are not in general equivalent in the distribution-free PAC model. In the realizable PAC setting, there are known natural separations between the CSQ and SQ models. Notably, Feldman (2011) showed that Boolean halfspaces are not efficiently CSQ learnable (even though they are efficiently SQ learnable).

In the agnostic PAC model studied here, we are not aware of a natural concept class separating the two models. Our algorithm provides such a natural separation for the class of disjunctions. In more detail, Gollakota et al. (2020) asks:

> "Our CSQ lower bounds do not readily extend to the general SQ model, and a very compelling direction for future work is to investigate whether such an extension is possible".

We answer this question in the negative by establishing a super-polynomial separation between the CSQ and SQ models for agnostically learning one of the most basic concept classes.

1. Throughout this paper, we will assume that the failure probability δ is a small universal constant, e.g., $\delta = 1/10$. Standard arguments can boost this to any desired δ with only a $\mathrm{polylog}(1/\delta)$ complexity blowup.

It is worth pointing out that the L_1-regression algorithm is known to be implementable in the SQ model (but not in CSQ). We also point out (see Fact 7) that there exists a weak agnostic CSQ learner for disjunctions with complexity $2^{\tilde{O}(n^{1/2})}$.

A conceptual implication is that there exists a qualitative difference between distribution-specific and distribution-free agnostic learning. In the distribution-specific setting, in particular when the underlying feature distribution is a discrete product distribution or the standard Gaussian, prior work Dachman-Soled et al. (2015); Diakonikolas et al. (2021) has shown that the L_1-polynomial regression algorithm is optimal in the SQ model—that is, CSQ and SQ are polynomially equivalent.

Finally, by building on the agnostic learner of Theorem 2, we also obtain an α-approximate agnostic learner, i.e., an algorithm with error guarantee of $\alpha \cdot \mathrm{OPT} + \epsilon$ for some $\alpha \geq 1$, with complexity $2^{\tilde{O}(n^{1/3}\alpha^{-2/3})}\mathrm{poly}(1/\epsilon)$ (see Theorem 10).

1.1. Technical Overview

In this section, we summarize the key ideas of our algorithms.

Sample-Based Agnostic Learner We start by describing our first agnostic learner with complexity $2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$. We start by recalling the L_1-polynomial regression algorithm Kalai et al. (2008a). In particular, the L_1 regression algorithm allows one to agnostically learn a function f to error $\mathrm{OPT} + \epsilon$ in time $n^{O(d)}$ if f can be ϵ-approximated, in L_∞-norm, by polynomials of degree d. The work of Kalai et al. (2008a) shows that, since disjunctions can be approximated by polynomials of degree $O(\sqrt{n})$ (see Nisan and Szegedy (1994); Paturi (1992)), then the L_1-algorithm is an agnostic learner for disjunctions with sample size and running time roughly $n^{O(\sqrt{n})}$. Since the $O(\sqrt{n})$-degree bound for polynomial approximation to the class of disjunctions is tight, this complexity upper bound is best possible for L_1-regression.

For the rest of this section, we will focus on the special case of agnostically learning monotone disjunctions (as the general task can be easily reduced to it, by adding new coordinates for the negations of each of the original coordinates). The starting point of our algorithm is the following observation: using the L_1-polynomial approximation approach, we can show that monotone disjunctions can be approximated on all Hamming-weight at most r strings—where r is a parameter to be optimized in hindsight—using polynomials of degree $O(\sqrt{r})$ (see Lemma 6). This means that if the underlying distribution D_X on the domain $\{0,1\}^n$ is largely supported on "low-weight" strings, we obtain a more efficient agnostic learning algorithm.

If this is not true (i.e., if D_X assigns significant probability mass to strings of high Hamming weight), we consider two separate cases:

1. If the optimal conjunction, f^*, has $f^*(\mathbf{x}) = 1$ on almost all of the high weight inputs, we can return a hypothesis that returns 1 on the high weight inputs and uses the L_1 regression algorithm to learn a nearly optimal classifier on the low weight inputs.

2. If the optimal conjunction, f^*, assigns $f^*(\mathbf{x}) = 0$ to a reasonable fraction of high weight inputs, our algorithm can guess a specific high weight input for which $f^*(\mathbf{x}) = 0$. If we guessed correctly, we know that none of the 1-entries in $\mathbf{x}$ can be in the support of f^*, allowing us to throw away these r coordinates and recurse on a substantially smaller problem.

The algorithm iteratively removes coordinates using Case 2 until it eventually ends up in Case 1. Note that we can only land in Case 2 at most n/r times before the input size becomes trivial.

Each reduction requires that we guess a positive, high-weight input, which—unless we are in Case 1— will constitute at least an ϵ-fraction of the probability mass of the high-weight inputs. Thus, the probability that we guess correctly enough times will be roughly $\epsilon^{n/r}$. In other words, if we attempt this scheme $(1/\epsilon)^{n/r}$ times, we expect that at least one attempt will succeed. This gives us a final algorithm with sample and computational complexity roughly $(1/\epsilon)^{n/r} n^{O(\sqrt{r})}$. Setting r to be roughly $n^{2/3}$ gives us a complexity upper bound of $2^{\tilde{O}(n^{1/3})}$.

SQ Agnostic Learner Note that the algorithm described above is not efficiently implementable in the SQ model as it requires picking out particular high-weight inputs. We next show that there exists an SQ algorithm that works along similar lines and requires comparable resources. We start by pointing out that the L_1-regression part of the previous algorithm is known to be implementable in the SQ model (see, e.g., Dachman-Soled et al. (2015)). We can thus use this to perform agnostic learning on low-weight inputs. To deal with the high-weight inputs, we instead consider *heavy* coordinates. A coordinate is termed *heavy* if it shows up in more than roughly an r/n-fraction of inputs. Observe that once we have reduced to only high-weight inputs, there must be some heavy coordinates. For these coordinates, we again have two cases:

1. None of the heavy coordinates are in the support of our target disjunction. In this case, we can remove all such coordinates from our domain, and we will be left with many low-weight inputs on which we can agnostically learn the function as before.

2. There is some heavy coordinate in the support of our target disjunction. In this case, if we guess such a coordinate, we learn an r/n-fraction of inputs on which the true function must be true.

Overall, by guessing that we are either in Case 1 or guessing the correct coordinate in Case 2 n/r times, we can (if we guess correctly) learn the value of the target function on a constant fraction of inputs. The probability of success of these guesses is roughly $1/n^{n/r}$, so we obtain complexity of roughly $n^{n/r}\, n^{O(\sqrt{r})}$. By selecting $r = n^{2/3}$, this gives a final complexity bound of roughly $2^{\tilde{O}(n^{1/3})}$. It is straightforward to check that this algorithm is implementable in the SQ model with comparable complexity.

Interestingly, although we have given an SQ algorithm with complexity $2^{\tilde{O}(n^{1/3})}$, this complexity bound is not possible for CSQ algorithms. In particular, it can be shown that any CSQ algorithm requires either roughly $2^{\Omega(n^{1/2})}$ correlational queries or queries of accuracy better than roughly $2^{n^{-1/2}}$. This follows by combining known results. In particular, Nisan and Szegedy (1994) showed that the approximation degree of the class of disjunctions is $\Omega(\sqrt{n})$, and a result of Gollakota et al. (2020) shows that this implies a $2^{\Omega(\sqrt{n})}$ CSQ lower bound. This is essentially because LP duality implies that high approximation degree allows one to construct a moment-matching construction, which when embedded among a random subset of coordinates, is hard for a CSQ algorithm to learn.

Approximate Agnostic Learner We also explore the complexity of agnostic learning with approximate error guarantees, obtaining a time-accuracy tradeoff. In particular, if we only require our algorithm to obtain accuracy $\alpha \cdot \mathrm{OPT}$, for some $\alpha > 1$, it is sufficient to obtain a weak learner that does slightly better than 50% when $\mathrm{OPT} < 1/\alpha$. This means that we will only need to guess correctly in Case 2 roughly $n/(r\alpha)$ times. Furthermore, as our algorithm only needs to succeed when $\mathrm{OPT} < 1/\alpha$, the L_1 regression algorithm only needs to consider polynomials of degree $O(\sqrt{r/\alpha})$. This gives a final runtime of roughly $n^{n/(r\alpha)} n^{O(\sqrt{r/\alpha})}$. Optimizing r to be $n^{1/3}\alpha^{-1/3}$, we get an algorithm with runtime roughly $2^{O(n^{1/3}\alpha^{-2/3})}$.

Organization After basic background in Section 2, in Section 3 we give our main algorithm for agnostically learning disjunctions. In Section 4, we give an alternative SQ agnostic learner with qualitatively the same complexity. Finally, Section 5, gives our approximate agnostic learner. Some of the proofs and technical details have been deferred to an Appendix.

2. Preliminaries

Notation We use $\{0,1\}$ for Boolean values. For $n \in \mathbb{N}$, we let $[n]\{i \in \mathbb{N} \mid i \leq n\}$. For a finite set S, we use $u(S)$ to denote the uniform distribution over all elements in S. For $\mathbf{x} \in \{0,1\}^n$, we use $W(\mathbf{x})$ to denote the Hamming weight of $\mathbf{x}$, defined as $W(\mathbf{x}) \sum_{i \in [n]} \mathbf{x}_i$. We define the Hamming weight of $\mathbf{x}$ on a subset of coordinates $I \subseteq [n]$ as $W_I(\mathbf{x}) \sum_{i \in I} \mathbf{x}_i$. A monotone disjunction is any Boolean function $f : \{0,1\}^n \to \{0,1\}$ of the form $f(\mathbf{x}) = \bigvee_{i \in S} \mathbf{x}_i$, where $S \subseteq [n]$ is the set of relevant variables.

Probability Basics We will need the following well-known fact about uniform convergence of empirical processes. We start by recalling the definition of VC dimension.

Definition 3 (VC-Dimension) *For a class $\mathcal{C}$ of Boolean functions $f : X \to \{0,1\}$, the VC-dimension of $\mathcal{C}$ is the largest d such that there exist d points $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_d \in X$ so that for any Boolean function $g : \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_d\} \to \{0,1\}$, there exists an $f \in \mathcal{C}$ satisfying $f(\mathbf{x}_i) = g(\mathbf{x}_i)$, for all $1 \leq i \leq d$.*

Then the VC inequality is the following:

Fact 1 (VC-Inequality) *Let $\mathcal{C}$ be a class of Boolean functions on X with VC-dimension d, and let D be a distribution on X. Let $\epsilon > 0$ and let n be an integer at least a sufficiently large constant multiple of d/ϵ^2. Then, if $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n$ are i.i.d. samples from D, we have that:*

$$\mathbf{Pr}\left[\sup_{f \in \mathcal{C}} \left| \frac{\sum_{j=1}^{n} f(\mathbf{x}_j)}{n} - \mathbf{E}_{\mathbf{x} \sim D}[f(\mathbf{x})] \right| \geq \epsilon \right] = \exp(-\Omega(n\epsilon^2)) \; .$$

Approximate Degree and L_1-Regression We will need the following definitions and facts about approximate degree and polynomial L_1-regression.

Definition 4 (Approximate degree) *Let $f : X \to \{0,1\}$ be a Boolean-valued function, where X is a finite subset of $\mathbb{R}^n$. The ϵ-approximate degree $\deg_{\epsilon,X}(f)$ of f on X, $0 < \epsilon < 1$, is the least degree of a polynomial $p : X \to \mathbb{R}$ such that $|f(\mathbf{x}) - p(\mathbf{x})| \leq \epsilon$ for all $\mathbf{x} \in X$. For a class $\mathcal{C}$ of Boolean functions, we define the ϵ-approximate degree of $\mathcal{C}$ on X as $\deg_{\epsilon,X}(\mathcal{C}) \max_{f \in \mathcal{C}} \deg_{\epsilon,X}(f)$.*

The main known technique for agnostic learning is the L_1 polynomial regression algorithm Kalai et al. (2008a). This algorithm uses linear programming to compute a low-degree polynomial that minimizes the L_1-distance to the target function. Its performance hinges on how well the underlying concept class $\mathcal{C}$ can be approximated, in L_1 norm, by a low-degree polynomial. In more detail, if d is the (minimum) degree such that any $f \in \mathcal{C}$ can be ϵ-approximated in L_1 norm by a degree-d polynomial, the algorithm has sample and computational complexity $n^{O(d)}/\mathrm{poly}(\epsilon)$, where ϵ is the excess error. It is also well-known that this algorithm can be implemented in the SQ model; see, e.g., Dachman-Soled et al. (2015) Our algorithm will use L_1-polynomial regression as a subroutine.

Fact 2 (Kalai et al. (2008a)) *Let $\mathcal{C}$ be a concept class of functions $f : X \to \{0,1\}$, where X is a finite subset of $\mathbb{R}^n$. There is a degree-$O(\deg_{\epsilon,X}(\mathcal{C}))$ polynomial L_1-regression algorithm that distribution-free agnostically learns $\mathcal{C}$ to additive error ϵ and has sample and computational complexity $n^{O(\deg_{\epsilon,X}(\mathcal{C}))}$. Furthermore, the L_1-regression algorithm can be implemented in the SQ model with the same complexity, i.e., having T time complexity and using q queries to $\mathrm{STAT}(\tau)$, where $\max(T, q, 1/\tau) = n^{O(\deg_{\epsilon,X}(\mathcal{C}))}$.*

3. Sample-based Agnostic Learner

In this section, we give an agnostic learner for disjunctions with complexity $2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$, thereby establishing Theorem 2. The agnostic learner presented here makes essential use of the samples. An SQ agnostic learner with similar complexity is given in the Section 4.

We start by pointing out two simplifications that can be made without loss of generality. First, it suffices to consider *monotone* disjunctions. As is well-known, one can easily and efficiently reduce the general task to the task of agnostically learning monotone disjunctions by including negated variables as additional features. Second, it suffices to develop a *weak* agnostic learner with the desired complexity. In our context, a weak agnostic learner is an algorithm whose output hypothesis performs slightly better than a random guess, when such a hypothesis in $\mathcal{C}$ exists. Given such an algorithm, we can leverage standard agnostic boosting techniques to obtain a strong agnostic learner, i.e., an algorithm with accuracy $\mathrm{OPT} + \epsilon$, with qualitatively the same complexity (up to a polynomial factor). Specifically, it suffices to establish the following result:

Theorem 5 (Weak Agnostic Learner for Monotone Disjunctions) *Let D be an unknown distribution supported on $\{0,1\}^n \times \{0,1\}$ and $\epsilon \in (0, 1/2)$. Suppose there is a monotone disjunction $f : \{0,1\}^n \to \{0,1\}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[f(\mathbf{x}) \neq y] \leq 1/2 - \epsilon$. Then there is an algorithm that given i.i.d. sample access to D and ϵ, has sample and computational complexity $2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$, and with probability at least $2^{-O(n^{1/3}\log(1/\epsilon))}$ returns a hypothesis $h : \{0,1\}^n \to \{0,1\}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq 1/2 - \Omega(\epsilon)$.*

Note that by repeating the algorithm of Theorem 5 $2^{O(n^{1/3}\log(1/\epsilon))}$ times and testing the empirical error of the output hypothesis each time, we get a weak agnostic learner that succeeds with at least constant probability. We show how to apply standard boosting tools, in order to obtain Theorem 2, in Appendix A.

In the body of this section, we proceed to describe our weak agnostic learner, thereby proving Theorem 5. Let $f_S(\mathbf{x}) = \bigvee_{i\in S} \mathbf{x}_i$ be an arbitrary monotone disjunction with optimal loss, i.e., $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y] = \mathrm{OPT}$. We set the radius parameter $rn^{2/3}$ and partition the domain into two sets: $X_{\mathrm{light}}\{\mathbf{x} \in \{0,1\}^n \mid W(\mathbf{x}) \leq r\}$, the set of of "low" Hamming weight strings; and $X_{\mathrm{heavy}}\{\mathbf{x} \in \{0,1\}^n \mid W(\mathbf{x}) > r\}$, the set of "high" Hamming weight strings.

Since points in X_{light} have Hamming weight at most r, one can leverage the properties of Chebyshev polynomials to construct ϵ-approximate polynomials for monotone disjunctions on X_{light} of degree $O(r^{1/2}\log(1/\epsilon)) = O(n^{1/3}\log(1/\epsilon))$. This is shown in the lemma below.

Lemma 6 (Approximate Degree on Hamming weight $\leq r$ Strings) *Let $X \subseteq \{0,1\}^n$, $I \subseteq [n]$, and $\mathcal{C}$ be the concept class containing all monotone disjunctions of the form $f_S(\mathbf{x}) = \bigvee_{i\in S} \mathbf{x}_i$ for*

$S \subseteq I$ and the constant function $f(\mathbf{x}) \equiv 1$. Let $r \max_{\mathbf{x} \in X} W_I(\mathbf{x})$. Then

$$\deg_{\epsilon,X}(\mathcal{C}) = \begin{cases} O(r^{1/2}(1-2\epsilon)^{1/2}), & \epsilon \in [1/4, 1/2) \; ; \\ O(r^{1/2}\log(1/\epsilon)), & \epsilon \in (0, 1/4) \; . \end{cases}$$

Proof First notice that the constant function $f(\mathbf{x}) \equiv 1$ can be approximated by a polynomial of degree-0 with 0 error. Let $f_S(\mathbf{x}) = \bigvee_{i \in S} \mathbf{x}_i$ be the target monotone disjunction to be approximated by a polynomial. Note that $f_S(\mathbf{x}) = 1$ if $W_S(\mathbf{x}) > 0$; and 0 otherwise. We leverage the following standard fact about Chebyshev polynomials (see, e.g., Cheney (1966); Klivans and Servedio (2001)).

Fact 3 *Let $T_d : \mathbb{R} \to \mathbb{R}$ be the degree-d Chebyshev polynomial. Then T_d satisfies the following:*

1. *$|T_d(t) \leq 1|$ for $|t| \leq 1$ with $T_d(1) = 1$; and*
2. *$T'_d(t) \geq d^2$ for $t > 1$ with $T'_d(1) = d^2$.*

For the case that $\epsilon \in [1/4, 1/2)$, we construct the approximate polynomial as follows. Firstly, we take the univariate polynomial $p_1(t) = T_d\left(\frac{r-t}{r-1}\right)$, where $d = \lceil 2r^{1/2}(1-2\epsilon)^{1/2} \rceil$. Notice that $p_1(0) \geq 1 + 4(1-2\epsilon)$ and $p_1(t) \in [-1, 1]$ for any $t \in [1, r]$ from Fact 3. We then rescale p_1 and take the Hamming weight of $\mathbf{x}$ on S as input. Namely, we define the multivariate polynomial $p(\mathbf{x}) = -(1-\epsilon)p_1(W_S(\mathbf{x}))/p_1(0) + 1$. Notice that for $W_S(\mathbf{x}) = 0$, $p(\mathbf{x}) = \epsilon$. For $W_S(\mathbf{x}) \in [1, r]$, we have

$$|(1-\epsilon)p_1(W_S(\mathbf{x}))/p_1(0)| \leq \frac{1-\epsilon}{1+4(1-2\epsilon)} \leq \frac{\epsilon + (1-2\epsilon)}{1+4(1-2\epsilon)} \leq \max(\epsilon, 1/4) = \epsilon \; .$$

Therefore, for $W_S(\mathbf{x}) \in [1, r]$, we get $p(\mathbf{x}) \in [1-\epsilon, 1+\epsilon]$. Furthermore, the degree of the polynomial p is $O(r^{1/2}(1-2\epsilon)^{1/2})$.

For the case that $\epsilon \in (0, 1/4)$, we construct the approximate polynomial p similarly. Firstly, we take the univariate polynomial $p_1(t) = T_d\left(\frac{r-t}{r-1}\right)/2$, where $d = \lceil 2r^{1/2} \rceil$. Notice that $p_1(0) \geq 1$ and $p_1(t) \in [-1/2, 1/2]$ for any $t \in [1, r]$ from Fact 3. Then we take $p_2(t) = p_1(t)^{c\log(1/\epsilon)}$, for a sufficiently large constant c, which has $p_2(0) \geq 1$ and $p_2(t) \in [-\epsilon, \epsilon]$ for any $t \in [1, r]$. The multivariate polynomial defined as $p(\mathbf{x}) = -p_2\left(W_S(\mathbf{x})\right)/p_2(0) + 1$ has the desired property. The degree of the polynomial p is $O(r^{1/2}\log 1/\epsilon)$. This completes the proof. ■

Given the definitions of X_{heavy} and X_{light}, since $\text{OPT} \leq 1/2 - \epsilon$, the target disjunction f_S is $\Omega(\epsilon)$-correlated with the labels on either X_{light} or X_{heavy}. That is, we have that either $\mathbf{E}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x})(2y-1)\mathbb{1}(\mathbf{x} \in X_{\text{light}})] = \Omega(\epsilon)$ or $\mathbf{E}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x})(2y-1)\mathbb{1}(\mathbf{x} \in X_{\text{heavy}})] = \Omega(\epsilon)$. Our algorithm proceeds by one of the following cases:

1. Suppose that f_S is $\Omega(\epsilon)$-correlated with the labels on X_{light}. Then, since all the points in X_{light} have Hamming weight at most $r = n^{2/3}$, the ϵ-approximate degree of monotone disjunctions on X_{light} is at most $O(r^{1/2}\log(1/\epsilon)) = O(n^{1/3}\log(1/\epsilon))$ by Lemma 6. Therefore, we can simply apply the standard L_1-polynomial regression with degree-$O(n^{1/3}\log(1/\epsilon))$ to get a hypothesis with error $1/2 - \Omega(\epsilon)$ (see Fact 2).

2. Suppose that f_S is $\Omega(\epsilon)$-correlated with the labels on X_{heavy} and the labels are not balanced on X_{heavy}, i.e., $|\mathbf{Pr}_{(\mathbf{x},y)\sim D}[y = 1 \mid \mathbf{x} \in X_{\text{heavy}}] - 1/2| \geq c\epsilon$, for some constant $c > 0$. In this case, the constant classifier $h(\mathbf{x}) \equiv 0$ or $h(\mathbf{x}) \equiv 1$ works as a weak agnostic learner on X_{heavy}. Then we can find some constant $c' \in \{0, 1\}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[y \neq c' \mid \mathbf{x} \in X_{\text{light}}] \leq 1/2$, and the hypothesis $h(\mathbf{x}) = 1$ if $\mathbf{x} \in X_{\text{heavy}}$; $h(\mathbf{x}) = c'$ otherwise, should suffice for our weak agnostic learner.

3. If neither of the above two cases hold, then we use rejection sampling to sample $\mathbf{x}_{\text{guess}} \sim D$ conditioned on $\mathbf{x}_{\text{guess}} \in X_{\text{heavy}}$. Since f_S has $1/2 - \Omega(\epsilon)$ error and the labels are approximately balanced in X_{heavy}, with probability $\Omega(\epsilon)$ we have that $f_S(\mathbf{x}_{\text{guess}}) = 0$. Suppose that we correctly guess an $\mathbf{x}_{\text{guess}}$ such that $f_S(\mathbf{x}_{\text{guess}}) = 0$. Then by removing any coordinate i such that $(\mathbf{x}_{\text{guess}})_i = 1$, we remove at least $r = n^{2/3}$ coordinates from future consideration, since these coordinates cannot be in S. Then the algorithm proceeds by recursing on the remaining coordinates.

Notice that if neither Item 1 nor Item 2 hold, then Item 3 must hold; therefore, the algorithm can always make progress. It is easy to see that the depth of the recursion is $O(n^{1/3})$ and the overall success probability is $2^{-\tilde{O}(n^{1/3}\log(1/\epsilon))}$.

A detailed pseudocode is given in Algorithm 1. We are now ready to prove Theorem 5.

Proof [Proof of Theorem 5] We first analyze the sample and computational complexity. Since the algorithm only uses samples in P, which contains $2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$ i.i.d. samples from D, the sample complexity of the algorithm is at most $2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$. Furthermore, the computational complexity of the algorithm is at most $T\text{poly}\left(|P|, d^{O(r^{1/2}\log(1/\epsilon))}\right) = 2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$.

We then prove the correctness of the algorithm. We first show that the algorithm, with probability at least $2^{-O(n^{1/3}\log(1/\epsilon))}$ returns a hypothesis $h : \{0, 1\}^n \to \{0, 1\}$ with error over the sample set P $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h(\mathbf{x}) \neq y] \leq 1/2 - \epsilon/10$. Since we only need to show that the algorithm succeeds with $2^{-O(n^{1/3}\log(1/\epsilon))}$ probability, we assume that for all the h_1 and h_2, the algorithm always chooses the $c' \in \{0, 1\}$ that minimize $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h_1(\mathbf{x}) \neq y]$ and $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h_2(\mathbf{x}) \neq y]$. Given the algorithm only runs for T iterations, this happens with probability at least $2^{-O(T)} = 2^{-O(n^{1/3})}$. Let $f_S(\mathbf{x}) = \bigvee_{i\in S} \mathbf{x}_i$ be an arbitrary optimal monotone disjunction on D. From the assumption of the algorithm, we have that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y] \leq 1/2 - \epsilon$. Then using Fact 1 and the fact that the VC-dimension of the disjunctions is n, we have that $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) \neq y] \leq 1/2 - \epsilon/2$. We first show the following structural lemma.

Lemma 7 *Suppose that $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) \neq y] \leq 1/2 - \epsilon/2$, then for any iteration of Algorithm 1, at least one of the following holds:*

1. $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[y = 1 \wedge \mathbf{x} \in X_{\text{heavy}}] - \mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[y = 0 \wedge \mathbf{x} \in X_{\text{heavy}}] \geq \epsilon/4$,
2. $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) = y \wedge \mathbf{x} \in X_{\text{light}}] - \mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) \neq y \wedge \mathbf{x} \in X_{\text{light}}] \geq \epsilon/4$, *or*
3. $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) = 0 \wedge \mathbf{x} \in X_{\text{heavy}}] \geq \epsilon/4$.

Proof Notice that $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) \neq y] \leq 1/2 - \epsilon/2$ implies that

$$\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) = y] - \mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) \neq y] \geq \epsilon\,.$$

Algorithm 1 Weak Agnostic Learning of Monotone Disjunctions

Input: $\epsilon \in (0, 1/2)$ and sample access to a distribution D of $(\mathbf{x}, y)$ supported on $\{0,1\}^n \times \{0,1\}$, such that there is a monotone disjunction $f_S(\mathbf{x}) = \bigvee_{i \in S} \mathbf{x}_i$ with $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y] \leq 1/2 - \epsilon$.

Output: With $2^{-O(n^{1/3}\log(1/\epsilon))}$ probability, a hypothesis $h : \{0,1\}^n \to \{0,1\}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq 1/2 - \epsilon/100$.

Set $r \leftarrow n^{2/3}$, $T \leftarrow \lceil n/r \rceil + 1$ and initialize $I_0 \leftarrow [n]$.
▷ I keeps track of the remaining coordinates for consideration.
Let P be a set of $2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$ i.i.d. samples from D (with sufficiently large implied constant).
for $t = \{0, \cdots, T\}$ **do**
    Define $X_{\text{light}}\{\mathbf{x} \in \{0,1\}^n \mid W_{I_t}(\mathbf{x}) \leq r\}$, $X_{\text{heavy}}\{\mathbf{x} \in \{0,1\}^n \mid W_{I_t}(\mathbf{x}) > r\}$ and partition P as $P_{\text{light}}\{(\mathbf{x}, y) \in P \mid \mathbf{x} \in X_{\text{light}}\}$ and $P_{\text{heavy}}\{(\mathbf{x}, y) \in P \mid \mathbf{x} \in X_{\text{light}}\}$.
    Apply L_1-regression on $u(P_{\text{light}})$ for degree-$O\left(r^{1/2}\log(1/\epsilon)\right)$, which succeeds with at. least constant probability
Let h_1' be the output hypothesis.
    Sample $c' \sim u(\{0,1\})$ and define the hypothesis $h_1 : \{0,1\}^n \to \{0,1\}$ as $h_1(\mathbf{x}) = h_1'(\mathbf{x})$ if $\mathbf{x} \in X_{\text{light}}$; and $h_1(\mathbf{x}) = c'$ otherwise.
    Sample $c' \sim u(\{0,1\})$ and define the hypothesis $h_2 : \{0,1\}^n \to \{0,1\}$ as $h_2(\mathbf{x}) = 1$ if $\mathbf{x} \in X_{\text{heavy}}$; and $h_2(\mathbf{x}) = c'$ otherwise.
    Let $\hat{\text{err}}_i$ be $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h_i(\mathbf{x}) \neq y]$ for $i \in \{1,2\}$.
    if $\hat{\text{err}}_i \leq 1/2 - \epsilon/10$ for any $i \in \{1,2\}$ **then**
        return the h_i that satisfies the above condition.
    end if
    Sample a random $(\mathbf{x}_{\text{guess}}, y) \sim u(P_{\text{heavy}})$.
    Update $I_{t+1} \leftarrow I_t \backslash \{i \in [n] \mid (\mathbf{x}_{\text{guess}})_i = 1\}$.
end for

Suppose that neither Item 1 nor Item 2 hold. Since Item 2 does not hold and the above inequality, we get

$$\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) = y \wedge \mathbf{x} \in X_{\text{heavy}}] - \mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) \neq y \wedge \mathbf{x} \in X_{\text{heavy}}] \geq (3/4)\epsilon \ .$$

Combining this and the fact that Item 1 does not hold, we get

$$\begin{aligned}
&\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) = 0 \wedge \mathbf{x} \in X_{\text{heavy}}] \geq \mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) = y \wedge y = 0 \wedge \mathbf{x} \in X_{\text{heavy}}] \\
\geq &\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) = y \wedge \mathbf{x} \in X_{\text{heavy}}] + \mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[y = 0 \wedge \mathbf{x} \in X_{\text{heavy}}] \\
&- \mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[\mathbf{x} \in X_{\text{heavy}}] \\
\geq &\frac{1}{2}\left(\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[\mathbf{x} \in X_{\text{heavy}}] + (3/4)\epsilon\right) + \frac{1}{2}\left(\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[\mathbf{x} \in X_{\text{heavy}}] - \epsilon/4\right) \\
&- \mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[\mathbf{x} \in X_{\text{heavy}}] \\
= &\epsilon/4 \ .
\end{aligned}$$

■

Notice that if the algorithm terminates before T iterations and returns a hypothesis, it must return an h such that $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h(\mathbf{x}) \neq y] \leq 1/2 - \epsilon/10$. Furthermore, suppose P_{heavy} is always nonempty except in the iteration it terminates. Then Line 13 must remove at least r coordinates from S in each iteration (from the definition of P_{heavy}), and the algorithm must terminate in $\lceil n/r \rceil \leq T$ iterations. Therefore, it suffices for us to show that P_{heavy} is always nonempty except in the iteration it terminates.

We first argue that, with at least $2^{-O(n^{1/3}\log(1/\epsilon))}$ probability, $S \subseteq I_t$ for all iterations. For iteration t, suppose that $S \subseteq I_t$ and the algorithm did not terminate in iteration t. Then the "if" condition in Line 9 is not satisfied for both h_1 and h_2. Since $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h_1(\mathbf{x}) \neq y] > 1/2 - \epsilon/10$, given that the algorithm always chooses the c' that minimizes $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h_1(\mathbf{x}) \neq y]$, we have that

$$\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h_1'(\mathbf{x}) = y \wedge \mathbf{x} \in X_{\text{light}}] - \mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h_1'(\mathbf{x}) \neq y \wedge \mathbf{x} \in X_{\text{light}}] \leq \epsilon/5 \ .$$

From Fact 2, Lemma 6 and $S \subseteq I_t$, we have that L_1 regression learns a hypothesis h_1' such that

$$\mathbf{Pr}_{(\mathbf{x},y)\sim u(P_{\text{light}})}[h_1'(\mathbf{x}) \neq y] \leq \mathbf{Pr}_{(\mathbf{x},y)\sim u(P_{\text{light}})}[f_S(\mathbf{x}) \neq y] + \epsilon/100 \ .$$

This implies that

$$\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) = y \wedge \mathbf{x} \in X_{\text{light}}] - \mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) \neq y \wedge \mathbf{x} \in X_{\text{light}}] < \epsilon/4 \ ,$$

and therefore Item 2 in Lemma 7 does not hold. Then similarly, since $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h_2(\mathbf{x}) \neq y] > 1/2 - \epsilon/10$, it follows that

$$\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[y = 1 \wedge \mathbf{x} \in X_{\text{heavy}}] - \mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[y = 0 \wedge \mathbf{x} \in X_{\text{heavy}}] < \epsilon/4 \ ,$$

and therefore Item 1 in Lemma 7 does not hold. Combining the above arguments and Lemma 7, we get that $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[f_S(\mathbf{x}) = 0 \wedge \mathbf{x} \in X_{\text{heavy}}] \geq \epsilon/4$ (this also implies that P_{heavy} is nonempty). Therefore, with probability at least $\Omega(\epsilon)$, the algorithm always samples an $\mathbf{x}_{\text{guess}}$ such

that $f_S(\mathbf{x}_{\text{guess}}) = 0$ in Line 13, and any coordinate i removed from I_t in Line 13 cannot be in S. Therefore, we have $S \subseteq I_{t+1}$ with probability at least $\Omega(\epsilon)$.

Suppose the algorithm runs for T' iterations. We argue that with at least $2^{-O(n^{1/3}\log(1/\epsilon))}$ probability, $S \subseteq I_t$ for all iteration $t \in [T']$. Notice that we initialized $I_0 = [n]$, which satisfies $S \subseteq I_0$. For iteration t, given that $S \subseteq I_t$, if the algorithm does not terminate, then with probability at least $\Omega(\epsilon)$, $S \subseteq I_{t+1}$ (as argued in the paragraph above). Since there are at most T iterations, with probability at least $\Omega(\epsilon)^T = 2^{-O(n^{1/3}\log(1/\epsilon))}$, we will have $S \subseteq I_t$ for all iterations $t \in [T']$. This also implies that P_{heavy} is always nonempty except in the last iteration (as we argued above). Therefore, the algorithm must terminate in at most T iterations and return a hypothesis h such that $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h(\mathbf{x}) \neq y] \leq 1/2 - \epsilon/10$ over the sample set P.

Now, it only remains for us to show that any hypothesis h that the algorithm returns such that $\mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h(\mathbf{x}) \neq y] \leq 1/2 - \epsilon/10$ must also satisfies $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq 1/2 - \epsilon/100$. To do so, notice that both h_1 and h_2 are always an intersection of two halfspaces. Therefore, the VC-dimension of the class of functions of all possible hypotheses the algorithm can return is $O(n)$. By Fact 1, we have that

$$\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq \mathbf{Pr}_{(\mathbf{x},y)\sim u(P)}[h(\mathbf{x}) \neq y] + \epsilon/100 \leq 1/2 - \epsilon/100 \,.$$

This completes the proof. ■

4. Statistical Query Agnostic Learner

Here we provide a Statistical Query version of the previous algorithm with qualitatively the same complexity. Interestingly, this algorithm can be implemented in the SQ model, but not in the CSQ model. Furthermore, it outperforms the CSQ lower bound of Gollakota et al. (2020) (see Appendix D), which implies a strong separation between the power of SQ and CSQ algorithms in the distribution-free agnostic model. For concreteness, we include the basics on the SQ model.

Statistical Query (SQ) Model The class of SQ algorithms Kearns (1998) is a family of algorithms that are allowed to query expectations of bounded functions on the underlying distribution through an (SQ) oracle rather than directly access samples.

Definition 8 (SQ Model) *Let D be a distribution on $X \times \{0,1\}$. A* statistical query *is a bounded function $q : X \times \{0,1\} \to [-1,1]$, and we define $\text{STAT}_D(\tau)$ to be the oracle that given any such query q, outputs a value $v \in [-1,1]$ such that $|v - \mathbf{E}_{(\mathbf{x},y)\sim D}[q(\mathbf{x},y)]| \leq \tau$, where $\tau > 0$ is the* tolerance *parameter of the query. An* SQ algorithm *is an algorithm whose objective is to learn some information about an unknown distribution D by making adaptive calls to the $\text{STAT}_D(\tau)$ oracle.*

Theorem 9 *Let D be an unknown joint distribution of $(\mathbf{x}, y)$ supported on $\{0,1\}^n \times \{0,1\}$ and $\epsilon \in (0,1)$. There is an algorithm that makes at most q queries to $\text{STAT}_D(\tau)$, has T computational complexity, and distribution-free agnostically learns disjunctions to additive error ϵ, where $\max(q, 1/\tau, T) = 2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$.*

It is worth noting that Theorem 9 morally corresponds to a sample-based algorithm that has $2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$ sample and computational complexity, as one can simulate the answers from the SQ oracle by empirical estimation of queries using fresh samples.

The high-level intuition of the algorithm is the following. As we have discussed in the previous section, it suffices to consider learning monotone disjunctions. Let $f_S(\mathbf{x}) = \bigvee_{i \in S} \mathbf{x}_i$ be any optimal monotone disjunction. We set the radius parameter $r = n^{2/3}$ and call a coordinate $i \in [n]$ heavy if $i \in S$ and $\mathbf{E}_{(\mathbf{x},y)\sim D}[\mathbb{1}(\mathbf{x}_i = 1)] \geq r/n$. Given an input distribution D of $(\mathbf{x}, y)$, we will either (a) guess that some $i \in [n]$ is a heavy coordinate; or (b) guess that there is no heavy coordinate. Suppose that with probability $1/2$ we sample $i \sim u([n])$ and guess that i is heavy, and with probability $1/2$ we guess that there is no heavy coordinate. Then this guess is always correct with probability $\Omega(1/n)$. Suppose that our guess is correct.

1. Suppose we guessed that i is heavy. Let $B = \{\mathbf{x} \in \{0,1\}^n \mid \mathbf{x}_i = 1\}$. Then we know that $f_S(\mathbf{x}) = 1$ for any $\mathbf{x} \in B$, and we can remove any $\mathbf{x} \in B$ from the input distribution for further consideration. By the definition of heavy coordinates, the probability mass removed is $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B] \geq r/n$.

2. Suppose we guessed that there is no heavy coordinate. Then we can remove any coordinate i such that $\mathbf{E}_{(\mathbf{x},y)\sim D}[\mathbb{1}(\mathbf{x}_i = 1)] \geq r/n$, since such a coordinate cannot be in S. Let I be the set of remaining coordinates. Then the expected Hamming weight on I satisfies $\mathbf{E}_{(\mathbf{x},y)\sim D}[W_I(\mathbf{x})] \leq r$. Let $B = \{\mathbf{x} \in \{0,1\}^n \mid W_I(\mathbf{x}) \leq 2r\}$. If we apply the degree-$O(r^{1/2}\log(1/\epsilon))$ L_1-regression algorithm on B, we will get a hypothesis h such that the error of h on B is at least as good as f_S on B. Therefore, we can just label every $\mathbf{x} \in B$ by $h(\mathbf{x})$ and remove B from the distribution for further consideration. Since $\mathbf{E}_{(\mathbf{x},y)\sim D}[W_I(\mathbf{x})] \leq r$ and $B = \{\mathbf{x} \in \{0,1\}^n \mid W_I(\mathbf{x}) \leq 2r\}$, by Markov's inequality, the mass removed is $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B] \geq 1/2$.

Now let D' be the new distribution of $(\mathbf{x}, y) \sim D$ conditioned on $\mathbf{x} \notin B$ with the irrelevant coordinates removed. We can repeat the above process on D'. Since each time we remove at least $r/n = n^{-1/3}$ fraction of the input distribution, we only need to guess correctly $n^{1/3}$ times. In the end, our output hypothesis will be a decision list combining the partial classifiers on all the sets B we removed.

The algorithm establishing Theorem 9 and the proof of its correctness are deferred to Appendix B.

5. Agnostic Learning with Approximate Error Guarantees

In the setup of Definition 1, given an approximation factor $\alpha \in (1, \infty)$ and additive error $\epsilon \in (0, 1)$, if an algorithm $\mathcal{A}$ outputs a hypothesis $h : X \to \{0, 1\}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq \alpha\,\mathrm{OPT} + \epsilon$, we will say that $\mathcal{A}$ (α, ϵ)-approximately agnostically learns $\mathcal{C}$. In this section, we give an SQ algorithm that provides a smooth trade-off between error and complexity. In particular, assuming $\alpha \in [32, \sqrt{n}]$ and $\epsilon \in (0, 1)$, there is an algorithm $\mathcal{A}$ that asks q queries to $\mathrm{STAT}_D(\tau)$, has computational complexity T, and (α, ϵ)-approximately agnostically learns disjunctions, where $\max(q, 1/\tau, T) = 2^{\tilde{O}(n^{1/3}\alpha^{-2/3})}\mathrm{poly}(1/\epsilon)$. Therefore, on one extreme point of the trade-off curve, we recover the guarantee of Algorithm 2 from Section 4 (the requirement $\alpha \geq 32$ here is only used for convenience of the algorithm description); and on the other extreme point we recover the guarantee of an earlier algorithm by Peleg (2007) (see also Awasthi et al. (2010)) that runs in polynomial time and outputs a hypothesis with error $O(n^{1/2})\mathrm{OPT} + \epsilon$.

We give the main theorem of this section.

Theorem 10 (Approximate Agnostic Learner) *Let D be an unknwon joint distribution of $(\mathbf{x}, y)$ supported on $\{0,1\}^n \times \{0,1\}$, $\alpha \in [32, \sqrt{n}]$ and $\epsilon \in (0,1)$. Then there is an algorithm that makes at most q queries to $\mathrm{STAT}_D(\tau)$, has T computational complexity, and (α, ϵ)-approximately agnostically learns disjunctions, where $\max(q, 1/\tau, T) = 2^{\tilde{O}(n^{1/3}\alpha^{-2/3})}\mathrm{poly}(1/\epsilon)$.*

Similar to how we described the high-level intuition of the algorithm in Section 3, we start by pointing out two simplifications that can be made without loss of generality. First, it suffices to consider *monotone* disjunctions, as discussed in the previous sections. Second, similar to Section 3, it suffices to develop a *weak* agnostic learner with the desired complexity. In the context of (α, ϵ)-approximate agnostic learning, a corresponding weak learner is an algorithm whose hypothesis performs slightly better than a random guess, when the input distribution satisfies $\mathrm{OPT} \leq 1/(2\alpha)$. Given such an algorithm, we can leverage standard agnostic boosting techniques to obtain our (α, ϵ)-approximate agnostic learner, i.e., an algorithm with accuracy $\alpha\mathrm{OPT}+\epsilon$, with qualitatively the same complexity (up to a polynomial factor).

Specifically, it suffices to establish the following result:

Theorem 11 (Weak Learner for Monotone Disjunctions given $\mathrm{OPT} \leq 1/\alpha$) *Let D be an unknown distribution supported on $\{0,1\}^n \times \{0,1\}$, $\alpha \in [64, \sqrt{n}]$ and $\epsilon \in (0,1)$. Suppose there is a monotone disjunction $f : \{0,1\}^n \to \{0,1\}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[f(\mathbf{x}) \neq y] \leq 1/\alpha$. Then there is an algorithm that makes at most q queries to $\mathrm{STAT}_D(\tau)$, has T computational complexity, and with probability at least p returns a hypothesis $h : \{0,1\}^n \to \{0,1\}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq 1/2 - 1/\mathrm{poly}(n)$, where $\max(q, 1/\tau, T, 1/p) = 2^{\tilde{O}(n^{1/3}\alpha^{-2/3})}$.*

The high-level idea of the weak learner is similar to Algorithm 2, with the main differences being that the degree of the L_1 regression is lower, and we only need to remove at most $O(1/\alpha)$ mass of the domain through guessing heavy coordinates. Let $f_S(\mathbf{x}) = \bigvee_{i \in S} \mathbf{x}_i$ be any optimal monotone disjunction as before. We set the radius parameter $r = n^{2/3}\alpha^{-1/3}$ and call a coordinate $i \in [n]$ heavy if $i \in S$ and $\mathbf{E}_{(\mathbf{x},y)\sim D}[\mathbb{1}(\mathbf{x}_i = 1)] \geq r/n$. Given an input distribution D of $(\mathbf{x}, y)$, we will either (a) guess that some $i \in [n]$ is a heavy coordinate, or (b) guess that there is no heavy coordinate. Suppose that with probability $1/2$ we sample $i \sim u([n])$ and guess that i is heavy, and with probability $1/2$ we guess that there is no heavy coordinate. Then this guess is always correct with probability $\Omega(1/n)$. Suppose that our guess is correct.

1. If the algorithm guesses (a), then we can obtain a partial classifier $h(\mathbf{x}) = 1$ for any $\mathbf{x}$ that has $\mathbf{x}_i = 1$ and remove these points from the domain. We can also remove coordinate i from further consideration, since all $\mathbf{x}$ remaining have $\mathbf{x}_i = 0$.

2. If the algorithm guesses (b), then we can discard any coordinate i such that $\mathbf{E}_D[\mathbf{x}_i] \geq r/n$ (as in Algorithm 2), since they cannot be in S by the definition of (b). After that, we get $\mathbf{E}_D[W_I(\mathbf{x})] \leq r$, where I is the set of coordinates remaining in consideration. By Markov's inequality, at least $1/2$ of the probability mass satisfies $W_I(\mathbf{x}) \leq 2r$. Since we assumed that $\mathrm{OPT} = O(1/\alpha)$, this implies that the error conditioned on this $1/2$ is still $O(1/\alpha)$. By Fact 2 and Lemma 6, applying L_1-regression with degree-$cr^{1/2}\alpha^{-1/2}$ (where c is a sufficiently large constant) allows us to learn within additive error $1/2 - 1/(c\alpha)$, which suffices for weak learning.

Notice that once the algorithm guesses case (b) (and is correct), then we immediately get a weak learner. However, if the algorithm guesses case (a), we will also be able to remove r/n mass and

obtain a partial classifier h that agrees with f_S on these mass. Since we assumed that $\mathrm{OPT} = O(1/\alpha)$, this can happen at most $\mathrm{OPT}/(r/n) = \mathrm{OPT}/(n^{-1/3}\alpha^{-1/3}) = n^{1/3}\alpha^{-2/3}$ times before we see a partial classifier that is non-trivially correlated with the labels on the mass we remove. This in turn gives a weak learner. Given the weak learner in Theorem 11, we can use standard boosting techniques Kalai et al. (2008b); Kalai and Kanade (2009); Feldman (2010) to get a (α, ϵ)-approximate agnostic learner. The algorithm and the proofs of Theorem 10 and Theorem 11 are deferred to Appendix C.

6. Conclusions and Open Problems

In this work, we give an $2^{\tilde{O}(n^{1/3})}$ time algorithm for agnostically learning disjunctions, substantially improving on the previous bound of $2^{\tilde{O}(n^{1/2})}$. As a corollary, we obtain the first super-polynomial separation between CSQ and SQ in the context of agnostic learning. The obvious open question is whether significantly faster agnostic learners for disjunctions exist. We note that any improvement on the complexity of our algorithm would also imply a similar improvement on the complexity of (realizable) PAC learning of DNFs, which would in turn improve upon the previous bound of Klivans and Servedio (2001). Finally, it is worth pointing out that even for the much broader class of linear threshold functions, the best known lower bounds (SQ and cryptographic) are only quasi-polynomial in n (for constant ϵ) (see Daniely (2016); Diakonikolas et al. (2022); Tiegel (2023)). Closing this large gap between known upper and lower bounds is a challenging direction for future work.

References

P. Awasthi, A. Blum, and O. Sheffet. Improved guarantees for agnostic learning of disjunctions. In *COLT 2010 - The 23rd Conference on Learning Theory*, pages 359–367, 2010.

A. Blum. Machine learning: My favorite results, directions, and open problems. In *44th Symposium on Foundations of Computer Science (FOCS 2003)*, pages 11–14, 2003.

N. Bshouty and V. Feldman. On using extended statistical queries to avoid membership queries. *Journal of Machine Learning Research*, 2:359–395, 2002.

M. Bun and J. Thaler. Approximate degree in classical and quantum computing. *Found. Trends Theor. Comput. Sci.*, 15(3-4):229–423, 2022.

E. Cheney. *Introduction to Approximation Theory*. McGraw-Hill, New York, New York, 1966.

D. Dachman-Soled, V. Feldman, L.-Y. Tan, A. Wan, and K. Wimmer. Approximate resilience, monotonicity, and the complexity of agnostic learning. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015*, pages 498–511. SIAM, 2015.

A. Daniely. Complexity theoretic limitations on learning halfspaces. In *Proceedings of the 48th Annual Symposium on Theory of Computing, STOC 2016*, pages 105–117, 2016.

I. Diakonikolas, D. M. Kane, T. Pittas, and N. Zarifis. The optimality of polynomial regression for agnostic learning under gaussian marginals. In *Proceedings of The 34th Conference on Learning Theory, COLT*, 2021.

I. Diakonikolas, D. M. Kane, P. Manurangsi, and L. Ren. Cryptographic hardness of learning halfspaces with massart noise. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 3624–3636, 2022. Available at https://doi.org/10.48550/arXiv.2207.14266.

V. Feldman. A complete characterization of statistical query learning with applications to evolvability. In *Proc. 50th Symposium on Foundations of Computer Science (FOCS)*, pages 375–384, 2009.

V. Feldman. Distribution-specific agnostic boosting. In *Proceedings of Innovations in Computer Science*, pages 241–250, 2010.

V. Feldman. Distribution-independent evolvability of linear threshold functions. In *COLT 2011 - The 24th Annual Conference on Learning Theory*, volume 19 of *JMLR Proceedings*, pages 253–272. JMLR.org, 2011.

V. Feldman, V. Guruswami, P. Raghavendra, and Y. Wu. Agnostic learning of monomials by halfspaces is hard. In *FOCS*, pages 385–394, 2009.

A. Gollakota, S. Karmalkar, and A. R. Klivans. The polynomial method is universal for distribution-free correlational SQ learning. *CoRR*, abs/2010.11925, 2020.

D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100:78–150, 1992.

A. Kalai and V. Kanade. Potential-based agnostic boosting. In *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009*, pages 880–888, 2009.

A. Kalai, A. Klivans, Y. Mansour, and R. Servedio. Agnostically learning halfspaces. *SIAM Journal on Computing*, 37(6):1777–1805, 2008a. Special issue for FOCS 2005.

A. Kalai, Y. Mansour, and E. Verbin. On agnostic boosting and parity learning. In *Proc. 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 629–638, 2008b.

M. Kearns, R. Schapire, and L. Sellie. Toward Efficient Agnostic Learning. *Machine Learning*, 17 (2/3):115–141, 1994.

M. J. Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM*, 45(6): 983–1006, 1998.

A. Klivans and R. Servedio. Learning DNF in time $2^{\tilde{O}(n^{1/3})}$. In *Proc. 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 258–265. ACM Press, 2001.

A. R. Klivans and A. A. Sherstov. Lower bounds for agnostic learning via approximate rank. *Comput. Complex.*, 19(4):581–604, 2010.

N. Nisan and M. Szegedy. On the degree of Boolean functions as real polynomials. *Comput. Complexity*, 4:301–313, 1994.

R. Paturi. On the degree of polynomials that approximate symmetric Boolean functions. In *Proceedings of the 24th Symposium on Theory of Computing*, pages 468–474, 1992.

D. Peleg. Approximation algorithms for the label-cover$_{\max}$ and red-blue set cover problems. *J. Discrete Algorithms*, 5(1):55–64, 2007.

A. Sherstov. The pattern matrix method for lower bounds on quantum communication. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 85–94, 2008.

S. Tiegel. Hardness of agnostically learning halfspaces from worst-case lattice problems. In *The Thirty Sixth Annual Conference on Learning Theory, COLT*, volume 195 of *Proceedings of Machine Learning Research*, pages 3029–3064, 2023.

L. G. Valiant. A theory of the learnable. In *Proc. 16th Annual ACM Symposium on Theory of Computing (STOC)*, pages 436–445. ACM Press, 1984.

Appendix

Appendix A. Omitted Proofs from Section 3

We formalize the notion of weak agnostic learning in the following definition.

Definition 12 ((α,γ)-weak learner) *Let $\mathcal{C}$ be a concept class of Boolean-valued functions $f : X \to \{0,1\}$. Given $\alpha, \gamma \in (0, 1/2)$ where $\alpha > \gamma$ and a distribution D on $X \times \{0,1\}$ such that $\mathrm{OPT} \min_{f\in\mathcal{C}} \mathbf{Pr}_{(\mathbf{x},y)\sim D}[f(\mathbf{x}) \neq y] \leq 1/2 - \alpha$, we call a hypothesis $h : X \to \{0,1\}$ a γ-weak hypothesis if $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq 1/2 - \gamma$. Given i.i.d. samples from D, the goal of the learning algorithm A is to output a γ-weak hypothesis with at least constant probability. We will say that the algorithm $\mathcal{A}$ distribution-free (α,γ)-weak agnostically learns $\mathcal{C}$.*

Given a distribution-free (α,γ)-weak agnostic learner for a concept class $\mathcal{C}$, it is possible to boost it to a learner that distribution-free agnostically learns $\mathcal{C}$ to additive error α as stated in the following fact (see Theorem 1.1 of Feldman (2010)). We remark that the results from Kalai et al. (2008b) and Kalai and Kanade (2009) would also suffice for the same purpose.

Fact 4 *There exists an algorithm ABoost that for every concept class $\mathcal{C}$, given a distribution-free (α,γ)-weak agnostic learning algorithm $\mathcal{A}$ for $\mathcal{C}$, distribution-free agnostically learns $\mathcal{C}$ to additive error α. Furthermore, ABoost invokes $\mathcal{A}$ $O(\gamma^{-2})$ times and runs in time $\mathrm{poly}(T, 1/\gamma)$, where T is the time and sample complexity of $\mathcal{A}$.*

Proof [Proof of Theorem 2] Given the above setup, a direct application of Fact 4 with $\alpha = \epsilon$ and $\gamma = \epsilon/100$, gives a learning algorithm for distribution-free agnostic learning monotone disjunctions to error $\mathrm{OPT} + \epsilon$ with sample and computational complexity

$$\mathrm{poly}\left(2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}, 1/\epsilon\right) = 2^{\tilde{O}\left(n^{1/3}\log(1/\epsilon)\right)}.$$

Since one can easily reduce learning general disjunctions (which includes negation of the variables) to learning monotone disjunctions by including negated variables as additional features, this completes the proof. ■

Appendix B. Omitted Proofs from Section 4

We give the detailed pseudocode in Algorithm 2.

We are now ready to prove the main theorem of this section. We restate Theorem 9 below for convenience.

Theorem 13 *Let D be an unknown joint distribution of $(\mathbf{x}, y)$ supported on $\{0,1\}^n \times \{0,1\}$ and $\epsilon \in (0,1)$. There is an algorithm that makes at most q queries to $\mathrm{STAT}_D(\tau)$, has T computational complexity, and distribution-free agnostically learns disjunctions to additive error ϵ, where $\max(q, 1/\tau, T) = 2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$.*

Proof [Proof of Theorem 9] For convenience of the analysis, let f_S be the same optimal hypothesis we fixed in the algorithm to maintain a consistent definition of heavy coordinates. We first prove the correctness of Algorithm 2, i.e., the algorithm will, with probability at least $2^{-\tilde{O}(n^{1/3}\log(1/\epsilon))}$,

Algorithm 2 Distribution-free Agnostic Learning Monotone Disjunctions to Additive Error (SQ)

Input: $\epsilon \in (0, 1/2)$ and SQ query access to a joint distribution D of $(\mathbf{x}, y)$ supported on $\{0,1\}^n \times \{0,1\}$.

Output: With at least $2^{-\tilde{O}(n^{1/3}\log(1/\epsilon))}$ probability, the algorithm outputs a hypothesis $h : \{0,1\}^n \to \{0,1\}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq \mathrm{OPT} + \epsilon$, where OPT is the error of the optimal monotone disjunction.

▷ For convenience of the algorithm description and analysis, we fix any optimal monotone disjunction $f_S(\mathbf{x}) = \bigvee_{i\in S} \mathbf{x}_i$ (unknown to the algorithm).

Let $r = n^{2/3}$, c be a sufficiently large constant, $T = cn\log(1/\epsilon)/r$ and initialize $U_0 \leftarrow \{0,1\}^n$ and $I_0 \leftarrow [n]$.

▷ U and I keep track of the remaining domain and coordinates.

for $t = \{0, \cdots, T\}$ **do**

  Define a coordinate i as heavy if $i \in S$ and $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbb{1}(\mathbf{x}_i = 1)|\mathbf{x} \in U_t] \geq r/n$.

  With probability $1/2$, sample an $i \sim u(I_t)$, guess that i is heavy and run REMOVEHEAVYCOORDINATE. With the remaining $1/2$ probability, guess that there is no heavy coordinate and run L_1-REGRESSIONONLIGHT.

  $U_{t+1} \leftarrow U_t \backslash B_t$.

  Let $\hat{P}_{U_{t+1}}$ be the answer of $\mathrm{STAT}_D(\epsilon/100)$ for the query function $q_i(\mathbf{x}, y) = \mathbb{1}(\mathbf{x} \in U_t)$.

  if $\hat{P}_{U_{t+1}} \leq \epsilon/3$ **then**

    Sample $c' \sim u(\{0,1\})$ and define $h : \{0,1\} \to \{0,1\}$ as $h(\mathbf{x}) = h'_k(\mathbf{x})$, where $k \in N$ is the smallest integer such that $\mathbf{x} \in B_k$, and $h(\mathbf{x}) = c'$ otherwise. Return h.

  end if

end for

---

procedure REMOVEHEAVYCOORDINATE

  Let $B_t = \{\mathbf{x} \in U_t | \mathbf{x}_i = 1\}$.

  Let $h'_t : B \to \{0,1\}$ be a partial classifier on B_t defined as $h'_t(\mathbf{x}) = 1$ if $\mathbf{x} \in B_t$.

  $I_{t+1} \leftarrow I_t \backslash \{i\}$.

end procedure

---

procedure L_1-REGRESSIONONLIGHT

  Let $\hat{P}_U$ and $\hat{P}_i$ (for all $i \in I_t$) be the answer of $\mathrm{STAT}_D(\epsilon r/(800n))$ for the query function $q_U(\mathbf{x}, y) = \mathbb{1}(\mathbf{x} \in U_t)$ and $q_i(\mathbf{x}, y) = \mathbb{1}(\mathbf{x}_i = 1 \wedge \mathbf{x} \in U_t)$ respectively.

  ▷ $\hat{P}_i/\hat{P}_U$ will be the empirical estimation of $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x}_i = 1 \mid \mathbf{x} \in U_t]$.

  $I_{t+1} \leftarrow I_t \backslash \{i \mid \hat{P}_i/\hat{P}_U \geq (1 + 1/100)r/n\}$ and $B_t = \{\mathbf{x} \in U_t | W_{I_{t+1}}(\mathbf{x}) \leq 2r\}$.

  Let D' be the joint distribution of $(\mathbf{x}, y) \sim D$ conditioned on $\mathbf{x} \in B_t$, which we have SQ query access to by asking queries on the distribution D.

  Apply the degree-$\left(cr^{1/2}\alpha^{-1/2}\right)$ polynomial L_1-regression algorithm (Fact 2) on D' and let h'_t be the output hypothesis.

end procedure

---

outputs a hypothesis h such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq \mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y] + \epsilon$. Notice that if there is any heavy coordinate in I, the algorithm will guess such a heavy coordinate with probability at least $1/(2n)$. If there is no heavy coordinate, with probability $1/2$, the algorithm will guess that there is no heavy coordinate. Therefore, the guess made by the algorithm is always correct with probability at least $1/(2n)$. Suppose the algorithm runs for T' iterations. Then, with probability at least $\Omega(1/n)^{T'} \geq \Omega(1/n)^{T} = 2^{-\tilde{O}(n^{1/3}\log(1/\epsilon))}$, all the guesses made by the algorithm are correct, and it suffices for us to show that if all the guesses are correct, then the algorithm outputs a hypothesis with error at most $\mathrm{OPT} + \epsilon$ deterministically. We first prove the following lemma on the partial classifiers h_t obtained in each iteration.

Lemma 14 *In Algorithm 2, given that all the guesses are correct, then for both* REMOVEHEAVYCOORDINATE *and* L_1-REGRESSIONONLIGHT *procedures, we have the following properties:*

1. $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_t(\mathbf{x}) \neq y \wedge \mathbf{x} \in B_t] - \mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y \wedge \mathbf{x} \in B_t] \leq \epsilon/(100T)$ *; and*
2. $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B_t | \mathbf{x} \in U_t] \geq n^{-1/3}$.

Proof We just need to show that the two properties hold for the output hypothesis of both REMOVEHEAVYCOORDINATE procedure and L_1-REGRESSIONONLIGHT procedure. It is easy to see that for the procedure REMOVEHEAVYCOORDINATE, both properties follow from the definition of heavy coordinate. So, it only remains to verify them for the L_1-REGRESSIONONLIGHT procedure. Notice that in the L_1-REGRESSIONONLIGHT procedure, we want to remove any coordinate i such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x}_i = 1 \mid \mathbf{x} \in U_t] \geq r/n$, since they cannot be in S. We do so by using $\hat{P}_i/\hat{P}_U$ as an estimate of $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x}_i = 1 \mid \mathbf{x} \in U_t]$. We will first need the following fact, which shows that $\hat{P}_i/\hat{P}_U$ is an accurate estimate.

Fact 5 *Let* $P_1, P_2, \hat{P}_1, \hat{P}_2 \in [0,1]$ *with* $P_1 \leq P_2$, $|P_1 - \hat{P}_1| \leq \tau$, $|P_2 - \hat{P}_2| \leq \tau$ *and* $\hat{P}_2 - \tau \geq \gamma > 0$. *Then we have* $|\hat{P}_1/\hat{P}_2 - P_1/P_2| \leq 2\tau/\gamma$.

Proof For the direction $\hat{P}_1/\hat{P}_2 - P_1/P_2 \geq 2\tau/\gamma$, we have

$$P_1/P_2 \leq \frac{\hat{P}_1 + \tau}{\hat{P}_2 - \tau} = \left(\frac{\hat{P}_1}{\hat{P}_2} + \frac{\tau}{\hat{P}_2}\right)\frac{\hat{P}_2}{\hat{P}_2 - \tau} \leq \frac{\hat{P}_1}{\hat{P}_2} + 2\frac{\tau}{\hat{P}_2 - \tau} = \frac{\hat{P}_1}{\hat{P}_2} + 2\tau/\gamma\ .$$

For the direction $\hat{P}_1/\hat{P}_2 - P_1/P_2 \leq 2\tau/\gamma$, we have

$$P_1/P_2 \geq \frac{\hat{P}_1 - \tau}{\hat{P}_2 + \tau} = \left(\frac{\hat{P}_1}{\hat{P}_2} - \frac{\tau}{\hat{P}_2}\right)\frac{\hat{P}_2}{\hat{P}_2 + \tau} \geq \frac{\hat{P}_1}{\hat{P}_2} - 2\frac{\tau}{\hat{P}_2 + \tau} = \frac{\hat{P}_1}{\hat{P}_2} - 2\tau/\gamma\ .$$

■

A direct application of Fact 5 implies that

$$\left|\hat{P}_i/\hat{P}_U - \mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x}_i = 1 \mid \mathbf{x} \in U_t]\right| \leq (1/100)r/n\ . \tag{1}$$

For Property 2, since we have removed any $i \in I_t$ such that $\hat{P}_i/\hat{P}_U \geq (1 + 1/100)r/n$, it follows from Equation 1 that for any remaining $i \in I_{t+1}$, $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x}_i = 1 \mid \mathbf{x} \in U_t] \leq (1 + 1/50)r/n$.

Therefore, $\mathbf{E}_{(\mathbf{x},y)\sim D}[W_I(\mathbf{x}) \mid \mathbf{x} \in U_t] = \sum_{i\in I} \mathbf{E}_{(\mathbf{x},y)\sim D}[\mathbb{1}(\mathbf{x}_i = 1) \mid \mathbf{x} \in U_t] \leq (4/3)r/n$. By Markov's inequality, we have $\mathbf{Pr}[\mathbf{x} \in B_t \mid \mathbf{x} \in U_t] \geq 1/3$.

For Property 1, we first show that $f_{S\cap I_{t+1}}(\mathbf{x}) = f_S(\mathbf{x})$ for any $\mathbf{x} \in U_t$, where $f_{S\cap I_{t+1}}(\mathbf{x}) = \bigvee_{i\in S\cap I_{t+1}} \mathbf{x}_i$. Notice that any coordinate i removed from I_t must have $\hat{P}_i/\hat{P}_U \geq n^{-1/3}+n^{-1/3}/100$. Therefore, by (1), any such coordinate i must satisfy $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x}_i = 1 \mid \mathbf{x} \in U] \geq n^{-1/3}$. Given that all the guesses are correct, any such coordinate i cannot be in S. Therefore, we have that any $i \in S\backslash I_{t+1}$ must be removed from I by the previous call to the REMOVEHEAVYCOORDINATE procedure. Since the REMOVEHEAVYCOORDINATE procedure removed any $\mathbf{x}$ such that $\mathbf{x}_i = 1$ from U, we must have that for any $i \in S\backslash I_{t+1}$ and $\mathbf{x} \in U_t$, $\mathbf{x}_i = 0$. Therefore, for any $\mathbf{x} \in U_t$,

$$f_{S\cap I_{t+1}}(\mathbf{x}) = \bigvee_{i\in S\cap I_{t+1}} \mathbf{x}_i = \bigvee_{i\in S\cap I_{t+1}} \mathbf{x}_i \vee \bigvee_{i\in S\backslash I_{t+1}} \mathbf{x}_i = \bigvee_{i\in S} \mathbf{x}_i = f_S(\mathbf{x}) \ .$$

Then from Lemma 6, we have the $\epsilon/(100T)$-approximate degree of disjunctions on B_t is

$$O(n^{1/3}\log(T/\epsilon)) = \tilde{O}(n^{1/3}\log(1/\epsilon)) \ .$$

Using Fact 2, we get that

$$\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_t(\mathbf{x}) \neq y \mid \mathbf{x} \in B_t] - \mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_{S\cap I_{t+1}}(\mathbf{x}) \neq y \mid \mathbf{x} \in B_t] \leq \epsilon/(100T) \ .$$

Since $f_{S\cap I_{t+1}}(\mathbf{x}) = f_S(\mathbf{x})$ for all $\mathbf{x} \in U_t$, we have

$$\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_t(\mathbf{x}) \neq y \mid \mathbf{x} \in B_t] - \mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y \mid \mathbf{x} \in B_t] \leq \epsilon/(100T) \ ,$$

which implies

$$\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_t(\mathbf{x}) \neq y \wedge \mathbf{x} \in B_t] - \mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y \wedge \mathbf{x} \in B_t] \leq \epsilon/(100T) \ .$$

This completes the proof. ■

Given Property 2 of Fact 14, since we are removing B_t from U_t in each iteration, we have that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in U_t]$ will decrease by a multiplicative factor of $1 - n^{-1/3}$ in each iteration. Therefore, after at most T iterations, we have $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in U_t] \leq \epsilon/4$ and the algorithm returns the hypothesis h and terminates via Line 7. Suppose the algorithm terminates at the T'th iteration. Then the error of the output hypothesis is

$$\begin{aligned}
\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq& \sum_{t=1}^{T'} \mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_t(\mathbf{x}) \neq y \wedge \mathbf{x} \in B_i] + \mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in U_{T'+1}] \\
\leq& \mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y] + T'(\epsilon/(100T)) + \epsilon/2 \\
\leq& \mathrm{OPT} + \epsilon \ .
\end{aligned}$$

It only remains to verify the query and computational complexity of Algorithm 2. Notice that the smallest tolerance of any query that the algorithm directly asks is at least $\epsilon r/(800n) = \epsilon n^{-1/3}/800$, and any query asked by the L_1-regression has tolerance at least $2^{-\tilde{O}(n^{1/3}\log(1/\epsilon))}$. Furthermore, the computational complexity of the algorithm is $Tn^{O(n^{1/3}\log(T/\epsilon))} = 2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$ and the total

number of queries the algorithm asks must be bounded by the same quantity. This completes the proof of the correctness of Algorithm 2.

Given that Algorithm 2 is correct, we can simply repeat Algorithm 2 for $N = 2^{\tilde{O}(n^{1/3}\log(1/\epsilon))}$ times with the error parameter set to $\epsilon/3$, and let $h_1, \cdots, h_N$ be the output hypotheses. Let $\hat{\mathrm{err}}_i$ be the answer of $\mathrm{STAT}_D(\epsilon/3)$ for the query function $q(\mathbf{x}, y) = \mathbb{1}(h_i(\mathbf{x}) \neq y)$, which estimates the error of each output hypothesis. Then simply output h_k, where $k = \mathrm{argmin}_k \hat{\mathrm{err}}_i$. The analysis here is straightforward. Since Algorithm 2 succeeds with probability at least $2^{-\tilde{O}(n^{1/3}\log(1/\epsilon))}$ and we repeat it for N times, with probability at least a constant, the algorithm succeeds at least once. Therefore, we must have $\hat{\mathrm{err}}_k \leq \mathrm{OPT} + 2\epsilon/3$. This implies that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_k(\mathbf{x}) \neq y] \leq \mathrm{OPT} + \epsilon$ from the definition of the SQ oracle. This gives us an SQ algorithm for distribution-free agnostic learning monotone disjunctions to additive error ϵ. To learn general disjunctions, as we have discussed in the previous section, one can easily reduce learning general disjunctions to learning monotone disjunctions by including negated variables as additional features. This completes the proof. ■

Appendix C. Omitted Proofs from Section 5

The algorithm establishing Theorem 11 is provided as Algorithm 3 below.

We now give the proof for Theorem 11.

C.1. Proof for Theorem 11

Proof [Proof for Theorem 11] The proof here is similar to that for Algorithm 2. For convenience of the analysis, fix f_S be the same optimal hypothesis we fixed in the algorithm to maintain a consistent definition of heavy coordinates. Then notice that as we have discussed before, the guess in Line 4 is always correct with $1/(2n)$ probability. With probability at least $2^{-\tilde{O}(n^{1/3}\alpha^{-2/3})}$, all the guesses made in Line 4 are correct, and it suffices for us to show that given all the guesses are correct, then the algorithm outputs a hypothesis with error at most $1/2 - 1/\mathrm{poly}(n)$ with at least constant probability. For the rest of the proof, we assume that all the guesses in Line 4 are correct.

We first show that the algorithm always succeeds with at least a constant probability if it terminates via Line 7. Since the "if" condition in Line 7 is satisfied, we get that with at least constant probability that

$$\begin{aligned}
\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] =& \mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y \wedge \mathbf{x} \in B_t] + \mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y \wedge \mathbf{x} \notin B_t] \\
\leq& \frac{1}{2}\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B_t] - \Omega(r/n) + \frac{1}{2}\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \notin B_t] \\
\leq& 1/2 - 1/\mathrm{poly}(n) \ ,
\end{aligned}$$

where the second from the last inequality follows from that we sampled $c' \sim u(\{0,1\})$.

Given the statement in the above paragraph, it suffices for us to prove that the algorithm will terminate via Line 7 deterministically if all guesses in Line 4 are correct. We first give the following lemma, which is an analog of Lemma 14.

Lemma 15 *In Algorithm 3, given all the guesses in Line 4 are correct, and suppose that the algorithm did not terminate via Line 8 in the first t iterations. Then for all the first t iterations, the algorithm must have guessed that there is a heavy coordinate and $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in U_{t+1}] \geq 1/2$.*

Algorithm 3 Trade-off Algorithm for Distribution-free Agnostic Learning Monotone Disjunctions (Weak Learner)

Input: $\alpha \in [64, \sqrt{n}]$ and SQ query access to a joint distribution D of $(\mathbf{x}, y)$ supported on $\{0,1\}^n \times \{0,1\}$, where the error of the optimal monotone disjunction $\mathrm{OPT} \leq 1/\alpha$.

Output: With probability at least $2^{-\tilde{O}(n^{1/3}\alpha^{-2/3})}$, the algorithm outputs a hypothesis $h : \{0,1\}^n \to \{0,1\}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq 1/2 - 1/\mathrm{poly}(n)$.

▷ For convenience of the algorithm description and analysis, we fix any optimal monotone disjunction $f_S(\mathbf{x}) = \bigvee_{i \in S} \mathbf{x}_i$, which is unknown to the algorithm.

Set $r \leftarrow n^{2/3}\alpha^{-1/3}$, $T \leftarrow cn/(\alpha r)$, where c is a sufficiently large constant and initialize $U_0 \leftarrow \{0,1\}^n$ and $I_0 \leftarrow [n]$.

▷ U and I keep track of the remaining domain and coordinates.

for $t = \{0, \cdots, T\}$ **do**

  Define a coordinate i as heavy if $i \in S$ and $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbb{1}(\mathbf{x}_i = 1) | \mathbf{x} \in U_t] \geq r/n$.

  With probability $1/2$, sample an $i \sim u(I_t)$, guess that i is heavy and run REMOVEHEAVYCOORDINATE. With the remaining $1/2$ probability, guess that there is no heavy coordinate and run L_1-REGRESSIONONLIGHT.

  $U_{t+1} \leftarrow U_t \backslash B_t$.

  Let $\hat{P}$ be the answer of $\mathrm{STAT}_D(r/(100n))$ for the query function

$$q(\mathbf{x}, y) = \mathbb{1}(\mathbf{x} \in B_t \wedge h_t(\mathbf{x}) = y) - \mathbb{1}(\mathbf{x} \in B_t \wedge h_t(\mathbf{x}) \neq y) \, .$$

  if $\hat{P}_U \geq r/(4n)$ **then**

    Sample $c' \sim u(\{0,1\})$ and define $h : \{0,1\} \to \{0,1\}$ as $h(\mathbf{x}) = h'_t(\mathbf{x})$ if $\mathbf{x} \in B_t$ and $h(\mathbf{x}) = c'$ otherwise. Return h.

  end if

end for

---

procedure REMOVEHEAVYCOORDINATE

  $I_{t+1} \leftarrow I_t \backslash \{i\}$

  Let $B_t = \{\mathbf{x} \in U_t | \mathbf{x}_i = 1\}$ and $h'_t : B_t \to \{0,1\}$ be a partial classifier on B_t defined as $h'_t(\mathbf{x}) = 1$ if $\mathbf{x} \in B_t$.

end procedure

---

procedure L_1-REGRESSIONONLIGHT

  Let $\hat{P}_U$ and $\hat{P}_i$ (for all $i \in I$) be the answer of $\mathrm{STAT}_D(r/(800n))$ for the query function $q_U(\mathbf{x}, y) = \mathbb{1}(\mathbf{x} \in U_t)$ and $q_i(\mathbf{x}, y) = \mathbb{1}(\mathbf{x}_i = 1 \wedge \mathbf{x} \in U_t)$ respectively. ▷ $\hat{P}_i/\hat{P}_U$ will be the empirical estimate of $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x}_i = 1 \mid \mathbf{x} \in U_t]$.

  $I_{t+1} \leftarrow I_t \backslash \{i \mid \hat{P}_i/\hat{P}_U \geq (1 + 1/100)r/n\}$.

  Let $B_t = \{\mathbf{x} \in U_t | W_{I_{t+1}}(\mathbf{x}) \leq 2r\}$ and D' be the joint distribution of $(\mathbf{x}, y) \sim D$ conditioned on $\mathbf{x} \in B_t$, which we have SQ query access to by asking queries on the distribution D.

  Apply the degree-$\left(cr^{1/2}\alpha^{-1/2}\right)$ polynomial L_1-regression algorithm in Fact 2 on D' to learn a hypothesis h'_t.

▷ The degree of the L_1 regression is lower compared with Algorithm 2 due to different r.

end procedure

Proof We will prove the statement by induction. The statement is trivially true for $t = -1$. Given the statement is true for any $i \leq t-1$, we will prove that the statement is true for t. Suppose the algorithm did not terminate in the first t iterations, then since the statement is true for $i \leq t-1$, we have that all the first $t-1$ iterations must all guess that there is a heavy coordinate and $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in U_i] \geq 1/2$ for all $i \leq t$.

We first show that the t-th iteration also guesses that there is a heavy coordinate. By Fact 5, we have that $|\hat{P}_i/\hat{P}_U - \mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x}_i = 1 \mid \mathbf{x} \in U_t]| \leq (1/100)(r/n)$, therefore,

$$\mathbf{E}_{(\mathbf{x},y)\sim D}[W_{I_{t+1}}(\mathbf{x}) \mid \mathbf{x} \in U_t] \leq (4/3)r \ .$$

Then from $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in U_t] \geq 1/2$, the definition of B_t and Markov's inequality, we get $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B_t] \geq 1/6$. Assume for the purpose of contradiction that the t-th iteration guesses that there is no heavy coordinate. Then since $\mathrm{OPT} \leq 1/\alpha$, we have $\mathbf{Pr}_{(\mathbf{x},y)\sim D'}[f_S(\mathbf{x}) \neq y] \leq 6/\alpha < 1/8$. Notice that any $i \notin I_{t+1}$ must be removed by REMOVEHEAVYCOORDINATE of previous iterations since all previous iterations guess that there is a heavy coordinate. Furthermore, since REMOVEHEAVYCOORDINATE also removes any $\mathbf{x}$ such that $\mathbf{x}_i = 1$ from U, we must have that for any $\mathbf{x} \in U_t$ and $i \notin I_{t+1}$, $\mathbf{x}_i = 0$. This implies that for any $\mathbf{x} \in B_t$, $W_S(\mathbf{x}) \leq W_{I_{t+1}}(\mathbf{x}) \leq 2r$. Then from Lemma 6 and Fact 2, we have that the L_1-regression in L_1-REGRESSIONONLIGHT will learn f_S to additive error $1/2 - 7/\alpha$ on D'. This implies that

$$\begin{aligned}\mathbf{Pr}_{(\mathbf{x},y)\sim D'}[h'_t(\mathbf{x}) \neq y] \leq& \mathbf{Pr}_{(\mathbf{x},y)\sim D'}[f_S(\mathbf{x}) \neq y] + (1/2 - 7/\alpha) \\ \leq& 6/\alpha + 1/2 - 7/\alpha \leq 1/2 - 1/\alpha \ .\end{aligned}$$

Therefore, we have $\mathbf{E}_{(\mathbf{x},y)\sim D}[q(\mathbf{x},y)] \geq 2\mathbf{E}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B_t]/\alpha \geq 1/(3\alpha)$ where q is the query function in Line 6, and $\hat{P}$ must satisfy the "if" condition in Line 7. Then the algorithm will terminate in iteration t. This contradicts the assumption, and therefore, in the t-th iteration, the algorithm must also guess that there is a heavy coordinate.

Now it only remains to show $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in U_{t+1}] \geq 1/2$. Given the algorithm guesses that there is a heavy coordinate for the first t iterations, we assume for the purpose of contradiction that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in U_{t+1}] < 1/2$. Then using the fact that the "if" condition in Line 7 is never satisfied for the first t iterations, we have that

$$\begin{aligned}&\mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y] \\ \geq& \sum_{i\in[t]} \mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y \wedge \mathbf{x} \in B_i] \\ =& \sum_{i\in[t]} \mathbf{Pr}_{(\mathbf{x},y)\sim D}[h'_t(\mathbf{x}) \neq y \wedge \mathbf{x} \in B_i] \\ =& \sum_{i\in[t]} \mathbf{Pr}_{(\mathbf{x},y)\sim D}[h'_t(\mathbf{x}) \neq y | \mathbf{x} \in B_i]\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B_i] \\ \geq& \min_i(\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h'_t(\mathbf{x}) \neq y | \mathbf{x} \in B_i]) \sum_{i\in[t]} \mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B_i] \\ \geq& \min_i(\mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y | \mathbf{x} \in B_i])/2 \ .\end{aligned}$$

Notice that from the definition of heavy coordinates, we have

$$\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B_i] \geq \mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in U_i](r/n) = (1/2)(r/n) \ .$$

Then,

$$\begin{aligned}
&\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_t'(\mathbf{x}) \neq y \mid \mathbf{x} \in B_i] \\
=&\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_t'(\mathbf{x}) \neq y \wedge \mathbf{x} \in B_i]/\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B_i] \\
=&\frac{\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B_i] - (\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_t'(\mathbf{x}) = y \wedge \mathbf{x} \in B_i] - \mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_t'(\mathbf{x}) \neq y \wedge \mathbf{x} \in B_i])}{2\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B_i]} \\
\geq&1/2 - \frac{(1/4)(r/n) + (1/100)(r/n)}{r/n} \geq 1/5 \,.
\end{aligned}$$

Therefore, plugging it back gives $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y] \geq 1/10$. This contradicts the assumption that $\mathrm{OPT} \leq 1/\alpha \leq 1/64$ and therefore, we must have $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in U_{t+1}] \geq 1/2$. This completes the proof. ■

Fact 15 implies that once the algorithm guesses that there is no heavy coordinate, the algorithm must terminate via Line 7. Therefore, we only need to show that the algorithm cannot keep guessing that there is a heavy coordinate for T iterations. Notice that since all guesses are correct and all guesses are that there is a heavy coordinate, from the definition of heavy coordinates, we have $h_i(\mathbf{x}) = f_S(\mathbf{x})$ for any $\mathbf{x} \in B_i$ for all iteration i. Given the algorithm does not terminate for T iterations and the "if" condition in Line 7 is never satisfied for these T iterations, we would have

$$\begin{aligned}
\mathbf{Pr}_{(\mathbf{x},y)\sim D}[f_S(\mathbf{x}) \neq y] \geq& \sum_{i=1}^{T} \mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_i(\mathbf{x}) \neq y \wedge \mathbf{x} \in B_i] \\
\geq& \sum_{i=1}^{T} (\mathbf{Pr}_{(\mathbf{x},y)\sim D}[\mathbf{x} \in B_i] \\
&- (\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_i(\mathbf{x}) = y \wedge \mathbf{x} \in B_i] - \mathbf{Pr}_{(\mathbf{x},y)\sim D}[h_i(\mathbf{x}) \neq y \wedge \mathbf{x} \in B_i])) \\
\geq& \sum_{i=1}^{T} (r/(2n) - r/(3n)) = \Omega\left(Tr/n\right) = c/\alpha \,,
\end{aligned}$$

where c is a sufficiently large constant. This contradicts the assumption that $\mathrm{OPT} \leq 1/\alpha$. This completes the proof that given all the guesses in Line 4 are correct, Algorithm 3 will, with at least constant probability, outputs a hypothesis h such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq 1/2 - 1/\mathrm{poly}(n)$.

It only remains to verify the query and computational complexity. Notice that the smallest query tolerance that the algorithm directly asked is at least $1/\mathrm{poly}(n)$, and the smallest query tolerance asked by the L_1-regression is at least $d^{-cr^{1/2}\alpha^{-1/2}} = 2^{-\tilde{O}(n^{1/3}\alpha^{-2/3})}$. Furthermore, the computational complexity of the algorithm is $Tn^{O(r^{1/2}\alpha^{-1/2})} = 2^{\tilde{O}(n^{1/3}\alpha^{-2/3})}$, and the total number of queries the algorithm asks must be bounded by the same quantity. This completes the proof for Algorithm 3. ■

C.2. Proof of Theorem 10

Given Theorem 11, we are now ready to prove Theorem 10. We will first need the following fact about boosting for agnostic learning to multiplicative error from Feldman (2010).

Fact 6 *There exists an algorithm ABoostDI that for every concept class C over X, given a distribution independent (α, γ)-weak agnostic learning algorithm A' for C, for every distribution $A = (D, f)$ over X and $\epsilon > 0$, produces a hypothesis h such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[f(\mathbf{x}) \neq y] \leq \mathrm{OPT}/(1 - 2\alpha) + \epsilon$. Further, ABoostDI invokes A' $O(\gamma^{-2}\Delta^{-1}\log(1/\Delta))$ times for $\Delta = \mathrm{OPT}/(1-2\alpha)$ and runs in time $\mathrm{poly}(T, 1/\gamma, 1/\epsilon)$, where T is the running times of A'.*

We then prove Theorem 10.

Proof [Proof of Theorem 10] Given Theorem 11, to (α', ϵ)-approximate agnostically learn disjunctions, we first, without loss of generality, assume that $\alpha'\mathrm{OPT} \geq \epsilon$ (if this is not true, take a new $\epsilon' = \epsilon/4$ and α'' such that $\epsilon/4 \leq \alpha''\mathrm{OPT} \leq \epsilon/2$).

Then we can simply apply Fact 6 and take the α in Fact 6 as $(1/2 - \alpha'/2)$ and the boosting algorithm ABoostDI invokes Algorithm 3 at most $\mathrm{poly}(n/\epsilon)$ times and its own runtime is $\mathrm{poly}(T, 1/\gamma, 1/\epsilon) = 2^{\tilde{O}(n^{1/3}\alpha^{-2/3})}\mathrm{poly}(1/\epsilon)$. This gives the algorithm for agnostic learning monotone disjunctions to error $\alpha'\mathrm{OPT} + \epsilon$. Notice that this algorithm is still SQ since the boosting algorithm does not require sample access to the distribution. As we have argued before, one can easily reduce learning general disjunctions (which includes negation of the variables) to learning monotone disjunctions by including negated variables as additional features. This completes the proof. ■

Appendix D. CSQ Complexity of Distribution-free Agnostic Learning Disjunctions

In this section, we characterize the complexity of distribution-free agnostic learning disjunctions in the *Correlational Statistical Query* (CSQ) model.

Basics on CSQ Model In the context of Definition 8, a *Correlational Statistical Query* is a bounded function $q : X \to [-1, 1]$. We define $\mathrm{CSTAT}(\tau)$ to be the oracle that, given any such query q, outputs a value $v \in [-1, 1]$ such that $|v - \mathbf{E}_{(\mathbf{x},y)\sim D}[(2y-1)q(\mathbf{x})]| \leq \tau$, where $\tau > 0$ is the *tolerance* parameter of the query. A *Correlational Statistical Query (CSQ)) algorithm* is an algorithm whose objective is to learn some information about an unknown distribution D by making adaptive calls to the corresponding $\mathrm{CSTAT}(\tau)$ oracle. The query complexity of a CSQ algorithm is defined as m/τ^2, where m is the number of queries and τ is the smallest tolerance of queries the algorithm calls to the corresponding $\mathrm{CSTAT}(\tau)$ oracle.

It is well-known that CSQ queries are a special case of SQ queries, and, therefore, have weaker power. In particular, any SQ query function $q_{\mathrm{sq}} : X \times \{0, 1\} \to [-1, 1]$ can always be decomposed to $q_{\mathrm{sq}}(\mathbf{x}, y) = q_1(\mathbf{x}, y) + q_2(\mathbf{x}, y)$, where $q_1(\mathbf{x}, y) = (q_{\mathrm{sq}}(\mathbf{x}, 0) + q_{\mathrm{sq}}(\mathbf{x}, 1))/2$ is a query function independent of the label y, and $q_2(\mathbf{x}, y) = (2y - 1)(-q_{\mathrm{sq}}(\mathbf{x}, 0) + q_{\mathrm{sq}}(\mathbf{x}, 1))/2$ is equivalent to a CSQ query. An intuitive interpretation is that, compared with the SQ model, the CSQ model loses exactly the power to make label-independent queries about the distribution, i.e., the power to ask queries about the marginal distribution of $\mathbf{x}$.

CSQ Upper Bound on Weak Agnostic Learning We note that there is a CSQ weak agnostic learner with $2^{\tilde{O}(n^{1/2}\log(1/\epsilon))}$ time and query complexity that outputs a hypothesis with error $1/2 - \Omega(\epsilon)$ given that $\mathrm{OPT} \leq 1/2 - \epsilon$.

Fact 7 *Let D be an unknown distribution supported on $\{0, 1\}^n \times \{0, 1\}$ and $\epsilon \in (0, 1/2)$. Suppose there is a monotone disjunction $f : \{0, 1\}^n \to \{0, 1\}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[f(\mathbf{x}) \neq y] \leq 1/2 - \epsilon$.*

There is an algorithm that makes at most q queries to $\mathrm{CSTAT}_D(\tau)$, and deterministically returns a hypothesis $h : \{0,1\}^n \to \{0,1\}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[h(\mathbf{x}) \neq y] \leq 1/2 - \Omega(\epsilon)$, where $\max(q, 1/\tau) = 2^{\tilde{O}(n^{1/2}\log(1/\epsilon))}$.

Proof For convenience of the proof, we will use $\{-1,1\}$ for Boolean values instead of $\{0,1\}$ for this proof. We start by proving the following fact.

Fact 8 *Let D be an unknown distribution supported on $\{-1,1\}^n \times \{-1,1\}$ and $\epsilon \in (0,1/2)$. Suppose there is a monotone disjunction $f : \{-1,1\}^n \to \{-1,1\}$ such that $\mathbf{Pr}_{(\mathbf{x},y)\sim D}[f(\mathbf{x}) \neq y] \leq 1/2 - \epsilon$. Then there is a polynomial p of degree at most $O(n^{1/2}\log(1/\epsilon))$ such that for all $\mathbf{x} \in \{-1,1\}^n$, $p(\mathbf{x}) \in [-1,1]$ and $\mathbf{E}_{(\mathbf{x},y)\sim D}[yp(\mathbf{x})] = \Omega(\epsilon)$.*

Proof The proof here directly follows from Lemma 6. Let p' be the degree-$O(n^{1/2}\log(1/\epsilon))$ polynomial that is a $c\epsilon$-approximate (for sufficiently small constant c) polynomial of f in Lemma 6 with $r = n$. Then let p be defined as $p(\mathbf{x}) = p'(\mathbf{x})/(1+\epsilon)$, which is a $2c\epsilon$-approximate polynomial of f. It is easy to see that for all $\mathbf{x} \in \{-1,1\}^n$, $p(\mathbf{x}) \in [-1,1]$ from its definition. For the correlation, we have

$$\mathbf{E}_{(\mathbf{x},y)\sim D}[yp(\mathbf{x})] \geq (1-2c\epsilon)\mathbf{E}_{(\mathbf{x},y)\sim D}[y = f(\mathbf{x})] - \mathbf{E}_{(\mathbf{x},y)\sim D}[y \neq f(\mathbf{x})] = \Omega(\epsilon) \ .$$

This completes the proof. ■

We then show that it is always possible to find such a polynomial in Fact 8 using CSQ queries with query complexity at most $2^{\tilde{O}(n^{1/2}\log(1/\epsilon))}$. We define a parity function over the set $S \subseteq [n]$ as $g_S(\mathbf{x}) = \bigoplus_{i\in S} \mathbf{x}_i$, where $\bigoplus$ is the exclusive or operator. Notice that parity functions over sets of size at most d spans any polynomials p of degree at most d over $\{-1,1\}^n$, i.e., $p(\mathbf{x}) = \sum_{\{S\subseteq[n]\wedge|S|\leq d\}} \alpha_S g_S(\mathbf{x})$ for some $\alpha_S \in \mathbb{R}$. Furthermore, since parities form an orthonormal basis on $u(\{-1,1\})^n$, the parameters α_S satisfy $|\alpha_S| = |\mathbf{E}_{\mathbf{x}\sim u(\{-1,1\}^n)}[p(\mathbf{x})g_S(\mathbf{x})]| \leq 1$. Notice that

$$\mathbf{E}_{(\mathbf{x},y)\sim D}[yp(\mathbf{x})] = \sum_{S\subseteq[n]\wedge|S|\leq d} \alpha_S \mathbf{E}_{(\mathbf{x},y)\sim D}[yg_S(\mathbf{x})].$$

Therefore, if we already know the value of $\mathbf{E}_{(\mathbf{x},y)\sim D}[yg_S(\mathbf{x})]$ (up to $2^{-\tilde{O}(n^{1/2}\log(1/\epsilon))}$ error) from the CSQ oracle, we can approximately calculate the value of $\mathbf{E}_{(\mathbf{x},y)\sim D}[yp(\mathbf{x})]$ (up to $o(\epsilon)$ error) with no additional queries. This suffices for us to find such a p in Fact 8. Namely, let $\hat{p}_S$ be the answer of the CSQ oracle for the parity function g_S with error tolerance $2^{-\tilde{O}(n^{1/2}\log(1/\epsilon))}$ (with sufficiently large implied constant), i.e.,

$$|\hat{p}_S - \mathbf{E}_{(\mathbf{x},y)\sim D}[yg_S(\mathbf{x})]| \leq \tau \ ,$$

where $\tau = 2^{-c(n^{1/2}\log(1/\epsilon))\log(n^{1/2}\log(1/\epsilon))^c}$ and c is a sufficiently large constant. Then consider the following LP for finding the polynomial $p(\mathbf{x}) = \sum_{S\subseteq[n]\wedge|S|\leq d} \alpha_S g_S(\mathbf{x})$:

$$\begin{aligned} \max \quad & \sum_{S\subseteq[n]\wedge|S|\leq d} \alpha_S \hat{p}_S \\ \text{s.t.} \quad & \textstyle\sum_{S\subseteq[n]\wedge|S|\leq d} \alpha_S g_S(\mathbf{x}) \in [-1,1], \quad \forall \quad \mathbf{x} \in \{-1,1\}^n \ , \\ & \alpha_S \in [-1,1], \quad \forall \quad S \subseteq [n] \wedge |S| \leq d \ . \end{aligned}$$

By Fact 8, the optimal solution of the LP must be $\Omega(\epsilon)$. Furthermore, let α_S be any optimal solution to the LP and let $p(\mathbf{x}) = \sum_{S\subseteq[n]\wedge|S|\leq d} \alpha_S p(\mathbf{x})$. Then we have

$$\begin{aligned}\mathbf{E}_{(\mathbf{x},y)\sim D}[yp(\mathbf{x})] =& \sum_{S\subseteq[n]\wedge|S|\leq d} \alpha_S \mathbf{E}_{(\mathbf{x},y)\sim D}[yg_S(\mathbf{x})] \geq \sum_{S\subseteq[n]\wedge|S|\leq d} \alpha_S \left(\hat{p}_S - 2^{-\tilde{O}(n^{1/2}\log(1/\epsilon))}\right)\\ =&\Omega(\epsilon)\ .\end{aligned}$$

It only remains for us to show how to get a hypothesis with error $1/2 - \Omega(\epsilon)$ from such a p.

Notice that given $p(\mathbf{x}) \in [-1,1]$ for all $\mathbf{x}$ and $\mathbf{E}_{(\mathbf{x},y)\sim D}[yp(\mathbf{x})] = \Omega(\epsilon)$, we have that the L_1 loss of p is

$$\mathbf{E}_{(\mathbf{x},y)\sim D}[|p(\mathbf{x}) - y|] = \mathbf{E}_{(\mathbf{x},y)\sim D}[1 - (yp(\mathbf{x}))] = 1 - \Omega(\epsilon)\ .$$

To convert the L_1 loss to the 0-1 loss of the output hypothesis, we first discretize the interval $[-1,1]$. Let $T\{0, c\epsilon, 2c\epsilon, \cdots, \lceil 2/\epsilon\rceil c\epsilon\}$, where $|T| \geq 2/(c\epsilon) - 1$ and c is a sufficiently small constant. Let $t \sim u(T)$, and define the corresponding random hypothesis h_t as $h(\mathbf{x}) = \mathrm{sign}(p(\mathbf{x}) - t)$. Then notice that the expected 0-1 loss of h_t is

$$\begin{aligned}\mathbf{E}_{t\sim u(T)}\left[\mathbf{E}_{(\mathbf{x},y)\sim D}[h_t(\mathbf{x}) \neq y]\right] =& \mathbf{E}_{(\mathbf{x},y)\sim D}\left[\mathbf{E}_{t\sim u(T)}[h_t(\mathbf{x}) \neq y]\right]\\ =& \mathbf{E}_{(\mathbf{x},y)\sim D}\left[\mathbf{E}_{t\sim u(T)}[t \in [p(\mathbf{x}), y] \cup [y, p(\mathbf{x})]]\right]\\ \leq& \mathbf{E}_{(\mathbf{x},y)\sim D}\left[\frac{|p(\mathbf{x}) - y|/(c\epsilon) + 1}{2/(c\epsilon) - 1}\right]\\ \leq& \mathbf{E}_{(\mathbf{x},y)\sim D}\left[\frac{|p(\mathbf{x}) - y| + 2c\epsilon}{2}\right]\\ =& \mathbf{E}_{(\mathbf{x},y)\sim D}[|p(\mathbf{x}) - y|]/2 + c\epsilon \leq 1/2 - \Omega(\epsilon)\ .\end{aligned}$$

Therefore, there must be a $t \in T$ such that $\mathbf{E}_{(\mathbf{x},y)\sim D}[h_t(\mathbf{x}) \neq y] \leq 1/2 - \Omega(\epsilon)$. Notice that we can query the value of $\mathbf{E}_{(\mathbf{x},y)\sim D}[h_t(\mathbf{x}) \neq y]$ using CSQ queries as

$$\begin{aligned}\mathbf{E}_{(\mathbf{x},y)\sim D}[h_t(\mathbf{x}) \neq y] =& 1/2(1 - (\mathbf{E}_{(\mathbf{x},y)\sim D}[h_t(\mathbf{x}) = y] - \mathbf{E}_{(\mathbf{x},y)\sim D}[h_t(\mathbf{x}) \neq y]))\\ =& 1/2(1 - \mathbf{E}_{(\mathbf{x},y)\sim D}[yh(\mathbf{x})])\ .\end{aligned}$$

Therefore, we can simply check $\mathbf{E}_{(\mathbf{x},y)\sim D}[h_t(\mathbf{x}) \neq y]$ for all $t \in T$ using CSQ queries with $c\epsilon$ error tolerance for a sufficiently small constant c, and then output the h_t with the smallest error. ■

CSQ Lower Bound on Weak Agnostic Learning The following fact about CSQ lower bound for distribution-free agnostic learning disjunctions is given in Gollakota et al. (2020).

Fact 9 *Any CSQ algorithm for distribution-free agnostic learning disjunctions on* $\{0,1\}^n$ *to error* $\mathrm{OPT} + 1/100$ *either requires a query of tolerance* $2^{-\Omega(n^{1/2})}$ *or* $2^{\Omega(n^{1/2})}$ *queries.*

We include the proof here for completeness. The CSQ lower bound here follows from the approximate degree of disjunctions. The following definition and facts from Gollakota et al. (2020) state the relation between CSQ lower bounds and the approximate degree. We first give the definition of pattern restriction from the pattern matrix method in Sherstov (2008).

Definition 16 (Pattern restrictions) *Let $C = \bigcup_{n\in\mathbb{N}} C_n$ be the union of some classes C_n of Boolean-valued function on $\{0,1\}^n$. We say C is closed under pattern matric restriction if for any k, n that is a multiple of k, and any $f \in C_k$, the function $\mathbf{x} \mapsto f(\mathbf{x}_V \oplus \mathbf{w})$ on $\{0,1\}^n$ lies in C_n for any $V \subseteq [n]$ of size k and $\mathbf{w} \in \{0,1\}^k$. In the common case where n is a small constant multiple of k, we will often be somewhat loose and not explicitly distinguish between C_k and C_n and just refer to C. Indeed, one can consider C_k to effectively be a subset of C_n using only some k out of n bits.*

Fact 10 (Theorem 1.2 of Gollakota et al. (2020)) *Let C be a Boolean-valued function class close under pattern restriction (Definition 16), with $1/2$-approximate degree $\Omega(d)$. Any distribution-free agnostic learner for C using only correlational statistical queries of tolerance $\tau \le 1/10$ requires at least $2^{\Omega(d)}\tau^2$ queries in order to agnostically learn C up to excess error $1/100$, i.e., true error $\mathrm{OPT} + 1/100$.*

We will combine the above fact with the following lower bound on the approximate degree of disjunctions.

Fact 11 (see, e.g., Theorem 23 of Bun and Thaler (2022)) *The $1/2$-approximate degree of disjunctions is $\Omega(\sqrt{n})$.*

Combining the above gives the CSQ lower bound.

Proof [Proof for Fact 9] This follows from combining Fact 10 and Fact 11. We take $k = cn$ and $\tau = 2^{cn^{1/2}}$ where c is a sufficiently small constant. Then Fact 11 implies that the function class of disjunctions on $\{0,1\}^k$ has an approximate degree of $\Omega(n^{1/2})$. Given this, an application of Fact 10 proves the statement. ■