Agnostically Learning Decision Trees

Parikshit Gopalan*
University of Washington
parik@cs.washington.edu

Adam Tauman Kalai†
Georgia Tech
atk@cc.gatech.edu

Adam R. Klivans‡
UT-Austin
klivans@cs.utexas.edu

ABSTRACT

We give a query algorithm for agnostically learning decision trees with respect to the uniform distribution on inputs. Given black-box access to an *arbitrary* binary function f on the n-dimensional hypercube, our algorithm finds a function that agrees with f on almost (within an ϵ fraction) as many inputs as the best size-t decision tree, in time $\mathrm{poly}(n, t, 1/\epsilon)$. This is the first polynomial-time algorithm for learning decision trees in a harsh noise model. We also give a *proper* agnostic learning algorithm for juntas, a sub-class of decision trees, again using membership queries.

Conceptually, the present paper parallels recent work towards agnostic learning of halfspaces [13]; algorithmically, it is significantly more challenging. The core of our learning algorithm is a procedure to implicitly solve a convex optimization problem over the L_1 ball in 2^n dimensions using an approximate gradient projection method.

1. INTRODUCTION

Decision tree learning is one of the central problems in computational learning [19, 15]. In practice, decision trees are a key ingredient in the most competitive machine learning and statistics systems such as CART and C4.5 [4, 2, 17]. Trees are often built top-down based on simple greedy splitting criteria. This raises a natural algorithmic question: How efficiently can one find the decision tree that best fits the data?

A seminal result, due to Kushilevitz and Mansour (KM) [15], is that decision trees are efficiently learnable under the uniform distribution using *membership queries*. Membership queries, a form of what is now popularly called "active learning," are black-box access to the *target function* to be learned $f : \{-1,1\}^n \to \{-1,1\}$, which, in this case, is assumed to be *noiseless*, i.e., computable by a poly-sized decision tree. Given such an oracle and $\epsilon > 0$ as input, the KM algorithm outputs a *hypothesis* $h : \{-1,1\}^n \to \{-1,1\}$ which disagrees with f on $\leq \epsilon$ fraction of $x \in \{-1,1\}^n$. More generally, their query algorithm learns sparse polynomials and is a key component to several other algorithms, including Jackson's celebrated result on learning DNF formulas [11].

However, the KM algorithm fails to address the following practical (and theoretical) concern about the *noiseless* assumption: in most cases of interest, the function to be learned is not believed to be *exactly computable* by a small decision tree. Indeed, the popularity of decision-tree induction is based on strong empirical evidence, across a number of fields, that decision trees often yield good approximations to complicated target functions. The present work aims to address this concern by giving a decision-tree learning algorithm in the *agnostic* setting [14]. In agnostic learning, no assumption is made about the target function to be learned. Instead, the goal of the learning algorithm is to output a hypothesis (not necessarily a decision tree) that predicts *nearly as well* as the best small decision tree. Hence, it is equivalent to learning with arbitrarily chosen or adversarial noise. For a concept class $\mathcal{C}$ and a target function f, let $\mathsf{opt}_{\mathcal{C}} = \min_{c \in \mathcal{C}} \Pr_{x \in \{-1,1\}^n}[c(x) \neq f(x)]$ be the error rate of the optimal concept in $\mathcal{C}$ with respect to f. The following is our main result:

THEOREM 1. *Let $\mathcal{C}$ be the class of decision trees with at most t leaves. There exists an algorithm that when given $t, \epsilon > 0$ and black-box access to any Boolean function $f : \{-1,1\}^n \to \{-1,1\}$, runs in time $\mathrm{poly}(n, t, \epsilon^{-1})$ and outputs a hypothesis $h : \{-1,1\}^n \to \{-1,1\}$ so that*

$$\Pr_{x \in \{-1,1\}^n}[h(x) \neq f(x)] \leq \mathsf{opt}_{\mathcal{C}} + \epsilon.$$

The hypothesis h is the sign of a sparse polynomial. Like KM, our algorithm actually learns the concept class of sparse polynomials, to which decision trees belong. More generally, our result holds for real-valued functions $f : \{-1,1\}^n \to [-1,1]$ where f is interpreted as a conditional probability: f specifies a distribution $\mathcal{D}_f$ on (x, y) where x is uniform over $\{-1,1\}^n$ and y is distributed so that $f(x) = \mathbb{E}_{\mathcal{D}_f}[y|x]$. Our result is the best agnostic extension one might hope for without further breakthroughs on the classical problem of learning decision trees without noise. Removing the assumption of membership queries looks hard since without queries, the fastest known algorithm for learning poly-size decision trees (with no noise) with respect to the uniform distribution from random examples takes time $n^{O(\log n)}$ [5].

When combined with results of Feldman *et al.* [6], our algorithm shows that the problem of agnostically learning

*Work done in part while the author was at UT-Austin.

†Research supported in part by NSF SES-0734780

‡Research supported by an NSF CAREER Award and NSF Grant CCF-0728536

STOC'08, May 17–20, 2008, Victoria, British Columbia, Canada.

sparse polynomials under the uniform distribution from random examples (without queries) reduces to the problem of learning parity with classification noise, a.k.a the noisy parity problem. Feldman *et al.* gave such a reduction from learning sparse polynomials with random classification noise.

Proper learning for Juntas: Another reason why decision trees are popular hypotheses in practice is because they are simple to understand. Thus, it is natural to look for learning algorithms that output a decision tree; such a learner that outputs a hypothesis from the target class $\mathcal{C}$ is called a *proper learner*. There are no proper learners known for decision trees even in the noiseless setting. However, we give a proper learner for the easier problem of agnostically learning k-juntas, which are functions that depend on only k out of the n inputs. A k-junta can be represented by a decision tree with 2^k leaves. The problem of agnostically learning k-juntas is to find the best predictor for a given function $f : \{-1,1\}^n \to \{-1,1\}$ which depends on at most k variables.

THEOREM 2. *Let $\mathcal{C}$ be the class of k-juntas. There exists an algorithm that, given $\epsilon > 0, k, n \geq 1$ and oracle access to arbitrary $f : \{-1,1\}^n \to \{-1,1\}$, runs in time* $\mathrm{poly}(n, k^k, \epsilon^{-k})$ *and outputs a k-junta h such that,*

$$\Pr_{x\in\{-1,1\}^n}[h(x) \neq f(x)] \leq \mathsf{opt}_{\mathcal{C}} + \epsilon.$$

The running time grows as k^k, which is polynomial in n only if $k = O(\frac{\log n}{\log\log n})$. In contrast, the (improper) algorithm in Theorem 1 agnostically learns $O(\log n)$-juntas in polynomial time.

1.1 Fourier-based Learning Algorithms

We describe three illuminating prior Fourier algorithms for learning under the uniform distribution in terms of the optimization problems they solve. Their approach to learning $f : \{-1,1\}^n \to \{-1,1\}$ can be described simply in terms of learning multivariate polynomials, where the monomials correspond to the character functions $\chi_S(x) = \prod_{i\in S} x_i$, for all $S \subseteq [n]$.

The *low-degree algorithm* of Linial, Mansour, and Nisan [16] (LMN) learns low-degree polynomials. Let $\mathcal{P}_d$ denote the polynomials of total degree $\leq d$. These have $n^{O(d)}$ terms. Their algorithm approximately solves the following minimization problem:

$$\min_{P\in\mathcal{P}_d} \mathop{\mathbb{E}}_{x\in\{-1,1\}^n} \left[|P(x) - f(x)|^2\right]. \tag{1}$$

The LMN algorithm fits these coefficients to $m = n^{O(d)}$ random labeled examples $(x_i, f(x_i))$, without the need for membership queries. Many natural classes of functions, ranging from halfspaces to AC^0, are well-approximated by polynomials of varying degree, and LMN learns these (noiselessly) with corresponding degrees of efficiency.

Kearns, Schapire, and Sellie [14] first considered Fourier-based methods for a type of "weak" agnostic learning. Further, Kalai *et al.* [13] (KKMS) showed that any concept class $\mathcal{C}$ with "good" low-degree Fourier concentration can be weakly agnostically learned; i.e., they showed that LMN could be easily modified to output a hypothesis with error $O(\mathsf{opt}) + \epsilon$ in the agnostic setting. Subsequently, Jackson (using an observation by Bshouty) presented an improved analysis which gives a bound of $2\mathsf{opt} + \epsilon$ [12].

The main result of Kalai et al. is a *strong* agnostic learning algorithm for halfspaces (and other suitably concentrated concept classes). That is, the KKMS algorithm outputs a hypothesis with error $\mathsf{opt} + \epsilon$. Achieving $\mathsf{opt} + \epsilon$, rather than $O(\mathsf{opt}) + \epsilon$, is fundamentally important in learning theory. Consider a typical boosting scenario where we can only guarantee the existence of a weak learner with accuracy $1/2 + 1/\mathrm{poly}(n)$, so that $\mathsf{opt} = 1/2 - 1/\mathrm{poly}(n)$. A weak agnostic learner might output a hypothesis with accuracy $1/2$, but this is useless if we wish to boost. A strong agnostic learner however, is guaranteed to find a hypothesis with accuracy bounded away from $1/2$.

To obtain their agnostic learning algorithm for functions with low-degree Fourier concentration, KKMS considered the following problem:

$$\min_{P\in\mathcal{P}_d} \mathop{\mathbb{E}}_{x\in\{-1,1\}^n} [\ |P(x) - f(x)|\ ]. \tag{2}$$

Their solution is to view this problem as a *linear* regression problem in the $n^{O(d)}$-dimensional space of coefficients and solve it by linear programming. Intuitively, ℓ_1 regression is better suited for agnostic learning – imagine starting with a (noiseless) low-degree f and flipping any η fraction of the $\{-1,1\}$ values of f. This can change the ℓ_1 "score" above of P by at most η, but the ℓ_2 score can change by significantly more (since P might take values outside $[-1,1]$).

The KM algorithm, on the other hand, learns *sparse* polynomials of arbitrary degree with respect to the ℓ_2 norm. It is well-known that sparsity, meaning having few nonzero coefficients, is closely related to the notion of the sum of magnitudes of Fourier coefficients being small. Let K_t denote the set of polynomials for which this sum is at most t (decision trees with at most t leaves fit into this category [15]). The KM algorithm approximately solves the following problem:

$$\min_{P\in K_t} \mathop{\mathbb{E}}_{x\in\{-1,1\}^n} \left[|P(x) - f(x)|^2\right] \tag{3}$$

Equivalently, one can view KM as an agnostic learner for parities, finding all Fourier coefficients above a certain threshold (like the Goldreich-Levin algorithm [10]). In terms of techniques, (3) is more challenging than (1) or (2), since in those cases the set of coefficients is fixed, whereas KM must discover the list of large coefficients amongst all 2^n possibilities.

1.2 Sparse ℓ_1 Regression

We present an agnostic analog of KM for concepts such as decision trees that are well-approximated by sparse polynomials. Our main algorithm approximately solves the following problem that we refer to as the *sparse ℓ_1 regression problem*:

$$\min_{P\in K_t} \mathop{\mathbb{E}}_{x\in\{-1,1\}^n} [\ |P(x) - f(x)|\ ] \tag{4}$$

One can cast (4) as a convex optimization problem with 2^n variables, using either the Fourier coefficients $\hat{P}(S), S \subset [n]$, or its pointwise values $P(x), x \in \{-1,1\}^n$ as variables. Since sparse polynomials have compact Fourier representations, it is natural to use the former approach. If we knew the support of the optimal P, then one could find P by solving the resulting LP. It is, however, unclear how to do this. One may guess that a natural candidate ought to be the set of Fourier coefficients returned by running KM on f, but it is unclear why this should be the case. Although Jackson's

result implies a guarantee of $2\mathsf{opt}+\epsilon$ [12], the true optimal solution could well involve coefficients that are not in the support of f. Our main result is a strong agnostic learner for decision trees that outputs a hypothesis with error $\mathsf{opt}+\epsilon$.

Our solution to (4) uses the gradient-projection method from convex optimization which iterates a *gradient* step and a *projection* step. In the *gradient* step, we move in the direction opposite the gradient of the function to be minimized. Since the gradient step might take us outside of the feasible set K_t, one moves back via a *projection* step to the closest point in K_t (in Euclidean distance). A simple analysis due to Zinkevich [20] shows that this procedure approaches the optimum fairly quickly on a wide class of problems.

Differentiating the objective function in (4) gives the (sub)gradient function $\mathrm{sgn}(P(x)-f(x))$. While this function is easy to compute pointwise, we will need to rewrite it in the Fourier basis, where it need not have a compact Fourier representation. Thus, in polynomial time, we can only compute a very weak approximation to the gradient, namely we can only guarantee that we have a good L_∞ estimate for every Fourier coefficient via the KM algorithm. This is problematic, as the L_1 or L_2 difference could be large, as we are working in 2^n dimensions. Since the gradient computation is rather inaccurate, the gradient step may take us to a point that is far in L_2 distance from where we should be (were the algorithm run without using KM). This is a problem as Zinkevich's analysis proceeds by showing that the squared L_2 distances from the optimal solution decrease.

Our key insight is that given points P, P' so that $L_\infty(P-P') \leq \epsilon$, we can project them onto the L_1 ball, retrieving points Q and Q' where $L_2(Q-Q') \leq O(\sqrt{\epsilon t})$. Thus, if we take $\epsilon = 1/\mathrm{poly}(t)$, the projection step gets us close in Euclidean distance to the correct point, which allows us to use the standard analysis of the gradient descent algorithm. This is a departure point from previous work on the gradient-projection method [20, 7], since the properties of the L_1 ball in high dimensions are crucial for us. Indeed, the same arguments no longer work for the L_p ball for $p > 1$.

Our proper agnostic learner for k-juntas uses a different approach: the crucial step is to characterize the best k-junta for predicting a given function f in terms of the Fourier expansion of f. We then show that one can find a good predictor among subsets of a set R of $O(k^2)$ variables with large low-degree influence.

We present our algorithm for sparse ℓ_1 regression and defer details of how it solves the problem of agnostic learning to Appendix A. We discuss relations between our work and recent work on Compressed Sensing in Appendix B.

2. PRELIMINARIES

Any function $P : \{-1,1\}^n \to \mathbb{R}$ can be represented as a polynomial, $P(x) = \sum_{S\subseteq[n]} \hat{P}(S)\chi_S(x)$, where $\chi_S(x) = \prod_{i\in S} x_i$ and $\hat{P}(S)$ is the Fourier coefficient of S. Let $\mathrm{supp}(P) = \{S \mid \hat{P}(S) \neq 0\}$ be the support of P. Define a binary sign function, $\mathrm{sgn} : \mathbb{R} \to \{-1,1\}$, where $\mathrm{sgn}(x) = 1$ iff $x \geq 0$.

We define the L_p norms of the coefficient vectors in 2^n dimensions: $L_1(P) = \sum_S |\hat{P}(S)|, L_2(P) = (\sum_S \hat{P}(S)^2)^{\frac{1}{2}}$, $L_\infty(P) = \max_S |\hat{P}(S)|$.

We define the ℓ_p norms of the function P for $p \geq 1$ as

$$\|P(x)\|_p = \mathbb{E}_{x\in\{-1,1\}^n}[|P(x)|^p]^{\frac{1}{p}},$$
$$\|P(x)\|_\infty = \max_{x\in\{-1,1\}^n} |P(x)|.$$

We define the inner product of two functions as,

$$P \cdot Q = \mathbb{E}_x[P(x)Q(x)].$$

By orthogonality of characters, $P \cdot Q = \sum_S \hat{P}(S)\hat{Q}(S)$. A special case is Parseval's identity: $P\cdot P = \|P(x)\|_2^2 = L_2(P)^2$.

Given an oracle for $P : \{-1,1\}^n \to \mathbb{R}$ and $\theta > 0$, the KM algorithm returns a list of size at most $L_2(P)^2\theta^{-2}$ containing all S such that $|\hat{P}(S)| \geq \theta$. One can estimate these coefficients accurately by sampling, and set the other coefficients to 0 to get a sparse approximation Q for P.

LEMMA 3. *[15] Given an oracle for $P : \{-1,1\}^n \to \mathbb{R}$, $\mathrm{KM}(P,\theta)$ returns $Q : \{-1,1\}^n \to \mathbb{R}$ with $|\mathrm{supp}(Q)| \leq O(L_2(P)^2\theta^{-2})$ and $L_\infty(P-Q) \leq \theta$. The running time is $\mathrm{poly}(n, \theta^{-1}, L_2(P))$.*

In general, P need not have a good sparse L_2 approximation: for instance if all the Fourier coefficients of P are less than θ, then $Q = 0$. We say that a polynomial is t-sparse if $L_1(P) \leq t$. It is well-known that decision trees with t leaves are t-sparse [15]. Let K_t denote the convex set $\{P : L_1(P) \leq t\}$. For t-sparse polynomials, most of the Fourier mass is concentrated on a few Fourier coefficients, and we get the following stronger guarantee:

LEMMA 4. *[15] If P is t-sparse, then $KM(P, \frac{\epsilon^2}{2t})$ returns Q s.t. $\|P-Q\|_2 \leq \epsilon$.*

Let $\mathcal{D}$ be a distribution on $X \times Y$ for $X = \{-1,1\}^n$ and $Y = \{-1,1\}$ such that the marginal distribution on X is uniform. A membership query oracle for $\mathcal{D}$ returns $y \in Y$ distributed according to $\mathcal{D}|x$ for a query $x \in X$. Let $\mathcal{C}$ be a concept class of Boolean functions. Define the error of $c \in \mathcal{C}$ on $\mathcal{D}$ as $\mathrm{err}_{\mathcal{D}}(c) = \Pr_{\langle x,y\rangle\leftarrow\mathcal{D}}[c(x) \neq y]$ and the optimal error of $\mathcal{C}$ on $\mathcal{D}$ as $\mathsf{opt} = \min_{c\in\mathcal{C}} \mathrm{err}_{\mathcal{D}}(c)$.

DEFINITION 1. *A concept class $\mathcal{C}$ is agnostically learnable with queries under the uniform distribution if there is an algorithm which when given a query oracle for $\mathcal{D}$ and parameters ϵ, δ as inputs, returns a hypothesis $h : \{-1,1\}^n \to \{-1,1\}$ such that $\Pr_{\langle x,y\rangle\leftarrow\mathcal{D}}[h(x) \neq y] \leq \mathsf{opt}+\epsilon$ with probability $1-\delta$.*

3. IDEALIZED PROJECTED SUBGRADIENT DESCENT FOR SPARSE ℓ_1 REGRESSION

In this section we give an *idealized* algorithm for solving the sparse ℓ_1 regression problem via a steepest descent optimization procedure. The goal of the procedure is, given a convex function $F : K \to \mathbb{R}$ on compact convex set $K \subseteq \mathbb{R}^N$, to find $P \in K$ such that $F(P) \leq \min_{Q\in K} F(Q) + \epsilon$. While the method is quite old [18], we know of no simpler rate bounds than a recent analysis due to Zinkevich [20] for a more general online version of the algorithm.

Since the function we minimize is convex but non-differentiable, we use the generalization of gradients to non-differentiable functions. Formally, $V \in \mathbb{R}^N$ is a *subgradient* of convex $F : K \to \mathbb{R}$ at P, written $V \in \nabla_F P$, if for every $Q \in K$, we have $F(Q) \geq F(P) + V \cdot (Q-P)$. Assume

that we have an oracle that computes a subgradient of F at any point P. Also assume that the convex set $K \subset \mathbb{R}^N$ is represented by a projection oracle, $\text{proj}_K(P)$ which returns the point in K which is closest in L_2 to P. The gradient projection method (often called the *projected subgradient method*) chooses a sequence of points, starting with an arbitrary $P_1 \in K$ and then taking $P_{i+1} = \text{proj}_K(P_i - \eta V_i)$, where $\eta > 0$ is a *step size* and $V_i \in \nabla_F(P_i)$.

One can translate this to our setting to get an algorithm for sparse ℓ_1 regression that takes time $2^{O(n)}$. We wish to optimize over the convex set $K_t = \{P : L_1(P) \leq t\}$. We view functions $P : \{-1,1\}^n \to \mathbb{R}$ as vectors in 2^n dimensions, where the co-ordinates correspond to the Fourier coefficients. The Fourier representation is used since polynomials in K_t have sparse representations. The objective function we wish to minimize is $\text{err}_f : \mathbb{R}^{2^n} \to \mathbb{R}$, which is defined as $\text{err}_f(P) = \|P - f\|_1$. It is easy to give a projection oracle for the L_1 ball. We next address the sub-gradient computation. Given a function P, define the function $\nabla_f P : \{-1,1\}^n \to \{-1,1\}$ as $\nabla_f P(x) = \text{sgn}(P(x) - f(x))$. While we have defined $\nabla_f P$ by its pointwise values, we may view it as a vector in $\mathbb{R}^{2^n}$ via its Fourier expansion (though rewriting it in the Fourier basis takes time 2^n). The next claim shows that $\nabla_f P$ is indeed a sub-gradient for err_f at P ($\nabla_f P \in \nabla_{\text{err}_f} P$).

LEMMA 5. *For any polynomials P and Q, $\nabla_f P \cdot (P - Q) \geq err_f(P) - err_f(Q)$.*

PROOF: We use the inequality that $|a - c| \geq |b - c| + (a - b)\,\text{sgn}(b - c)$ for reals a, b, c: $|b-c| + (a-b)\,\text{sgn}(b-c) = (b - c)\,\text{sgn}(b-c) + (a-b)\,\text{sgn}(b-c) = (a-c)\,\text{sgn}(b-c) \leq |a-c|$. By applying this to $Q(x), P(x), f(x)$, we get

$$\begin{aligned} &|Q(x) - f(x)| \\ &\geq |P(x) - f(x)| + (Q(x) - P(x))\,\text{sgn}(P(x) - f(x)) \\ &= |P(x) - f(x)| + (Q(x) - P(x))\nabla_f P(x). \end{aligned}$$

Taking expectations on both sides and using $A \cdot B = \mathbb{E}_x[A(x)B(x)]$ we have,

$$\mathbb{E}_x[|Q(x) - f(x)|] \geq \mathbb{E}_x[|P(x) - f(x)|] + \mathbb{E}_x[(Q(x) - P(x))\nabla_f P(x)]$$

$$\Rightarrow \|Q(x) - f(x)\|_1 = \|P(x) - f(x)\|_1 + (Q - P) \cdot \nabla_f P.$$

Hence $\text{err}_f(Q) \geq \text{err}_f(P) + (Q - P) \cdot \nabla_f P$. The claim follows by rearranging terms. □

ALGORITHM 1. IDEALIZED ALGORITHM
Inputs: **integer** $T \geq 1$ **and real** $\eta \in (0,1)$.
$P_0 := 0$.
For $k = 1, 2, \ldots, T$:
1. $P'_k := P_{k-1} - \eta \nabla_f P_{k-1}$.
2. **Let** $P_k := \text{proj}_K(P'_k)$.

Return the best P_k **over** $k = 1, 2, \ldots, T$.

One can use the standard analysis of gradient descent [20] to show that following algorithm will successfully find a polynomial in K that approximately minimizes $err_f(P)$. However, the algorithm takes time $\Omega(2^n)$ since it works with vectors in 2^n dimensions.

THEOREM 6. *Let $P_* \in K_t$ be the polynomial that minimizes err_f. Let $T \geq 1$ and $\eta = t/\sqrt{T}$. If we run Algorithm 1 for T steps, for some $k \leq T$, P_k satisfies $\text{err}_f(P_k) \leq \text{err}_f(P_*) + \eta$.*

4. AN EFFICIENT IMPLEMENTATION OF THE IDEALIZED ALGORITHM

In order to design an efficient analogue of Algorithm 1, rather than working with polynomials with 2^n coefficients, we only compute and store sparse approximations to the various polynomials involved using KM. Computing the gradient via KM is problematic since it may not be even weakly approximated (in L_2) by sparse polynomials. We circumvent this by analyzing the projection operator onto the L_1 ball in detail and show that it works well even with a weak L_∞ approximation given by KM. Additionally, we need to show how to compute the sub-gradient and projection operators efficiently from these approximations. We state our efficient gradient descent algorithm using KM:

ALGORITHM 2. GRADIENT DESCENT USING KM
Inputs: **integer** $T \geq 1$ **and reals** $\eta, \theta \in (0,1)$.
$P_0 := 0$.
For $k = 1, 2, \ldots, T$:
1. $P'_k := P_{k-1} - \eta\,\text{KM}(\nabla_f P_{k-1}, \theta)$.
2. **Let** $P_k := \text{KM}(\text{proj}_K(P'_k), \theta)$.

Return the best P_k **over** $k = 1, 2, \ldots, T$.

The parameter θ will be fixed later. For all k, P_k will be a t-sparse polynomial with $\ell = \text{poly}(t, \epsilon^{-1})$ non-zero coefficients. To compute $KM(\nabla_f P_{k-1}, \theta)$ we need an oracle for $\nabla_f P_{k-1} = \text{sgn}(P_{k-1} - f)$. We can simulate this oracle, as P_{k-1} is stored as a sparse polynomial, and we are given an oracle for f. Although P_{k-1} is sparse, $\nabla_f P_{k-1}$ could be far from sparse, and all we can guarantee (using Lemma 3) is an L_∞ approximation. Lemma 7 shows how to compute $\text{proj}_K(P'_k)$ from P'_k efficiently. Applying KM in step 2 maintains the invariant that $|\text{supp}(P_k)| \leq \ell$.

In Section 4.1 we analyze the projection step in Algorithm 1 in detail, and in Section 4.2 we show that if P' is such that $||P - P'||_\infty$ is small then $||\text{proj}_K(P) - \text{proj}_K(P')||_2$ is small. In Section 4.3 we present the full analysis of Algorithm 2.

4.1 Projecting onto the L_1 Ball

The *project* operator $\text{proj}_K(P)$ for $P : \{-1,1\}^n \to \mathbb{R}$ maps P to the closest Q in Euclidean distance that satisfies $L_1(Q) \leq t$ (we write proj_K rather than proj_{K_t} for simplicity). Formally, $\text{proj}_K(P) = \arg\min_{L_1(Q) \leq t} \|P - Q\|_2$. If we wanted $|\text{supp}(Q)| = t$, then truncating P to its t largest Fourier coefficients suffices. However since we want $L_1(Q) \leq t$, we need to be more careful.

DEFINITION 2. *Given a function P and $\ell \geq 0$, define* $\text{shrink}(P, \ell)$ *as the function Q where*

$$\hat{Q}(S) = \begin{cases} \hat{P}(S) - \ell & \text{if } \hat{P}(S) \geq \ell \\ \hat{P}(S) + \ell & \text{if } \hat{P}(S) \leq -\ell \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

LEMMA 7. *For any P, $\text{proj}_K(P) = \text{shrink}(P, \ell)$ for the smallest $\ell \geq 0$ so that $\text{shrink}(P, \ell) \in K_t$.*

PROOF: If $L_1(P) \leq t$, then clearly $\text{proj}_K(P) = P$ and the claim holds. So assume that $L_1(P) = t' > t$. Since $\|P - Q\|_2 = L_2(P - Q)$, we can restate the problem as

$$\text{Minimize} \sum_S (\hat{P}(S) - \hat{Q}(S))^2 \text{ over } Q \in K \quad (6)$$

We claim that the optimal solution Q satisfies $\text{sgn}(\hat{P}(S)) = \text{sgn}(\hat{Q}(S))$ for every S. If this were not true, setting $\hat{Q}(S) = 0$ would simultaneously reduce $L_2(P-Q)$ and $L_1(Q)$, thus giving a better solution to (6). Similarly, one can show that $|\hat{Q}(S)| \leq |\hat{P}(S)|$. From now on, we will assume that $\hat{P}(S) \geq 0$ for all S. Let $\hat{Q}(S) = \hat{P}(S) - \ell(S)$ where $0 \leq \ell(S) \leq \hat{P}(S)$. Note that the set K is convex and P lies outside this set, so Q will lie on the surface of K, hence $L_1(Q) = t$.

By the above conditions, we can rewrite (6) as

$$\text{Minimize} \sum_S \ell(S)^2$$

subject to

$$\sum_S \ell(S) = t' - t, \quad 0 \leq \ell(S) \leq \hat{P}(S).$$

Without the upper bounds $\ell(s) \leq \hat{P}(S)$, the best solution would be to take all $\ell(S)$ equal. With these bounds, we claim the best solution is to take all $\ell(S)$ as equal as possible. Fix the optimal solution to (4.1) and say that S is tight if $\ell(S) = \hat{P}(S)$. We claim that for any S, T both of which are not tight, $\ell(S) = \ell(T)$. For contradiction, assume $\ell(S) > \ell(T)$. Then increasing $\ell(T)$ and decreasing $\ell(S)$ by small amounts gives a feasible solution (since they are not tight), and decreases the objective function. Let $\ell = \ell(S)$ for any non-tight set S. Thus $\hat{Q}(S) = 0$ if S is tight, and $\hat{Q}(S) = \hat{P}(S) - \ell$ otherwise, which implies our claim. □

Lemma 7 shows how to compute $\text{proj}_K(P)$ if P is written as a sum of Fourier coefficients. We start decreasing all the Fourier coefficients of P by equal amounts, keeping their signs the same. If some coefficient reaches 0, it then stays at 0. We continue this till we reach a Q where $L_1(Q) = t$.

4.2 Projecting L_∞ Approximations

Algorithm 2 uses KM to get a sparse approximation to the gradient in Step 1. Since the gradient might be far from sparse, in time $\text{poly}(n, \epsilon^{-1})$, KM only guarantees an L_∞ approximation (see Lemma 3). Thus, if P is the point reached using the exact gradient, Step 1 takes us to P' s.t. $L_\infty(P - P') \leq \epsilon$. However $L_1(P - P')$ and $L_2(P - P')$ could be huge since we are in 2^n dimensions. However we will show that $L_2(\text{proj}_K(P) - \text{proj}_K(P')) \leq \sqrt{4\epsilon t}$, using the fact that proj_K does not change much under coordinate-wise perturbations.

LEMMA 8. *Let P, P' be such that $L_\infty(P - P') \leq \epsilon$. Then $L_\infty(\text{proj}_K(P) - \text{proj}_K(P')) \leq 2\epsilon$.*

PROOF: Let

$$Q = \text{proj}_K(P) = \text{shrink}(P, \ell),$$
$$Q' = \text{proj}_K(P') = \text{shrink}(P', \ell').$$

First assume that one of the points, say P already lies in the convex set K. Then it is clear that after reducing each coefficient $\hat{P}'(S)$ by at most ϵ, we get a point Q' such that $L_1(Q') \leq L_1(P) \leq t$. Thus we have $L_\infty(P - Q') \leq L_\infty(P - P') + L_\infty(P' - Q') \leq 2\epsilon$.

So assume that $P, P' \notin K$. We will show that in this case $|\ell - \ell'| < \epsilon$. The claim then follows by plugging this into Equation 5 and some simple case analysis. For contradiction, assume that $\ell < \ell' - \epsilon$. Define the set $\mathcal{S} = \{S : |\hat{Q}'(S)| > 0\}$. This set is non-empty since $L_1(Q') = t$. Note that $|\hat{P}(S)| \geq |\hat{P}'(S)| - \epsilon$. If we shrink P by ℓ and P' by $\ell' > \ell + \epsilon$, we get $|\hat{Q}(S)| = |\hat{P}(S)| - \ell > |\hat{P}'(S)| - \ell' = |\hat{Q}'(S)| \geq 0$. Summing over all sets $S \in \mathcal{S}$, we get $L_1(Q) \geq \sum_{S \in \mathcal{S}} |\hat{Q}(S)| > \sum_{S \in \mathcal{S}} |\hat{Q}'(S)| = t$. But this is a contradiction since $L_1(Q) = t$. □

LEMMA 9. *For P, P' such that $L_\infty(P - P') \leq \epsilon$, we have $\|\text{proj}_K(P) - \text{proj}_K(P')\|_2 \leq (4\epsilon t)^{\frac{1}{2}}$.*

PROOF: By Lemma 9,

$$L_\infty(\text{proj}_K(P) - \text{proj}_K(P')) \leq 2\epsilon.$$

Also,

$$\begin{aligned} L_1(\text{proj}_K(P) - \text{proj}_K(P')) &\leq L_1(\text{proj}_K(P)) + L_1(\text{proj}_K(P')) \\ &\leq 2t. \end{aligned}$$

By Hölder's inequality

$$\begin{aligned} &L_2(\text{proj}_K(P) - \text{proj}_K(P'))^2 \leq \\ &L_\infty(\text{proj}_K(P) - \text{proj}_K(P')) \cdot L_1(\text{proj}_K(P) - \text{proj}_K(P')) \\ &\leq 4\epsilon t. \end{aligned}$$

The claim now follows from Parseval's identity. □

This lemma shows that given only an oracle for P (and not its Fourier expansion), we can still project onto the L_1 ball with small L_2 error, by applying proj_K to $P' = KM(P, \epsilon)$. It is not clear if this is possible when K is the L_p ball for $p > 1$.

4.3 Analysis of Subgradient Descent using KM

To analyze Algorithm 2, we define the polynomials $Q'_k = P_{k-1} - \eta \nabla_f P_{k-1}$ and $Q_k = \text{proj}_K(Q'_k)$, which correspond to executing the k^{th} iteration of Algorithm 2 without using KM for sparsification. The crux of our analysis is to bound $\|P_k - Q_k\|_2$. A bound of $O(t)$ is trivial since both points lie in K_t. The next lemma shows that running KM with accuracy parameter θ actually gives $\|P_k - Q_k\|_2 = O(\sqrt{\theta t})$. Thus by running KM for $\text{poly}(t)$ time, this distance will approach 0.

LEMMA 10. *The polynomials P_k and Q_k satisfy $\|P_k - Q_k\|_2 \leq 4(\theta t)^{\frac{1}{2}}$.*

PROOF: We first show that $L_\infty(P'_k - Q'_k) \leq \theta$. We have

$$\begin{aligned} &P'_k - Q'_k \\ &= (P_{k-1} - \eta\, \text{KM}(\nabla_f P_{k-1}, \theta)) - (P_{k-1} - \eta \nabla_f P_{k-1}) \\ &= \eta(\nabla_f P_{k-1} - \text{KM}(\nabla_f P_{k-1}, \theta)). \end{aligned}$$

By Theorem 3, $L_\infty(\nabla_f P_{k-1} - \text{KM}(\nabla_f P_{k-1}, \theta)) \leq \theta$ which implies $L_\infty(P'_k - Q'_k) \leq \theta$, since $\eta \leq 1$.

Applying Lemma 9 to P' and Q', we get $\|\text{proj}_K(P'_k) - \text{proj}_K(Q'_k)\|_2 < (4\theta t)^{\frac{1}{2}}$. Note that $P_k = KM(\text{proj}_K(P'_k), \theta)$, and that $\text{proj}_K(P'_k)$ is t-sparse, hence by Lemma 4, KM gives a good ℓ_2 approximation: $\|P_k - \text{proj}_K(P'_k)\|_2 \leq (2\theta t)^{\frac{1}{2}}$. Since $Q_k = \text{proj}_K(Q'_k)$, by the triangle inequality,

$$\begin{aligned} &\|P_k - Q_k\|_2 \\ &\leq \|P_k - \text{proj}_K(P'_k)\|_2 + \|\text{proj}_K(P'_k) - \text{proj}_K(Q'_k)\|_2 \\ &\leq (4\theta t)^{\frac{1}{2}} + (2\theta t)^{\frac{1}{2}} \\ &< 4(\theta t)^{\frac{1}{2}}. \end{aligned}$$

□

The following lemma, which is the key to analyzing gradient descent shows that as long as $\text{err}_f(P_k)$ is much larger than $\text{err}_f(P_*)$, we move close to P_*.

LEMMA 11. *Let $P_* \in K$ be the polynomial that minimizes err_f. Then (for suitable choice of θ),*

$$\|P_k - P_*\|_2^2 - \|P_{k+1} - P_*\|_2^2 \geq 2\eta(\mathrm{err}_f(P_k) - \mathrm{err}_f(P_*)) - 2\eta^2.$$

PROOF:

Using Lemma 10 and the triangle inequality, $\|P_k - P_*\|_2 \leq \|Q_k - P_*\|_2 + 4(\theta t)^{\frac{1}{2}}$. Now observe that $\|Q_k - P_*\|_2 = L_2(Q_k - P_*) \leq L_1(Q_k - P_*) \leq L_1(Q_k) + L_1(P_*) \leq 2t$ where the last inequality holds since $Q_k, P_* \in K$. Hence for all k and $C < 100$,

$$\begin{aligned}&\|P_k - P_*\|_2^2 \leq \|Q_k - P_*\|_2^2 + 16t(\theta t)^{\frac{1}{2}} + 16\theta t \\ &\leq \|Q_k - P_*\|_2^2 + Ct\sqrt{\theta t}.\end{aligned}$$

Therefore, we have

$$\begin{aligned}&\|P_k - P_*\|_2^2 - \|P_{k+1} - P_*\|_2^2 \geq \\ &\|P_k - P_*\|^2 - \|Q_{k+1} - P_*\|_2^2 - Ct\sqrt{\theta t}. \quad \text{(A)}\end{aligned}$$

We now use the fact that projecting a point onto a convex set K reduces the distance to points in K, hence $\|Q_{k+1} - P_*\|_2 \leq \|Q'_{k+1} - P_*\|_2$. Plugging this into Equation (A),

$$\begin{aligned}&\|P_k - P_*\|_2^2 - \|P_{k+1} - P_*\|_2^2 \\ &\geq \|P_k - P_*\|^2 - \|Q'_{k+1} - P_*\|_2^2 - Ct\sqrt{\theta t} \\ &= (P_k - P_*)^2 - (P_k - P_* - \eta\nabla_f P_k)^2 - Ct\sqrt{\theta t} \\ &= 2\eta\nabla_f P_k \cdot (P_k - P_*) - \eta^2\nabla_f P_k^2 - Ct\sqrt{\theta t} \\ &\geq 2\eta(\mathrm{err}_f(P_k) - \mathrm{err}_f(P_*)) - \eta^2 - Ct\sqrt{\theta t} \quad \text{(By Lemma 5)}\end{aligned}$$

We will choose θ small enough that $Ct\sqrt{\theta t} < \eta^2$, which completes the proof. □

Using this lemma, we can now prove Theorem 12, which formally states the convergence properties of Algorithm 2.

THEOREM 12. *Let $P_* \in K_t$ be the polynomial that minimizes err_f. Let T be any positive integer. If Algorithm 2 is run for T steps with $\eta \leq \frac{t}{\sqrt{T}}$, $\theta \leq \frac{\eta^2}{C^2t^3}$ (for sufficiently small C) then for some $k \leq T$, $\mathrm{err}_f(P_k) \leq \mathrm{err}_f(P_*) + 2\eta$. The overall running time is $\mathrm{poly}(n, t, T)$.*

PROOF: By Lemma 11, the distance from P_k to P_* decreases as long as

$$2\eta(\mathrm{err}_f(P_k) - \mathrm{err}_f(P_*)) - 2\eta^2 \geq 0$$

and therefore implies $\mathrm{err}_f(P_k) \geq \mathrm{err}_f(P_*) + \eta$.

Moreover, for each k where $\mathrm{err}_f(P_k) \geq \mathrm{err}_f(P_*) + 2\eta$, we have $\|P_k - P_*\|_2^2 - \|P_{k+1} - P_*\|_2^2 \geq 2\eta^2$. Initially $P_0 = 0$, so $\|P_0 - P_*\|_2^2 = \|P_*\|_2^2 \leq L_1(P_*)^2 \leq t^2$. So by our choice of η, after $T = \frac{t^2}{2\eta^2}$ steps, there must be some $k \leq T$ such that $\|P_k - P_*\|_2^2 \leq 2\eta^2$. For this P_k by Lemma 5,

$$\begin{aligned}\mathrm{err}_f(P_k) - \mathrm{err}_f(P_*) &\leq (P_k - P_*) \cdot \nabla_f P_k \\ &\leq \|P_k - P_*\|_2 \|\nabla_f P_k\|_2 \\ &\leq 2\eta\end{aligned}$$

hence the claim holds. □

5. PROPERLY LEARNING JUNTAS

Recall that $h : \{-1,1\}^n \to \{-1,1\}$ is a k-junta if it depends on only k out of n variables. Given $f : \{-1,1\}^n \to \{-1,1\}$, our goal is to find the k-junta h such that $\Pr_{x\in\{-1,1\}^n}[f(x) \neq h(x)]$ is minimized. Define $\mathrm{err}_f(h) = \Pr_x[h(x) \neq f(x)]$. Let η be the minimum value of $\mathrm{err}_f(h)$ over all k-juntas h. If h only depends on variables in $K \subseteq [n]$, we will call it a K-junta. We first characterize the best K-junta for a function f. For a vector $x \in \{-1,1\}^n$ and $K \subseteq [n]$, let x_K denote the projection of x onto the coordinates in the set K.

LEMMA 13. *Given $K \subset [n]$, and $f : \{-1,1\}^n \to \{-1,1\}$, let $f_K(x) = \sum_{S\subseteq K} \hat{f}(S)\chi_S(x)$. The K-junta that minimizes err_f is given by $h_K(x) = \mathrm{sgn}(f_K(x))$. Further, $\mathrm{err}_f(h_K) = \frac{1}{2}(1 - \|f_K(x)\|_1)$.*

PROOF. Firstly, observe that h_K is really a K-junta since f_K depends only on x_K. Let us fix a value $u \in \{-1,1\}^k$. By $x|x_K = u$ we denote the random variable x where the indices in K are set according to u and the rest are uniformly random. This identifies a sub-cube $C_K(u)$ of $\{-1,1\}^n$. A K-junta will evaluate to the same value at every point in this sub-cube. Hence the agreement with f is maximized by the function $g_K : \{-1,1\}^k \to \{-1,1\}$ defined as

$$\begin{aligned}g_K(u) &= \mathrm{Maj}_{x\in C_K(u)} f(x) = \mathrm{sgn}(\mathbb{E}[f(x)|x_K = u]) \\ &= \sum_{S\subseteq[n]} \hat{f}(S)\,\mathbb{E}[\chi_S(x)|x_k = u].\end{aligned}$$

Since x_i is an unbiased $\{-1,1\}$ variable when $i \notin K$,

$$\mathbb{E}[\chi_S(x)|x_k = u] = \begin{cases}\chi_S(u) & \text{if } S \subseteq K \\ 0 & \text{otherwise}\end{cases}$$

Hence $\mathbb{E}[f(x)|x_K = u] = \sum_{S\subseteq K} \hat{f}_S\chi_S(u) \Rightarrow g_K(u) = \mathrm{sgn}(f_K(u)) = h_k(u)$.

Since $\mathbb{E}[f(x)|x_K = u] = f_K(x)$, and $f(x) \in \{-1,1\}$, we have

$$\begin{aligned}\Pr[f(x) = \mathrm{sgn}(f_K(x))|x_K = u] &= \tfrac{1}{2} + \tfrac{|f_K(u)|}{2} \\ \Pr[f(x) \neq \mathrm{sgn}(f_K(x))|x_K = u] &= \tfrac{1}{2} - \tfrac{|f_K(u)|}{2}\end{aligned} \quad (7)$$

Averaging this over all $u \in \{-1,1\}^k$, and observing that the uniform distribution on x induces the uniform distribution on x_K, we obtain $\mathrm{err}_f(h_K) = \Pr[h_K(x) \neq f(x)] = \frac{1}{2} - \frac{\mathbb{E}_x|f_K(x)|}{2} = \frac{1}{2}(1 - \|f_K(x)\|_1)$. □

Thus our goal is to find the k-subset $K \subseteq [n]$ such that $\mathbb{E}_x[|f_K(x)|]$ is maximized (which need not be the k-subset with the most Fourier mass). One can show that any function g_K which is close to f_K in ℓ_1 distance gives a good predictor.

LEMMA 14. *Let $g_K : \{-1,1\}^k \to \mathbb{R}$ be such that $\|f_K(x) - g_K(x)\|_1 < \epsilon$, and let $h'_K = \mathrm{sgn}(g_K)$. Then $\mathrm{err}_f(h'_K) < \mathrm{err}_f(h_K) + 2\epsilon$.*

PROOF. Let us fix $x_k = u$. Then by equation 7, we have

$$\mathrm{err}_f(h'_K|x_k = u] = \begin{cases}\mathrm{err}_f(h_K|x_k = u) \text{ if } \mathrm{sgn}(g(u)) = h_K(u) \\ \mathrm{err}_f(h_K|x_K = u) + 2|f_K(u)| \\ \quad \text{if } \mathrm{sgn}(g(u)) \neq h_K(u)\end{cases}$$

In either case,

$$\mathrm{err}_f(h'_K|x_K = u) \leq \mathrm{err}_f(h_K|x_K = u) + 2|f_K(u) - g_K(u)|$$

Averaging over all choices of u,

$$\mathrm{err}_f(h'_K) \leq \mathrm{err}_f(h_K) + 2\,\mathbb{E}_x\,|f_K(x) - g_K(x)| \leq \mathrm{err}_f(h_K) + 2\epsilon$$

□

Our algorithm first uses KM to identify all large Fourier coefficients in f. We retain only those variables which have large low-degree influence on g: let $I_i^{\leq k}(g) = \sum_{i \in S, |S| \leq k} \hat{g}(S)^2$ and discard variables where $I_i^{\leq k}(g) \leq \frac{\epsilon^2}{2k}$. The effect of this is to reduce the number of surviving variables to $O(k^2)$, while ensuring that the Fourier mass associated with the best set K does not reduce by much. We then return the best k-junta found by brute-force search.

ALGORITHM 3. AGNOSTIC JUNTA LEARNER

1. Run KM on f with $\theta = \epsilon 2^{-k/2}$ to get

$$g(x) = \sum_S \hat{g}(S)\chi_S(x).$$

2. Let $R = \{i \mid I_i^{\leq k}(g) \geq \frac{\epsilon^2}{k}\}$ and let

$$g'(x) = \sum_{S \subseteq R} \hat{g}(S)\chi_S(x).$$

3. For every $K \subseteq R$ of size k, let $h'_K = \mathrm{sgn}(g'_K)$ and estimate $\mathrm{err}_f(h'_K)$.
4. Return h'_K which minimizes $\mathrm{err}_f(h'_K)$.

THEOREM 15. *Algorithm 3 finds a k-junta h' such that $\mathrm{err}_f(h') \leq \eta + 5\epsilon$ in time $\mathrm{poly}(n, k^k, \epsilon^{-k})$.*

PROOF: Let K be the set such that $h_K = \mathrm{sgn}(f_K)$ has the least error η. In Step 1, we have $|\hat{g}(S) - \hat{f}(S)| < \theta$ for all $S \subseteq K$. Hence $\mathbb{E}_x |g_K(x) - f_K(x)|^2 = \sum_{S \subseteq K} |\hat{f}(S) - \hat{g}(S)|^2 \leq 2^k\theta^2 \leq \epsilon^2$.

In Step 2, we drop those variables i from g where $I_i^{\leq k}(g) < \frac{\epsilon^2}{k}$. Assume that we drop $k' \leq k$ variables from the set K. The total Fourier mass on the coefficients involving these variables is bounded by $k'\frac{\epsilon^2}{k} \leq \epsilon^2$. Hence $\mathbb{E}_x |g_K(x) - g'_K(x)|^2 \leq \epsilon^2$, so

$$\begin{aligned}
&\mathbb{E}_x |f_K(x) - g'_K(x)| \\
&\leq \mathbb{E}_x |f_K(x) - g_K(x)| + \mathbb{E}_x |g_K(x) - g'_K(x)| \\
&\leq (\mathbb{E}_x |f_K(x) - g_K(x)|^2)^{1/2} + (\mathbb{E}_x |g_K(x) - g'_K(x)|^2)^{1/2} \\
&\leq \epsilon + \epsilon \\
&= 2\epsilon.
\end{aligned}$$

Thus by Lemma 14, $\mathrm{err}_f(g'_K) \leq \eta + 4\epsilon$. It is easy to show using Chernoff bounds that the predictor h'_K returned in Step 4 is not much worse, $\mathrm{err}_f(h'_K) \leq \eta + 5\epsilon$.

To bound the time taken in Step 3, we show that $|R| = O(k^2)$. Note that $\sum_{i \in [n]} I_i^{\leq k}(g) = \sum_{S:|S| \leq k} |S|\hat{g}(S)^2 \leq k\sum_S \hat{g}(S)^2 \leq k$. So at most $k^2\epsilon^{-2}$ variables satisfy $I_i^{\leq k}(g) \geq \epsilon^2/k$. Hence $|R| \leq k^2\epsilon^{-2}$. Thus the number of choices in Step 3 is bounded by $\binom{|R|}{k} \leq (ek\epsilon^{-2})^k$. Thus the running time is bounded by $\mathrm{poly}(n, k^k, \epsilon^{-k})$. □

6. FURTHER EXTENSIONS

Our results show that for the uniform distribution without queries, agnostically learning sparse polynomials reduces to learning parities with random noise aka the noisy parity problem. This generalizes a result of [6] who showed such a reduction in the random noise setting.

A natural question to ask is whether one can come with an agnostic analog of Jackson's celebrated algorithm for DNFs [11]. An agnostic learner for DNFs will give a weak learner for depth-3 AC^0 circuits with queries under the uniform distribution in the noiseless setting; there is no known polynomial time algorithm for this problem. We note that Algorithm 2 will not solve this problem; one can construct a function f where Algorithm 2 gives a hypothesis with accuracy 1/2, whereas there is a polynomial size DNF formula with correlation at least $1/\mathrm{poly}(n)$ with f.

Is it possible to agnostically learn decision trees with queries in polynomial time for other distributions? Decision trees are known to be learnable in polynomial time in the exact model of learning with membership and equivalence queries [3, 1]. It would be interesting to find analogues of these algorithms for the agnostic setting.

7. REFERENCES

[1] A. BEIMEL, F. BERGADANO, N. H. BSHOUTY, E. KUSHILEVITZ, AND S. VARRICCHIO, *Learning functions represented as multiplicity automata*, J. ACM, 47 (2000), pp. 506–530.

[2] L. BREIMAN, J. FRIEDMAN, R. OLSHEN, AND C. STONE, *Classification of Regression Trees*, Wadsworth, 1984.

[3] N. H. BSHOUTY, *The monotone theory for the PAC-model*, Inf. Comput, 186(1) (2003), pp. 20–35.

[4] R. CARUANA AND A. NICULESCU-MIZIL, *An empirical comparison of supervised learning algorithms*, in Proc. 23^{rd} Intl. Conf. Machine learning (ICML'06), 2006, pp. 161–168.

[5] A. EHRENFEUCHT AND D. HAUSSLER, *Learning decision trees from random examples*, Information and Computation, 82 (1989), pp. 231–246.

[6] V. FELDMAN, P. GOPALAN, S. KHOT, AND A. K. PONNUSWAMI, *New results for learning noisy parities and halfspaces*, in Proc. 47^{th} IEEE Symp. on Foundations of Computer Science (FOCS'06), 2006.

[7] A. FLAXMAN, A. T. KALAI, AND H. B. MCMAHAN, *Online convex optimization in the bandit setting: gradient descent without a gradient*, in ACM Symposium on Discrete Algorithms (SODA'05), 2005, pp. 385–394.

[8] A. C. GILBERT, S. GUHA, P. INDYK, S. MUTHUKRISHNAN, AND M. STRAUSS, *Near-optimal sparse Fourier representations via sampling*, in Proc. 34^{th} Ann. ACM Symp. on Theory of Computing (STOC'02), 2002, pp. 152–161.

[9] A. C. GILBERT, M. J. STRAUSS, J. A. TROPP, AND R. VERSHYNIN, *One sketch for all: Fast algorithms for compressed sensing*, in Proc. 39^{th} ACM Symposium on the Theory of Computing (STOC'07), 2007.

[10] O. GOLDREICH AND L. LEVIN, *A hard-core predicate for all one-way functions.*, in Proc. 21^{st} ACM Symp. on the Theory of Computing (STOC'89), 1989, pp. 25–32.

[11] J. JACKSON, *The Harmonic sieve: a novel application of Fourier analysis to machine learning theory and practice*, PhD thesis, Carnegie Mellon University, August 1995.

[12] J. C. JACKSON, *Uniform-distribution learnability of noisy linear threshold functions with restricted focus of attention*, in Proc. Conf. on Learning Theory (COLT'06), 2006, pp. 304–318.

[13] A. T. Kalai, A. R. Klivans, Y. Mansour, and R. Servedio, *Agnostically learning halfspaces*, in Proc. 46^{th} IEEE Symp. on Foundations of Computer Science (FOCS'05), 2005.

[14] M. Kearns, R. Schapire, and L. Sellie, *Toward Efficient Agnostic Learning*, Machine Learning, 17 (1994), pp. 115–141.

[15] E. Kushilevitz and Y. Mansour, *Learning decision trees using the Fourier spectrum*, SIAM Journal of Computing, 22(6) (1993), pp. 1331–1348.

[16] N. Linial, Y. Mansour, and N. Nisan, *Constant depth circuits, Fourier transform and learnability*, Journal of the ACM, 40 (1993), pp. 607–620.

[17] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1992.

[18] J. B. Rosen, *The gradient projection method for nonlinear programming. part i. linear constraints*, Journal of the Society for Industrial and Applied Mathematics, 8 (1960), pp. 181–217.

[19] L. Valiant, *A theory of the learnable*, Communications of the ACM, 27 (1984), pp. 1134–1142.

[20] M. Zinkevich, *Online convex programming and generalized infinitesimal gradient ascent*, in Proc. 20^{th} Intl. Conf. on Machine Learning (ICML'03), 2003, pp. 928–936.

APPENDIX

A. AGNOSTIC LEARNING VIA ℓ_1 MINIMIZATION

In this section, we show how to use the algorithm for solving the ℓ_1 minimization problem described in the introduction for agnostic learning. We first give an algorithm for agnostic learning assuming that the examples given to the learner are labeled by an arbitrary deterministic function. We then show how to extend this solution to the full agnostic setting where the learner receives examples labeled according to an arbitrary *distribution* on $\{-1,1\}^n \times \{-1,1\}$.

Recall the definition of the sparse ℓ_1 regression problem:

Definition 3. *The sparse ℓ_1 regression problem takes as input an oracle encoding an arbitrary (deterministic) function $f : \{-1,1\}^n \to [-1,1]$, parameters t and ϵ, and with high probability outputs a polynomial P such that $L_1(P) \leq t$ and $\mathbb{E}_{x\in\mathcal{D}}[|P(x) - f(x)|] \leq \min_{Q,L_1(Q)\leq t} \mathbb{E}_{x\in\mathcal{D}}[|Q(x) - f(x)|] + \epsilon$.*

Theorem 16. *Let $\mathcal{C}$ be a concept class such that for every $c \in \mathcal{C}$, there exists a Real polynomial $p(x)$ such that $L_1(p) \leq t$ and $\mathbb{E}_{x\in\mathcal{D}}[|p(x) - c(x)|] \leq 2\epsilon/3$. Let A be an algorithm that solves the sparse ℓ_1 regression problem with respect to $\mathcal{D}$ in time $r(t, 1/\epsilon, n)$. Then there exists an algorithm for agnostically learning $\mathcal{C}$ that runs in time polynomial in r.*

Let $\mathrm{I}(A)$ be the function that is 1 if predicate A holds and 0 otherwise. To prove Theorem 16, we need the following lemma, which is implicit in the analysis of the main theorem in Kalai et al. [13]. It gives a way to convert from $P : \{-1,1\}^n \to \mathbb{R}$ which is close to f in terms of $\mathbb{E}_x[|P(x) - f(x)|]$ to binary $h : \{-1,1\}^n \to \{-1,1\}$ with low $\Pr[h(x) \neq f(x)]$. One takes $h(x) = \mathrm{I}(x \geq \theta)$, where θ is chosen by picking a number of random labeled examples $(x_1, f(x_1)), \ldots, (x_m, f(x_m))$ and then solves $\min_{\theta\in[0,1]} |\{i \in [m] \mid f(x_i) \neq \mathrm{I}(P(x_i) \geq \theta)\}|$. This minimization can be performed in time $O(m \log m)$ by first sorting the examples based on $P(x_i)$.

Lemma 17. *For any functions $f : \{-1,1\}^n \to \{-1,1\}$, $P : \{-1,1\}^n \to \mathbb{R}$, and parameters $\epsilon, \delta > 0$, take $m = O(\epsilon^{-2}\log(1/\delta\epsilon))$ uniformly random examples from $\{-1,1\}^n$ and let $\theta \in [-1,1]$ be such that it minimizes the number of disagreements between $f(x)$ and $h(x) = \mathrm{I}(P(x) \geq \theta)$ on the m examples. Then, with probability $1 - \delta$ over the m examples, the resulting h satisfies,*

$$\Pr_x[h(x) \neq f(x)] \leq \frac{\mathbb{E}_x[|P(x) - f(x)|]}{2} + \epsilon.$$

Proof: The proof follows that of Theorem 5 of [13]. Let $\mathcal{Z} = \langle x_1, \ldots, x_m\rangle \in \{-1,1\}^{n\times m}$ be the uniformly random examples chosen, and $\theta \in [-1,1]$ minimize the *empirical error* $\mathrm{err}_{\mathcal{Z}}(h) = \frac{1}{m}|\{i \in [m] \mid f(x_i) \neq \mathrm{I}(P(x_i) \geq \theta)\}|$. Let $x' = P(x)$ and $y' = f(x)$. The uniform distribution on x induces a distribution on $(x', y') \in \mathbb{R} \times \{-1,1\}$, to which we can apply VC theory. Since the VC dimension of thresholds on the line is 1, and the hypothesis h, viewed in terms of x', is simply a threshold, for $m = O(\log(\delta^{-1}\epsilon^{-1})\epsilon^{-2})$,

$$\Pr_{\mathcal{Z}}[\mathrm{err}(h) \geq \mathrm{err}_{\mathcal{Z}}(h) + \epsilon/2] \leq \delta/2.$$

Hence, it suffices to show that with probability $\geq 1 - \delta/2$, $\mathrm{err}_{\mathcal{Z}}(h) \leq \frac{\mathbb{E}[|P(x)-f(x)|]}{2} + \epsilon/2$.

Next we make the following claim about the empirical error $\mathrm{err}_{\mathcal{Z}}(h)$, for all $\mathcal{Z} \in \{-1,1\}^{n\times m}$

$$\frac{1}{m}|\{i \in [m] \mid f(x_i) \neq \mathrm{I}(P(x_i) \geq \theta)\}| \leq \frac{1}{m}\sum_{i=1}^{m} \min\Big\{\frac{|P(x_i) - f(x_i)|}{2}, 1\Big\}.$$

Note that in the above, θ is a function of $\mathcal{Z}$. To see why the above inequality holds, suppose for a moment that instead we had independently chosen θ uniformly at random in $[-1,1]$. Then, $\forall\ x \in \{-1,1\}^n$ $\Pr_{\theta\in[0,1]}[\mathrm{I}(P(x) \geq \theta) \neq f(x)] \leq \min\{\frac{|P(x)-f(x)|}{2}, 1\}$. The reason is that $\mathrm{I}(P(x_i) \geq \theta) \neq f(x)$ if and only if θ lies between $P(x_i)$ and $f(x_i)$. In other words, let $A = [\min\{P(x), f(x)\}, \max\{P(x), f(x)\}]$. Then $h(x) \neq f(x)$ iff $\theta \in A$. Since θ is uniform over $[-1,1]$, this happens with probability equal to the width of $A \cap [-1,1]$, which is upper-bounded by both $\frac{|P(x)-f(x)|}{2}$ and 1. Thus the expectation of the left hand side of the inequality, over random θ, is less than the right hand side. Hence, the inequality holds for the θ chosen to minimize empirical error as well.

Next, by Chernoff bounds, for $m = O(\log(\delta^{-1}\epsilon^{-1})\epsilon^{-2})$, with probability $\geq 1 - \delta/2$, $\frac{1}{m}\sum_{i=1}^{m} \min\{\frac{|P(x_i)-f(x_i)|}{2}, 1\} \leq \mathbb{E}_x\left[\min\{\frac{|P(x)-f(x)|}{2}, 1\}\right] + \epsilon/2 \leq \frac{\mathbb{E}_x[|P(x)-f(x)|]}{2} + \epsilon/2$. Combining this with (8), gives that with probability $\geq 1 - \delta/2$, $\mathrm{err}_{\mathcal{Z}}(h) \leq \frac{\mathbb{E}[|P(x)-f(x)|]}{2} + \epsilon/2$, which completes the proof. □

Proof Proof of Theorem 16. Fix an arbitrary function $f(x)$ and let $c(x)$ be the optimal concept for $f(x)$ with respect to $\mathcal{D}$. Let $P(x)$ be the polynomial output by running A with oracle access to $f(x)$ and parameters t and $\epsilon/3$. Take m new examples $(x^1, f(x)^1), \ldots, (x^m, f(x^m))$ chosen

at random from $\{0,1\}^n$ according to $\mathcal{D}$ (m will be chosen later). Compute θ to minimize the classification error of the function $h(x) = \text{sgn}(P(x)-\theta)$ with respect to the m new examples (there are only $m+1$ choices for θ). We now analyze the error of $h(x)$, our output hypothesis.

Let p^* be the polynomial such that $\mathbb{E}_x[|p^*(x) - c(x)|] \leq 2\epsilon/3$. Since $P(x)$ is a solution to the sparse ℓ_1 regression problem, we have with high probability that $\mathbb{E}_x[|P(x) - f(x)|] \leq \mathbb{E}_x[|p^*(x) - f(x)|] + \epsilon/3$. Choosing m sufficiently large according to Lemma 17 and applying the triangle inequality we have

$$\Pr_x[h(x) \neq f(x)] \leq \frac{\mathbb{E}_x[|p^*(x) - c(x)|]}{2} + \frac{\mathbb{E}_x[|c(x) - f(x)|]}{2}$$

$$+2\epsilon/3.$$

Recall that $\mathbb{E}_x[|c(x) - f(x)|]$ is precisely $2\mathsf{opt}$. Therefore, by setting $m = O(\log(1/\delta\epsilon)/\epsilon^2)$, with probability at least $1-\delta$, the classification error of h is at most $\mathsf{opt} + \epsilon$. □

As stated earlier, Theorem 16 holds in the setting where the learner has access to examples labeled by a fixed deterministic function. We can remove this restriction and learn in the "true" agnostic setting where the learner receives examples labeled according to an arbitrary distribution on $\{-1,1\}^n \times \{-1,1\}$ whose marginal on $\{-1,1\}^n$ is uniform. We defer the details of this reduction to Appendix A.1.

It is well known that for every decision tree T, there exists a polynomial p computing T with $L_1(p)$ equal to the number of leaves of T [15]. Combining Theorem 16 with the polynomial-time solution to the sparse ℓ_1 regression problem from Section 4 we obtain our main result:

THEOREM 18. *The class of polynomial-size decision trees can be agnostically learned (using queries) to accuracy ϵ in time $\text{poly}(n, 1/\epsilon)$ with respect to the uniform distribution.*

A.1 From Fixed Functions to Distributions

In Section A, we described an algorithm for agnostically learning decision trees in a setting where the learner has access to an arbitrary but fixed deterministic functions. In this section, we give a more general agnostic learner that works with respect to distributions on $\{-1,1\}^n \times \{-1,1\}$ whose marginal distribution on $\{-1,1\}^n$ is uniform. The statements in this subsection may be folklore, but we have been unable to find a proof.

Define

$$\mathsf{opt} = \arg\min_{c \in \mathcal{C}} \Pr_{x,y \in D}[c(x) \neq y].$$

For a hypothesis h and a function f let $\text{err}_f(h) = \Pr_x[h(x) \neq f(x)]$ and for a distribution D let $\text{err}_D(h) = \Pr_{x,y \in D}[h(x) \neq y]$.

Fix D on $\{-1,1\}^n \times \{-1,1\}$. Define the distribution $F(D)$ over deterministic functions $f : \{-1,1\}^n \to \{-1,1\}$ to be the distribution where, for each $x \in \{-1,1\}^n$ *independently*, $f(x)$ is chosen to be -1 or 1 with probabilities $P_D[y = 1|x]$ and $P_D[y = -1|x]$. Running an algorithm A with oracle access to D is equivalent[1] to running A with oracle access to a random f chosen according to $F(D)$.

[1]Note that we need that A never queries the same x twice, which is a reasonable requirement since A is designed for a deterministic query function.

We will need the following lemma relating $\text{err}_f(h)$ for a randomly chosen f and $\text{err}_D(h)$:

LEMMA 19. *Let A^f be any algorithm that makes q distinct queries to some fixed function $f : \{-1,1\}^n \to \{-1,1\}$ and outputs $h : \{-1,1\}^n \to \{-1,1\}$. (If A is randomized, we may think of it as being deterministic by fixing its random bits). Imagine running A after choosing f according to $F(D)$. Then*

$$\mathbb{E}_{f \leftarrow F(D)}[\text{err}_D(h)] \leq \mathbb{E}_{f \leftarrow F(D)}[\text{err}_f(h)] + q2^{-n}.$$

PROOF. Let f be drawn according to $F(D)$ and let the points queried by A be $Q \subseteq \{-1,1\}^n$, with $|Q| = q$. Let h be the resulting hypothesis. Then we have

$$\text{err}_D(h) = \Pr_x[h(x) \neq y] = \mathbb{E}_{f' \leftarrow F(D)}[\Pr_x[h(x) \neq f'(x)]],$$

where f' is an independent function chosen according to $F(D)$. How do the quantities $\text{err}_f(h)$ and $\text{err}_{f'}(h)$ differ, *in expectation*? Note that h was constructed only based on the results of the queries to q different x's. On the remaining $2^n - q$ points, f and f' are identically distributed (and unknown). Hence, in expectation, the two quantities differ by at most $q2^{-n}$. □

THEOREM 20. *Let $\mathcal{C}$ be a concept class of $c : \{-1,1\}^n \to \{-1,1\}$ and let $q > 0$. Suppose that algorithm $A^f(r, \epsilon, \delta)$ (using random bits r) has the following property: for any $\epsilon, \delta > 0$, for any $c \in \mathcal{C}$, $f : \{-1,1\}^n \to \{-1,1\}$, A makes polynomially many queries to f, outputting $h : \{-1,1\}^n \to \{-1,1\}$ such that*

$$\mathbb{E}_r[\text{err}_f(h)] \leq \min_{c \in C} \text{err}_f(c) + \epsilon.$$

Then there is an algorithm that has the following guarantee for any distribution D over $\{-1,1\}^n \times \{-1,1\}$, when the oracle to f is replaced by an oracle to D: A makes polynomially many queries to D and outputs $h : \{-1,1\}^n \to \{-1,1\}$ such that, with probability $\geq 1-\delta$,

$$\mathbb{E}_{f \leftarrow F(D),r}[\text{err}_D(h)] \leq \min_{c \in C} \text{err}_D(c) + \epsilon.$$

PROOF. First, we claim that,

$$\mathbb{E}_{f \leftarrow F(D)}\left[\min_{c \in \mathcal{C}} \text{err}_f(c)\right] \leq \min_{c \in \mathcal{C}} \text{err}_D(c).$$

To see this, let $c^* \in \mathcal{C}$ be a function such that $\text{err}_D(c^*) = \min_{c \in \mathcal{C}} \text{err}_D(c)$. Then note that

$$\mathbb{E}_{f \leftarrow F(D)}\left[\min_{c \in \mathcal{C}} \text{err}_f(c)\right] \leq \mathbb{E}_{f \leftarrow F(D)}[\text{err}_f(c^*)] = \text{err}_D(c^*).$$

Next note that since with probability $1-\delta$, A is within ϵ of opt, we have the following:

$$\mathbb{E}_r[\text{err}_f(h)] \leq \text{err}_D(c^*) + \epsilon + \delta.$$

By the previous lemma, we have,

$$\mathbb{E}_{f \leftarrow F(D),r}[\text{err}_D(h)] \leq \text{err}_D(c^*) + \epsilon + \delta + q2^{-n}.$$

It is straightforward to transform this into a learning algorithm that achieves error $\mathsf{opt} + \epsilon$ by repeatedly running the algorithm several times on independent test sets and choosing the hypothesis with lowest error.

□

B. RELATION TO COMPRESSED SENSING

While problems of similar flavor have been investigated in the compressed sensing literature, there seem to be some important differences between their settings and ours. A typical CS scenario is one where a learner is allowed to make measurements of a signal f with $N = 2^n$ entries and asked to find the "best" sparse representation P of this signal with respect to the ℓ_1 or ℓ_2 norm. The best sparse approximation is obtained by taking the largest co-ordinates of the signal f. There are numerous algorithm that give strong guarantees in this setting (see for instance [9]).

In our setting, let $\hat{f} = \{\hat{f}(S)\}_{S \subseteq n}$ denote P written in the Fourier basis, while $f = \{f(x)\}_{x \in \{-1,1\}^n}$ denotes f written pointwise. Our goal is to find a good approximation P which minimizes the ℓ_1 distance $\mathbb{E}_x[|P(x) - f(x)|]$ in the function domain but which is sparse in the Fourier domain. It is no longer clear that picking the largest Fourier coefficients of f gives the best approximation. In contrast, if we were working with respect to the ℓ_2 norm, then by Parseval's identity, this would be the best sparse approximation.

Another difference in the settings is that we only have black-box access to $f(x)$, hence we need to design a sampling algorithm that makes *local* measurements involving only a few co-ordinates of a 2^n dimensional vector. In contrast, locality is usually not generally required in the CS scenario. The sparse Fourier sampling algorithm of [8], which gives a generalization of KM over other domains is local, but it minimizes the ℓ_2 error. We are unaware of an analogous algorithm in the CS literature for ℓ_1 error.