Computational Cost of Achieving Adversarial Robustness for PTFs

author names withheld

Editor: Under Review for COLT 2019

Abstract

We study the design of computationally efficient algorithms that are robust to adversarial or test time perturbations. A robust learning algorithm must satisfy two properties: a) it must achieve low test error in the standard sense, and b) its predictions must be robust to perturbations of some magnitude on most of the data distribution. While there are many motivations for studying the design of robust learning algorithms, there has been an explosion of recent work on this topic due to its connections with test time robustness of deep networks. Existing theoretical works on adversarial robustness have focused on establishing generalization bounds, test-error vs robustness tradeoffs, or algorithms for certifying robustness around a given point. Another aspect about which we currently have minimal understanding is: what is the price of achieving robustness in a computationally efficient manner? In this work, we take first steps towards answering this question for a large class of function classes, namely linear classifiers and degree-2 polynomial threshold functions (PTFs).

Focusing on the realizable case, where there exists a function in the class that labels the data perfectly with a radius of robustness of \( \delta \), we ask if one can find in polynomial time a function in the class (potentially a different one) that has low error and has a radius of robustness at least \( \delta/\gamma \). Here, \( \gamma \geq 1 \) characterizes the price of achieving robustness in polynomial time. In particular, if \( \gamma = 1 \), then the learning algorithm is optimally robust. Our first result shows that it is computationally hard to design a learning algorithm for degree-2 PTFs that is optimally robust. On the other hand, for degree-1 PTFs (linear classifiers) we give a polynomial time learning algorithm that is optimally robust, and also design a polynomial time algorithm that achieves \( \gamma = O(\sqrt{\log n}) \) for degree-2 PTFs. Our algorithmic results are established via a general framework that relates polynomial time robust learnability of PTFs to a rich class of polynomial optimization problems. Our results together draw a sharp contrast with the standard non-robust setting where constant degree PTFs can be learned in polynomial time, thereby demonstrating the computational price of robustness for learning PTFs.

Keywords: Adversarial Robustness, PAC Learning, Computational complexity.

1. Introduction

The empirical success of deep learning has led to numerous questions and unexplained phenomena about which our current theoretical understanding is limited. Examples include the ability of complex models to generalize well and effectiveness of first order methods on optimizing training loss. The focus of this paper is on the phenomenon of adversarial robustness, that was first pointed out by Szegedy et al. (2013). On many benchmark data sets, deep networks optimized on the training set can often be fooled into misclassifying a test example by making a small adversarial perturbation that is imperceptible to a human labeler. This has led to a proliferation of work on designing robust algorithms that defend against such adversarial perturbations, as well as attacks that aim to break these defenses. These works span many flavors of adversarial robustness, each differing in what constitutes a valid perturbed example, how much information the attacker has about the model, and what the success metric is (Gilmer et al., 2018a).
In this work we choose to focus on perturbation defense, the most widely studied formulation of adversarial robustness. The current empirical state of the art solution for perturbation defense proposes to train a robust loss function in order to defend against adversarial attacks (Madry et al., 2017), although it has also been shown to be vulnerable to certain kinds of attacks (Sharma and Chen, 2017; Zhang et al., 2019). In the perturbation defense model, given a classifier $f$, an adversary can take a test example $x$ generated from the data distribution and perturb it to $\tilde{x}$ such that $\|x - \tilde{x}\| \leq \delta$. Here $\delta$ characterizes the amount of power the adversary has and the distance is usually measured in the $\|\cdot\|_\infty$ norm (other norms that have been studied include the $\ell_2$ norm). Given a loss function $\ell(\cdot)$, the goal then is to optimize the robust loss defined as

$$E_{(x,y)\sim D} \left[ \max_{\tilde{x} : \|x - \tilde{x}\|_\infty \leq \delta} \ell(f(\tilde{x}), y) \right].$$

(1)

It is reasonable to expect that if $\delta$ is small, the label $y$ of an example does not change and hence the robust loss is a good objective to optimize. The study of robust optimization problems, such as equation (1), dates far back (El Ghaoui and Lebret, 1997; Ben-Tal and Nemirovski, 1999; Bertsimas and Sim, 2004) and precedes the motivations stemming from deep learning research. Even in the context of machine learning, there are other reasons to optimize (1) such as improving out of distribution generalization, dealing with missing feature values and gaining a different perspective on standard regularization algorithms (Bhattacharyya, 2004; Globerson and Roweis, 2006; Shivaswamy et al., 2006; Xu et al., 2009; Xu and Mannor, 2012).

There has been a recent surge in efforts to theoretically understand the perturbation defense model. The work of Xu et al. (2009); Xu and Mannor (2012) shows that for linear function classes, optimizing (1) is equivalent to optimizing a regularized loss function. A recent line of work aims to characterize the generalization properties of robust loss minimization, by either bounding the Rademacher complexity of a robust loss function class (Yin et al., 2018; Khim and Loh, 2018), characterizing generalization for specific settings (Schmidt et al., 2018; Feige et al., 2015; Attias et al., 2018), or using combinatorial notions analogous to VC dimension (Cullina et al., 2018). Another line of work studies the trade-off between traditional test error and robust error as defined in Equation 1 (Gilmer et al., 2018b; Tsipras et al., 2017; Mahloujifar et al., 2018; Mahloujifar and Mahmood, 2018; Diochnos et al., 2018). These works ignore one important consideration, namely how much robustness, if any, can one achieve if constrained to polynomial time learning algorithms. This is precisely the question we study in this paper:

**What is the price of achieving robustness in a computationally efficient manner?**

In order to answer the above question, we study a natural model of robust learning and characterize the price of achieving robustness in polynomial time for a large family of function classes, namely degree-1 and degree-2 polynomial threshold functions (PTFs). We now summarize our main results.

**Our Contributions**

**Model.** We define a natural learning model that incorporates robustness. Our definition, that has similarities to others that have been studied recently (Cullina et al., 2018; Bubeck et al., 2018a,b), introduces an additional parameter $\gamma$, that helps clarify the tradeoff when computational efficiency is desired. We focus on the 0/1 error and say that a sub-class $F$ of PTFs of VC dimension $\Delta$ is $\gamma$-approximately robustly learnable if there exists a randomized algorithm that, for any given $\epsilon, \delta > 0$, takes as input $\text{poly}(\Delta, \frac{1}{\epsilon})$ examples generated from a distribution and labeled by a function in $F$ that
Aversarially ROBUST LEARNING OF PTFs

has zero $\delta$-robust error (realizable case), outputs a classifier from $\mathcal{F}$ that has $\delta/\gamma$-robust error at most $\varepsilon$. Furthermore, the algorithm should run in polynomial time. See Section 2 for the formal definition.

We are interested in characterizing how large $\gamma$ should be in order to achieve $\varepsilon$-robust error in polynomial time. In fact, this was posed as an open problem in Bubeck et al. (2018b). We give both lower bounds and upper bounds towards characterizing this price of robustness for degree-1 and degree-2 PTFs.

Computational Hardness. For degree-2 and higher PTFs, we show that one indeed has to pay a price for computational robustness. We establish this by proving that optimally robust learning of degree-2 PTFs is computationally hard, assuming $NP$ does not have probabilistic bounded error polynomial time algorithms (see Theorem 14 and Corollary 15 for formal statements). This is in sharp contrast to the non-robust setting ($\delta = 0$), where there are polynomial time algorithms for constant degree PTFs (in the literature this is referred to as proper PAC learning in the realizable setting). In fact, we demonstrate a stronger hardness result that proves that for learning degree-2 and higher PTFs without any loss in the robustness parameter, i.e, $\gamma = 1$, it is computationally hard to even find a classifier of any constant error in the range $(0, \frac{1}{4})$.

Algorithms. We design $\gamma$-approximately robust learning algorithms for degree-1 and degree-2 PTFs over $n$ variables, with $\gamma = 1$ and $\gamma = O(\sqrt{\log n})$ respectively. Our algorithms are based on semidefinite relaxations of a natural class of polynomial optimization problems that stem from a general framework, that we show captures robust learnability of PTFs. These problems are also closely related to a rich class of well-studied problems that include the Grothendieck problem and its generalizations (Alon and Naor, 2004; Charikar and Wirth, 2004; Alon et al., 2006; Khot and Naor, 2007).

Comparison to Related Work. There have been several recent and concurrent works on this topic. The most relevant to our result is the cryptographic lower bound of Bubeck et al. (2018a,b) who design a computational task in $\mathbb{R}^n$ that is robustly learnable using $\text{poly}(n)$ samples to any given robustness parameter $M$, but is hard to learn robustly to any non-trivial robustness parameter $\varepsilon > 0$, in polynomial time. When translated to our model, this provides an instance of a cryptographic learning task that is computationally hard to $\gamma$-approximately robustly learn for any constant $\gamma \geq 1$.

However, this does not rule out the possibility that natural function classes can be robustly learned without any loss in robustness parameter. Our result rules this out for the class of degree-2 and higher PTFs, even in the realizable setting, i.e., when there exists a robust classifier of zero error! Finally, to the best of our knowledge, our upper bounds are the first to establish the robustness tradeoff for computationally efficient learning for a large natural class of functions.

We defer other related work to Section A. In the rest of the paper, we first define our model formally and give an overview of our techniques in Section 2. We then describe the algorithmic framework and algorithmic guarantees in Sections 3 and 4. The computational intractability result is in Section 5, followed by open questions in Section 6.

2. Model and Preliminaries

We focus on binary classification, and adversarial perturbations are measured in $\ell_\infty$ norm. Recall that for any vector $x \in \mathbb{R}^n$, we have $\|x\|_\infty = \max_i |x_i|$. Given $x \in \mathbb{R}^n$, we study robust learning of polynomial threshold functions (PTFs). These are functions of the form $\text{sgn}(p(x))$, where $p(x)$ is a polynomial in $n$ variables over the reals. Here $\text{sgn}(t)$ equals $+1$, if $t \geq 0$ and $-1$ otherwise. We are
interested in the 0/1 loss. Given \( y, y' \in \{-1, 1\} \), the 0/1 loss is defined as \( \ell(y, y') = 1 \) if \( y \neq y' \) and 0 otherwise. We now define the notion of robust error of a classifier.

**Definition 1 (\( \delta \)-robust error)** Let \( f(x) \) be a Boolean function mapping \( \mathbb{R}^n \) to \( \{-1, 1\} \). Let \( D \) be a distribution over \( \mathbb{R}^n \times \{-1, 1\} \). Given \( \delta > 0 \), we define the \( \delta \)-robust error of \( f \) with respect to \( D \) as

\[
err_{\delta,D}(f) = \mathbb{E}_{(x,y) \sim D} \left[ \sup_{z \in B^n_\infty(0,\delta)} \ell(f(x+z),y) \right].
\]

Here \( B^n_\infty(0,\delta) \) denotes the \( \ell_\infty \) ball of radius \( \delta \), i.e., \( B^n_\infty(0,\delta) = \{ x \in \mathbb{R}^n : \|x\|_\infty \leq \delta \} \).

Notice that the case \( \delta = 0 \) corresponds to the standard definition of test error. Analogous to the notion of empirical error in PAC learning, we next define the notion of robust empirical error.

**Definition 2 (\( \delta \)-robust empirical error)** Let \( f(x) \) be a Boolean function mapping \( \mathbb{R}^n \) to \( \{-1, 1\} \). Let \( S \) be a set of \( m \) examples \( (x_1, y_1), \ldots, (x_m, y_m) \) where \( x_i \in \mathbb{R}^n \) and \( y_i \in \{-1, 1\} \). Given \( \delta > 0 \), we define the \( \delta \)-robust empirical error of \( f \) on \( S \) as

\[
eerr_{\delta,S}(f) = \frac{1}{m} \sum_{(x,y) \sim S} \sup_{z \in B^n_\infty(0,\delta)} \ell(f(x+z),y).
\]

To bound generalization gap, we will use the notion of adversarial VC dimension as introduced in Cullina et al. (2018). Additionally, the authors show that for linear classifiers the adversarial VC dimension remains the same as that of the original class. The bound below then follows by viewing PTFs as linear classifiers in a higher dimensional space.

**Lemma 3** Let \( F \) be a class of degree-\( d \) polynomial threshold functions from \( \mathbb{R}^n \mapsto \{-1, 1\} \) of VC dimension \( \Delta = O(n^d) \). Given \( \delta, \eta > 0 \), and a set \( S \) of \( m \) examples \( (x_1, y_1), \ldots, (x_m, y_m) \) generated from a distribution \( D \) over \( \mathbb{R}^n \times \{-1, 1\} \), with probability at least \( 1 - \eta \), we have that

\[
\sup_{f \in F} |err_{\delta,D}(f) - eerr_{\delta,S}(f)| \leq 2 \sqrt{\frac{2\Delta \log m}{m}} + \sqrt{\frac{\log \frac{1}{\eta}}{2m}}.
\]

Next we define our model of robust learning aimed at quantifying the price of achieving robustness in polynomial time.

**Definition 4 (\( \gamma \)-approximately robust learning)** Let \( F \) be a sub-class of PTFs from \( \mathbb{R}^n \mapsto \{-1, 1\} \) of VC dimension \( \Delta \). For \( \gamma \geq 1 \), we say that an algorithm \( A \) \( \gamma \)-approximately robustly learns \( F \) if the following holds for any \( \varepsilon, \delta, \eta > 0 \): Given \( m = \text{poly}(\Delta, \frac{1}{\varepsilon}, \frac{1}{\eta}) \) samples from a distribution \( D \) over \( \mathbb{R}^n \times \{-1, 1\} \), if \( F \) contains a function \( f^* \) such that \( err_{\delta,D}(f^*) = 0 \), then with probability at least \( 1 - \eta \), \( A \) runs in time polynomial in \( m \) and outputs a Boolean function \( f \in F \) such that

\[
err_{\delta/\gamma,D}(f) \leq \varepsilon.
\]

If the class \( F \) admits such an algorithm then we say that \( F \) is \( \gamma \)-approximately robustly learnable. Furthermore, if the learning algorithm \( A \) achieves the above guarantee for \( \gamma = 1 \), then we say that it is an optimally robust learning algorithm.
The approximation factor $\gamma$ quantifies the price of achieving computationally efficient robust learning. Notice that if $\delta = 0$ then the above definition exactly corresponds to the realizable case of polynomial time proper PAC learning (Valiant, 1984). Hence, if robustness is not a concern ($\delta = 0$) then any class that is properly PAC learnable is also learnable in our model. It is also easy to see that if polynomial running time is not required then any sub-class of PTFs of finite VC dimension is optimally robustly learnable, i.e., with $\gamma = 1$. In order to do this one would minimize the $\delta$-robust empirical error as defined in 3. Our goal in the next two sections is to establish upper and lower bounds on the optimal $\gamma$ possible for polynomial time robust learnability of PTFs. Before that, we conclude the section with an overview of the techniques involved in obtaining our main results.

**Overview of Techniques.** *Algorithms.* To obtain our upper bounds, we design a general algorithmic framework that relates robust learnability of PTFs to a natural polynomial maximization problem. As an example consider the PTF $\text{sgn}(g(x))$ that belong to some class $\mathcal{F}$. We show that if one can efficiently find $\hat{x} \in B_{\mathcal{F}}(0, \gamma \delta)$ given any $g(x) \in \mathcal{F}$ and $\delta > 0$, such that $g(\hat{x}) \geq \max_{x \in B_{\mathcal{F}}(0, \delta)} g(x)$, then the corresponding class $\mathcal{F}$ is $\gamma$-approximately robust learnable. We prove this by setting up a constrained convex program to do robust empirical risk minimization. Furthermore, an efficient algorithm for computing the solution $\hat{x}$ to the above optimization problem can be used to design an approximate separation oracle for the Ellipsoid algorithm to solve the convex program.

Our optimization problem is reminiscent of classic polynomial optimization problems like Quadratic Programming (QP) that are well-studied in the literature (Nesterov, 1998; Charikar and Wirth, 2004; Alon et al., 2006; Khot and Naor, 2007). The goal in Quadratic Programming (QP) is to compute $\max_{x \in \{-1, 1\}^n} x^T A x$ where $A$ is a matrix with 0s on the diagonal $^1$. One challenge for our setup is that existing approximation algorithms for polynomial maximization mainly deal with homogenous polynomials (in fact when maximizing non-homogenous polynomials over $\{-1, 1\}^n$, algorithms with finite approximation factors are not possible unless $P = NP$). More importantly, traditional approximation algorithms (for problems like QP) approximate the maximum value of the polynomial while maintaining the constraints. In contrast, we need to approximate the optimal output (relaxing the constraint) while not losing in the objective value. While the algorithm for our optimization problem is inspired by the SDP-based algorithm for Quadratic Programming (Charikar and Wirth, 2004), the above challenges necessitate non-trivial modifications, both to the algorithm (SDP and rounding) and its analysis. In particular, our SDP and the rounding algorithm are designed to carefully preserve the contribution of the non-homogeneous terms in the objective.

**Computational Intractability.** Our main hardness result shows that for degree-2 and higher PTFs, some loss in robustness is necessary for polynomial time learnability. In particular, we prove that optimally robust learning of the class of degree-2 PTFs is hard unless $NP = RP$. The starting point of our reduction is the NP-hardness of the QP problem: given a polynomial of the form $x^T A x$, it is NP-hard to distinguish whether $\max_{\|x\|_\infty \leq 1} x^T A x < \beta$ or if $\max_{\|x\|_\infty \leq 1} x^T A x > \beta$ for some $\beta$.

---

1. When the diagonal entries are all 0, it is easy to see that $\max_{\|x\|_\infty \leq \delta} x^T A x = \delta^2 \max_{x \in \{-1, 1\}^n} x^T A x$. 

\( \beta > 0 \). It is easy to see that this immediately implies that checking \((\delta = 1)\)-robustness of a given PTF, say \( \text{sgn}(\beta - x^T A x) \) at \( 0 \) is NP-hard. However, this does not suffice for hardness of learning since there could be several other PTFs that are robust and achieve zero error (in fact, given a single labeled data point \((0,1)\), there is a trivial constant polynomial that achieves \(1\)-robust empirical error of \(0\))!

The main challenge in our reduction is to ensure that there are no spurious degree-2 PTFs that are consistent with the labels, and \(\delta\)-robust at all the given points. To this end, we introduce a gadget in \( \mathbb{R}^{n+1} \) (with an additional co-ordinate \( z \)) with \( m = O(n^2) \) additional points of the form \((x^{(j)}, z^{(j)}) \in \mathbb{R}^{n+1} \) where \( x^{(j)} \in \mathbb{R}^n \) is sampled randomly and \( z^{(j)} = (x^{(j)})^T A x^{(j)} \). We then perturb each such \((x^{(j)}, z^{(j)}) \) along a specifically chosen direction to get points \((u^{(j)}, z_u^{(j)}) \) and \((v^{(j)}, z_v^{(j)}) \), one labeled as a positive example and the other labeled as a negative example, so as to satisfy two properties (see Figure 2 for an illustration):

1. any degree-2 PTF that is \( \delta \)-robust at these \( 2m \) points must pass through \( \{ (x^{(j)}, z^{(j)}) : j \in [m] \} \),
2. our intended PTF, i.e., \( \text{sgn}(z - x^T A x) \) must be \(\delta\)-robust at these points. Properties (1) and (2) together will allow us to conclude that \( \text{sgn}(z - x^T A x) \) is the unique PTF that is \( \delta \)-robust at the \( 2m \) points. To rule out the existence of any other polynomial of form \( g(x, z) \), we use (1) to set up a system of equations of the form \( g(x^{(j)}, z^{(j)}) = 0 \), where the variables are the coefficients of the candidate degree-2 polynomial \( q \). We prove that the coefficient matrix of this system is rank deficient by exactly one, hence ruling out the existence of any other polynomial. Finally deciding whether the “intended” PTF \( \text{sgn}(z - x^T A x) \) is \( \delta \)-robust at the point \((0, \beta)\) will correspond to whether \( x^T A x \leq \beta \).

We also remark that our techniques are powerful enough to also rule out the existence of any other polynomial. Finally deciding whether the “intended” PTF \( \text{sgn}(z - x^T A x) \) is \( \delta \)-robust at the point \((0, \beta)\) will correspond to whether \( x^T A x \leq \beta \).

We then perturb each such \((x^{(j)}, z^{(j)}) \) along a specifically chosen direction to get points \((u^{(j)}, z_u^{(j)}) \) and \((v^{(j)}, z_v^{(j)}) \), one labeled as a positive example and the other labeled as a negative example, so as to satisfy two properties (see Figure 2 for an illustration):

1. any degree-2 PTF that is \( \delta \)-robust at these \( 2m \) points must pass through \( \{ (x^{(j)}, z^{(j)}) : j \in [m] \} \),
2. our intended PTF, i.e., \( \text{sgn}(z - x^T A x) \) must be \(\delta\)-robust at these points. Properties (1) and (2) together will allow us to conclude that \( \text{sgn}(z - x^T A x) \) is the unique PTF that is \( \delta \)-robust at the \( 2m \) points. To rule out the existence of any other polynomial of form \( g(x, z) \), we use (1) to set up a system of equations of the form \( g(x^{(j)}, z^{(j)}) = 0 \), where the variables are the coefficients of the candidate degree-2 polynomial \( q \). We prove that the coefficient matrix of this system is rank deficient by exactly one, hence ruling out the existence of any other polynomial. Finally deciding whether the “intended” PTF \( \text{sgn}(z - x^T A x) \) is \( \delta \)-robust at the point \((0, \beta)\) will correspond to whether \( x^T A x \leq \beta \).

We also remark that our techniques are powerful enough to also rule out the existence of any degree-2 PTF that achieves a \( \delta \)-robust error of at most \(1/4\). In the following sections we detail our results and provide formal proofs.

3. Efficient Algorithms for Robust Learning of PTFs

In this section we design polynomial time robust learning algorithms for various sub-classes of PTFs. In particular, these include general degree-1 and degree-2 polynomial threshold functions. We obtain our upper bounds by establishing a general algorithmic framework that relates robust learnability of PTFs to the following natural polynomial maximization problem: given a polynomial \( g(x) \) such that \( \max_{x \in B_\infty^n(0, \delta)} g(x) = v^* \), can one output in polynomial time, a point \( \hat{x} \) in a larger \( \| \|_\infty \) ball such that \( g(\hat{x}) \geq v^* \). This is formalized in the definition below:

**Definition 6 (\( \gamma \)-factor admissibility)** For \( \gamma \geq 1 \), we say that a sub-class \( \mathcal{F} \) of PTFs is \( \gamma \)-factor admissible if \( \mathcal{F} \) has the following properties:

1. For any \( a \in \mathbb{R} \) and \( \text{sgn}(g(x)) \in \mathcal{F} \), we have that \( \text{sgn}(a + g(x)) \in \mathcal{F} \), \( \text{sgn}(ag(x)) \in \mathcal{F} \).
2. For any \( b \in \mathbb{R}^n \) and \( \text{sgn}(g(x)) \in \mathcal{F} \), we have that \( \text{sgn}(b - g(x + b)) \in \mathcal{F} \).
3. There is an algorithm that, given as input \( g(x) \) such that \( \text{sgn}(g(x)) \in \mathcal{F} \), and \( \eta, \delta > 0 \), runs in time \( \text{poly}(n, \log(\frac{1}{\delta})) \), and with probability at least \( 1 - \eta \), outputs a point \( \hat{x} \in B_\infty^n(0, \gamma \delta) \) such that \( g(\hat{x}) \geq \max_{x \in B_\infty^n(0, \delta)} g(x) \).

The first two conditions above are natural and are satisfied by many sub-classes of PTFs. The third condition in the above definition concerns an optimization problem that is closely related to the standard polynomial maximization problem where the goal is to obtain, in polynomial time, an objective value as close to the optimal one, without violating the \( \| \|_\infty \) ball constraint. Instead, our
1. Let $S = (x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$ be the given training set.

2. Find a degree polynomial $g \in F$ that satisfies

$$y_i g(x_i) > r_i, \quad \forall i \in [m]$$

(4)

$$r_i \geq \sup_{z \in B_{\infty}(0, \delta)} y_i \left( g(x_i) - g(x_i + z) \right), \quad \forall i \in [m]$$

(5)

Figure 1: The convex program for finding a polynomial $g \in F$ with zero robust empirical error.

The problem asks for the same objective value at the cost of an increase in the radius of the optimization ball. In approximation algorithms literature this will correspond to obtaining a $(1, \gamma)$-bicriteria approximation. The main result of this section, stated below, is the claim that any admissible sub-class of PTFs is also robustly learnable in polynomial time.

**Theorem 7** Let $F$ be a sub-class of PTFs that is $\gamma$-factor admissible for $\gamma \geq 1$. Then $F$ is $\gamma$-approximate robustly learnable.

**Remark 8** While we state our upper bounds for perturbations measured in the $\ell_\infty$ norm, we would like to point out that one can define analogously $\gamma$-factor admissibility for any $\ell_p$ norm and the above theorem will still hold true with the new definition.

To establish Theorem 7 we perform robust empirical risk minimization and use Lemma 3 to get robust learnability. This entails finding a polynomial $g \in F$ that correctly classifies all the training examples $(x_i, y_i)$. This corresponds to the constraint $y_i g(x_i) > 0$, a linear constraint in the unknown coefficients of the polynomial $g$. For example, if $g(x)$ is a degree-2 polynomial of the form $x^T A x + b^T x + c$, then the constraint $y_i g(x_i) > 0$ is linear in the unknown coefficients, $a_{i,j}, b_i$ and $c$, of the polynomial. Here $a_{i,j}$ corresponds to the $(i, j)$ entry of the matrix $A$ and $b_i$ is the $i$th coordinate of vector $b$. We also want to ensure that $g$ is robust around each point in the training set. These two constraints together can be enforced by the convex program in Figure 1, where the $r_i$’s are additional variables apart from the coefficients of $g$. While constraints in Equation 4 are linear in the variables and easy to implement, Equation 5 is really asking to check the robustness of $g$ at a given point $(x_i, y_i)$, which is an NP-hard problem (Charikar and Wirth, 2004).

Instead, we will use the fact that $F$ is $\gamma$-factor admissible to design an approximate separation oracle for the type of constraints enforced in Equation 5. We would like to mention that the classical literature on robust optimization of linear and convex programs studies a similar setting where typically the goal is to bound the probability of each constraint being violated while achieving the maximum objective value (Ben-Tal and Nemirovski, 1999; El Ghaoui and Lebret, 1997; Bertsimas and Sim, 2004). In contrast, we are interested in precisely quantifying how much a constraint can be violated by and relate the bound to the robustness of the final classifier obtained. We are now ready to prove the main theorem of this section.

**Proof [Proof of Theorem 7]** Let $\eta > 0$ be the success probability desired for the robust learning algorithm and $\varepsilon > 0$ be the final robust error that is desired. Let $B$ be an algorithm that achieves the $\gamma$-factor admissibility for the class $F$. Given $S$, we will run the Ellipsoid algorithm on the convex program in Figure 1. Let $T(m, n)$ be a (polynomial) upper bound on the number of iterations of the algorithm. In each iteration, given $g, r_1, r_2, \ldots, r_m$, we will first check whether $y_i g(x_i) > r_i$. If
not, then we have found a violated constraint with the corresponding separating hyperplane being 
\( \text{sgn}(r_i - y_i g(x_i)) \), and the algorithm proceeds to the next iteration. If all the constraints in Equation 4 are satisfied, then for each \( i \in [m] \), we run \( \mathcal{B} \) on the polynomial \( y_i (g(x_i) - g(x_i + z)) \), where \( z \) is the variable and \( x_i \) is fixed to be the \( i \)th data point. Furthermore, we will set \( \eta' \), the failure probability of \( \mathcal{B} \), to be equal to \( \eta/(mT(m, n)) \) and set \( \delta' \) that is input to \( \mathcal{B} \) to be \( \delta/\gamma \). From the guarantee of \( \mathcal{B} \) we get that if there exists an \( i \) such that

\[
 r_i < \sup_{z \in B_{\infty}^n(0, \delta)} y_i (g(x_i) - g(x_i + z)),
\]

with probability at least \( 1 - \eta/T(m, n) \), the \( \mathcal{B} \) will output a violated constraint of the convex program, i.e., an index \( i \in [m] \) and \( \hat{z} \in B_{\infty}^n(0, \delta) \) such that

\[
 r_i < \sup_{z \in B_{\infty}^n(0, \delta)} y_i (g(x_i) - g(x_i + \hat{z})).
\]

This gives us a separating hyperplane of the form \( \text{sgn}(y_i(g(x_i) - g(x_i + \hat{z}))-r_i) \), and the algorithm continues. Hence, we get that when the Ellipsoid algorithm terminates, with probability at least \( 1 - \eta \), it will output a polynomial \( g \in \mathcal{F} \) such that the constraints in Equations 4 and 6 are satisfied. This means that we would have the empirical robust error \( e_{\hat{r}_{\delta/\gamma,S}}(\text{sgn}(g)) = 0 \). Hence, by Lemma 3, we get that

\[
 \text{err}_{\delta/\gamma,D}(\text{sgn}(g)) \leq 2 \sqrt{\frac{2\Delta \log m}{m}} + \sqrt{\frac{\log 1}{2m}},
\]

where \( \Delta \) is the VC dimension of \( \mathcal{F} \). Choosing \( m = c\frac{\Delta + \log(1/\eta)}{\epsilon^2} \), makes \( \text{err}_{\delta/\gamma,D}(\text{sgn}(g)) \leq \epsilon \). □

In the next section we will use our algorithmic framework to show that many sub-classes of PTFs, including general degree-1 and degree-2 PTFs are admissible, and hence robustly PAC learnable.

4. Admissibility of Degree-1 and Degree-2 PTFs

In this section we instantiate the algorithmic framework developed in Section 3 and design polynomial time robust learning algorithms for the classes of degree-1 and degree-2 PTFs, by establishing admissibility of these classes. Then we use Theorem 7 to get the corresponding learning algorithm. Notice that conditions 1 and 2 in the definition of admissibility (See Definition 6) are easily satisfied by degree-\( d \) PTFs for any fixed \( d \). Hence our main goal in this section is to show that condition 3 in Definition 6 holds for our classes of interest. We begin with the following claim about admissibility of degree-1 PTFs.

**Theorem 9** The class of degree-1 PTFs is 1-admissible, i.e., there is a deterministic algorithm that given any \( n \)-variate degree-1 polynomial \( g(x) \) represented by \( g(x) := b^T x + c \) where \( b \in \mathbb{R}^n, c \in \mathbb{R} \), finds in polynomial time a solution \( \hat{x} \in B_{\infty}^n(0, \delta) \) such that \( g(\hat{x}) = \max_{x \in B_{\infty}^n(0, \delta)} g(x) \).

**Proof** The proof follows from the fact that in order to maximize the linear from \( b^T x + c \) within \( B_{\infty}^n(0, \delta) \), one should set each variable \( x_i \) to be \( \delta \) if the corresponding \( b_i \geq 0 \), and \(-\delta\), otherwise. □

As a result of the above theorem we get the following result about robust learning of degree-1 PTFs.
Corollary 10 The class of degree-1 PTFs is optimally robustly learnable.

Our second result of this section establishes admissibility of degree-2 PTFs.

Theorem 11 The class of degree-2 PTFs is $O(\sqrt{\log n})$-factor admissible i.e., given any $\delta, \eta > 0$, there is a randomized algorithm that runs in time $\text{poly}(n, \log(1/\eta))$, and with probability at least $1 - \eta$, finds a solution $\bar{x}$ with $\| \bar{x} \|_\infty \leq O(\sqrt{\log n}) \cdot \delta$ such that $g(\bar{x}) \geq \max_{\|x\|_\infty \leq \delta} g(x)$.

To prove Theorem 11 we use a semi-definite programming (SDP) based algorithm that is directly inspired by the SDP-based algorithm for quadratic programming (QP) by Nesterov (1998); Charikar and Wirth (2004). However, the goal in quadratic programming is to find an assignment $x \in \{-1, 1\}^n$ that maximizes $\sum_{i \neq j} a_{ij} x_i x_j$. There are three main differences from the QP problem. Firstly, unlike QP which finds a solution with $\|x\|_\infty = 1$ with sub-optimal objective value, our goal is to output a solution which attains at least as large a value as $\max_{\|x\|_\infty \leq \delta} g(x)$ while violating the $\ell_\infty$ length of the vector. Secondly, unlike QP where the diagonal terms are all zero, in our problem the diagonal terms can be non-zero and hence it is no longer true that the solution with $\|x\|_\infty \leq 1$ will have each co-ordinate being $\{-1\}$. Finally and most crucially, QP corresponds to optimizing a homogeneous degree 2 polynomial, with no linear term. These challenges necessitates non-trivial modifications to the algorithm and in the analysis. We also remark that it seems unlikely that the upper bound of $O(\sqrt{\log n})$ on the admissibility factor can be improved even for the special case of homogenous degree-2 polynomials, based on the current state of the approximability of Quadratic Programming (see Remark 18 for details).

The SDP we consider is given by the following equivalent vector program (the SDP variables correspond to $X_{ij} = \langle u_i, u_j \rangle$), which can be solved in polynomial time up to arbitrary additive error (using the Ellipsoid algorithm).

\[
\max_{\{u_0, u_1, \ldots, u_n\}} \sum_{i,j=1}^n A_{ij} \langle u_i, u_j \rangle + \sum_{i=1}^n b_i \langle u_i, u_0 \rangle + c \\
\text{s.t. } \|u_i\|_2^2 \leq \delta^2 \text{ } \forall i \in \{1, 2, \ldots, n\} \quad \text{(7)}
\]

\[
\|u_0\|_2^2 = 1. \quad \text{(9)}
\]

Let $\text{SDP}_{val}$ denote the optimal value of the above SDP relaxation. Clearly the above SDP is a valid relaxation of the problem: for any valid solution $x \in [-\delta, \delta]^n$, consider the solution given by $(u_i = x_i u_0 : i \in [n])$ for any unit vector $u_0$. Hence $\text{SDP}_{val} \geq \max_{\|x\|_\infty \leq \delta} g(x)$. We prove Theorem 11 by designing a polynomial time rounding algorithm with the following guarantee.

Lemma 12 There is a polynomial time randomized rounding algorithm that takes as input the solution of the SDP as defined in Equations 7, 8, and 9, and outputs a solution $\tilde{x}$ such that

\[
\mathbb{P}_{\tilde{x}} \left[ g(\tilde{x}) \geq \max_{\|x\|_\infty \leq \delta} g(x) \text{ and } \|\tilde{x}\|_\infty \leq O(\sqrt{\log n}) \cdot \delta \right] \geq \frac{1}{O(\log n)}. \quad \text{(10)}
\]

Assuming (10), we can repeat the algorithm at least $O(\log(1/\eta) \log n)$ times to get the guarantee of Theorem 11.

Rounding Algorithm. Given the SDP solution, let $u_i^\perp$ represent the component of $u_i$ orthogonal to $u_0$. Consider the following randomized rounding algorithm that returns a solution $\{ \tilde{x}_i : i \in [n] \}$:

\[
\forall i \in \{0, 1, \ldots, n\}, \quad \tilde{x}_i := \langle u_i, u_0 \rangle + \langle u_i, g \rangle = \langle u_i, u_0 \rangle + \langle u_i^\perp, g \rangle, \text{ with } g \sim \mathcal{N}(0, \Pi^\perp),
\]
where $\Pi^\perp$ is the projection matrix onto the subspace of $\text{span}\{u_1, \ldots, u_n\}$ that is orthogonal to $u_0$. For convenience, we can assume without loss of generality that $u_0 = e_0$, where $e_0$ is a standard basis vector, and $u_i \in \mathbb{R}^{n+1}$. Let $e_0, e_1, \ldots, e_n$ represent an orthogonal basis for $\mathbb{R}^{n+1}$. Then
\[ \forall i \in \{0, 1, \ldots, n\}, \, \bar{x}_i = \langle u_i, u_0 \rangle + \langle u_i^\perp, g \rangle \quad \text{where} \quad \langle g, e_0 \rangle = 0, \, \langle g, v \rangle \sim N(0, \|v\|_2^2) \quad \text{for every} \quad v \perp e_0, \]
and $\bar{x}_0 = 1$. The key observation in the analysis of the rounding algorithm is that $\forall i, j \in \{0, \ldots, n\}$,
\[
\mathbb{E} [\bar{x}_i \bar{x}_j] = \mathbb{E}_g \left[ \left( \langle u_i, u_0 \rangle + \langle u_i^\perp, g \rangle \right) \left( \langle u_j, u_0 \rangle + \langle u_j^\perp, g \rangle \right) \right] = \langle u_i, u_0 \rangle \langle u_j, u_0 \rangle + \mathbb{E}_g \left[ \langle u_i^\perp, g \rangle \langle u_j^\perp, g \rangle \right] \nonumber \]
\[
= \langle u_i, u_0 \rangle \langle u_j, u_0 \rangle + \langle u_i^\perp, u_j^\perp \rangle = \langle u_i, u_j \rangle. \tag{11} \nonumber
\]

Note that this also holds when $i = j$. This allows us to preserve the objective value in expectation. We defer the rest of the details of the proof of Lemma 12 and analysis to the appendix (see Section B). We conclude the section by stating the following corollary of Theorem 11.

**Corollary 13** The class of degree-2 PTFs is $O(\sqrt{\log n})$-approximately robustly learnable.

### 5. Computational Hardness of Learning Optimally Robust Classifiers

In this section we show that assuming $NP$ does not have randomized polynomial time algorithms, no polynomial time algorithm $1$-approximately robust learns degree-2 PTFs. We prove that there exists a $\delta > 0$ and a set of $N = \text{poly}(n)$ points such that it is hard to distinguish whether there exists a degree-2 PTF that is $\delta$ robust to all the points or not.

**Theorem 14** [Hardness] For every $\delta > 0$, assuming $NP \neq RP$ there is no polynomial time algorithm that given a set of $N = O(n^2)$ labeled points $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\}$ with $(x^{(j)}, y^{(j)}) \in \mathbb{R}^{n+1} \times \{-1, 1\}$ for all $j \in [N]$ can determine whether there exists a degree-2 PTF that has $\delta$-robust empirical error of 0 on these $N$ points (as in Definition 2).

The above theorem immediately implies the following result about hardness of optimal robust learning of degree-2 PTFs.

**Corollary 15** [Distributional Hardness] For every $\delta > 0$, there exists a distribution $D$ over $\mathbb{R}^n \times \{-1, +1\}$ and $\epsilon > 0$, such that assuming $NP \neq RP$ there is no polynomial time algorithm that given a set of $\text{poly}(n, \frac{1}{\delta})$ points from $D$ labeled by a degree-2 PTF that has $\delta$-robust error of 0 w.r.t. $D$, outputs a degree-2 PTF of $\delta$-robust error at most $\epsilon$ w.r.t. $D$.

The corollary above follows from the standard fact used in establishing learning theoretic hardness (Kearns et al., 1994), namely if there were a robust learning algorithm for every distribution and $\epsilon > 0$, the one could use it on the uniform distribution over the instance from Theorem 14 with $\epsilon = \frac{1}{2N}$ to determine whether there exists a degree-2 PTF that has $\delta$-robust empirical error of 0 on the points in the instance. Hence our main goal in this section is to prove Theorem 14. In order to do this, we give a reduction from Quadratic Programming (QP) where given a polynomial $p(x) = \sum_{i<j} a_{ij} x_i x_j$, the goal is to evaluate $\max_{x \in \{-1, 1\}^n} p(x)$. We will represent the polynomial $p(x) = x^T A x$ where $A$ is a symmetric matrix with zeros on the diagonal, and $A_{ij} = A_{ji} = a_{ij}/2$. Formally, the $NP$-hard problem $QP$ (Arora et al., 2005; Garey and Johnson, 2002) is the following: given $\beta > 0$ and a polynomial $p(x) = x^T A x$ distinguish whether

**No Case** : there exists an assignment $x^* \in \{-1, 1\}^n$ such that $p(x^*) > \beta$,

**Yes Case** : for every assignment $x \in \{-1, 1\}^n$, $p(x) < \beta$. 

10
which allows the algorithm to output a robust classifier that makes errors on constant fraction of the points! Hence, even when there is a degree-2 PTF that has \( \delta \) robust error of 0, it is computationally hard to output a degree-2 PTF that has \( \delta \)-robust error of \( \varepsilon \leq \frac{1}{4} \). Please see Section C for the proof.
1. Let \( p(x) := x^T Ax \) be the polynomial given by Problem QP, and let \( \beta, \delta \) be the given parameters. Set \( \alpha := \delta^2 \beta + \delta, \rho := c_\delta n^{3/2} m \), for some sufficiently large constant \( c_\delta \geq 1 \).

2. Using \( A \) we generate \( m \) points \( (x^{(j)}, z^{(j)}) \in \mathbb{R}^{n+1} \) as follows. Sample point \( x^{(j)} \) from \( \mathcal{N}(0, \rho^2)^n \), then set \( z^{(j)} = p(x^{(j)}) - (x^{(j)})^T A x^{(j)} \) for each \( j \in [m] \).

3. Define \( s^{(j)} = \text{sgn}(\nabla p(x^{(j)})) \) where the \( \text{sgn}(x) \in \{-1, 1\}^n \) refers to a vector with entry-wise signs, and \( \nabla p \) stands for the gradient of \( p \) at \( x^{(j)} \). From each \( (x^{(j)}, z^{(j)}) \) generate 
\[
(u^{(j)}, z^{(j)}_{u}) = (x^{(j)} - \delta s^{(j)}, z^{(j)} + \delta) \text{ with label } y^{(j)}_u = \text{sgn}(z^{(j)}_u - p(u^{(j)})) \] \[
(v^{(j)}, z^{(j)}_{v}) = (x^{(j)} + \delta s^{(j)}, z^{(j)} - \delta) \text{ with label } y^{(j)}_v = \text{sgn}(z^{(j)}_v - p(v^{(j)})).
\]

4. Generate \( \alpha \) (depends on \( \delta \) and \( \beta \) from problem QP) and input the \( 2m + 1 \) points in \( \mathbb{R}^{n+1} \times \{-1, 1\} \) given by \( (u^{(j)}, z^{(j)}_u, y^{(j)}_u), (v^{(j)}, z^{(j)}_v, y^{(j)}_v) \) for each \( j \in [m] \) and \( (0, \alpha, +1) \) to the algorithm.

![Figure 3: Reduction from the QP problem.](image)

**Theorem 16 [Stronger Distributional Hardness]** For every \( \delta > 0 \) and \( \varepsilon \in (0, \frac{1}{3}) \), there exists a distribution \( D \) over \( \mathbb{R}^n \times \{-1, +1\} \), such that assuming \( \text{NP} \neq \text{RP} \) there is no polynomial time algorithm that given a set of \( N = \text{poly}(n, \frac{1}{\varepsilon}) \) points from \( D \) labeled by a degree-2 PTF that has \( \delta \)-robust error of \( 0 \) w.r.t. \( D \), outputs a degree-2 PTF of \( \delta \)-robust error at most \( \varepsilon \) w.r.t. \( D \).

### 6. Conclusions and Open Problems

Design of polynomial time algorithms that achieve adversarial robustness is an important direction of research and our work provides the first results along these lines. Several open questions remain to be explored further. In Section 3 we provide a general algorithmic framework for designing polynomial time robust algorithms. It would be interesting to use our framework to design robust algorithms for general degree-\( d \) PTFs. While there have been works on approximation algorithms for maximizing degree-\( d \) polynomials, they focus on the homogeneous case which does not suffice for our purposes since homogeneous degree-\( d \) PTFs do not satisfy our definition of admissibility. Hence, robust learning of degree-\( d \) polynomials in our framework will require the design of new approximation algorithms for the corresponding polynomial maximization problem; a question of independent interest beyond the application to robustness. We also conjecture that for PTFs, the connection between adversarial robustness and polynomial maximization is tight, and that it is hard to \( \gamma \)-approximately robustly learn degree-2 polynomials for \( \gamma = \Omega(\sqrt{\log n}) \). Our current lower bound only establishes NP-hardness. It would also be interesting to prove computational hardness or cryptographic hardness of robust learning of PTFs when the learning algorithm is not restricted to output a PTF. Finally, it is worth investigating whether our algorithmic framework can be applied to function classes beyond PTFs, such as neural networks and kernel classifiers.

### References


Appendix A. Related Work

As mentioned in the introduction, there has been a recent explosion of works on understanding adversarial robustness from both empirical and theoretical aspects. Here we choose to discuss the theoretical works that are the most relevant to our paper. We refer the interested reader to a recent paper by Gilmer et al. (2018a) for a broader discussion. Prior to their relevance for deep networks, robust optimization problems as in Equation 1 have been studied in machine learning and other domains. The works of Bhattacharyya (2004); Globerson and Roweis (2006); Shivaswamy et al. (2006) studies optimization heuristics for optimizing a robust loss that can handle noisy or missing data. The works of Xu et al. (2009); Xu and Mannor (2012) proved an equivalence between robust optimization and various regularized variants of SVMs. They used this relation to re-derive standard generalization bounds for SVMs and their kernel versions. Akin to classifier stability, these bounds depend on the robustness of the classifier on the training set. A recent work of Bietti et al. (2018) views deep networks as functions in an RKHS and designs new norm based regularization algorithms to achieve robustness.

Motivated by connections to deep networks a recent line of work studies generalization bounds for robust learning. The work of Schmidt et al. (2018) provides specific constructions of a linear binary classification task where a single example is enough to learn the problem in the usual sense,
i.e., to achieve low test error, whereas learning the problem robustly requires a significantly large training set. The authors also show that in certain cases, non-linearity can help reduce the sample complexity of robust learning. The work of Cullina et al. (2018) proposes a PAC model for robust learning and defines adversarial VC dimension as a combinatorial quantity that captures robust learning via robust empirical risk minimization (ERM). The authors show that for linear classifiers the adversarial VC dimension is the same as the VC dimension, although there are functions classes and distributions where the gap between the two quantities could be much higher. The recent works of Yin et al. (2018) and Khim and Loh (2018) analyze Rademacher complexity of robust loss functions classes. In particular, it is observed that even for linear models with bounded weight norm, there is an unavoidable dependence on the data dimension in the Rademacher complexity of robust loss function classes. These results point to the fact that for many distributions robust learning could require many more training samples than their non-robust counterpart. The work of Feige et al. (2015); Attias et al. (2018) studies algorithms and generalization bounds for a model where the adversary can choose perturbations from a known finite set of small size $k$.

Another recent line of work studies the trade-off between traditional test error and robust error. The work of Tsipras et al. (2018) designs a classification task that is efficiently learnable with a linear classifier to low standard error, but has the property that any classifier that achieves low test error will have high robust error on the task. The work of Gilmer et al. (2018b) designs a task that is learnable by a degree-2 polynomial and relates the test error of any model to its robust error. Similar conclusions have been observed in Mahloujifar et al. (2018); Mahloujifar and Mahmoody (2018); Diochnos et al. (2018) and have been used to design various data poisoning attacks. These results essentially follows from the use of isoperimetric inequalities for distributions such as the Gaussian and the uniform distribution over the Boolean hypercube. However, as noted in Gilmer et al. (2018b), it is not clear if the same relation holds between test error and robust error for real world data distributions. The work of Fawzi et al. (2016) relates robustness to the curvature of the decision boundary and uses it to quantify robustness to random perturbations.

Yet another line of work concerns the design of certificates of perturbation robustness or distributional robustness of a given classifier (e.g., deep neural networks) at a given point (Wong and Kolter, 2018; Raghunathan et al., 2018; Sinha et al., 2018). This is achieved by the use of convex relaxations of the optimal robustness at a given point. These works also conclude that by augmenting the training objective with a penalty that depends on the certificates, one can empirically achieve increased robustness. However these algorithms do not give any guarantees for relating the bound achieved by the certificate of robustness to the optimal robustness around a given point.

**Appendix B. Admissibility of degree-2 PTFs**

We recall the SDP and the rounding algorithm.

\[
\max \{ u_0, u_1, \ldots, u_n \} \quad \sum_{i,j=1}^{n} A_{ij} \langle u_i, u_j \rangle + \sum_{i=1}^{n} b_i \langle u_i, u_0 \rangle + c \tag{12}
\]

s.t. \( \| u_i \|_2^2 \leq \delta^2 \quad \forall i \in \{1, 2, \ldots, n\} \) \tag{13}

\[
\| u_0 \|_2^2 = 1. \tag{14}
\]
Let SDP_{val} denote the optimal value of the above SDP relaxation. Clearly the above SDP is a valid relaxation of the problem; for any valid solution \( x \in [-\delta, \delta]^n \), consider the solution given by \( (u_i = x_i; u_0 : i \in [n]) \) for any unit vector \( u_0 \). Hence \( \text{SDP}_{val} \geq \max_{\|x\|_\infty \leq \delta} g(x) \).

**Rounding Algorithm.** Given the SDP solution, let \( u_i^\perp \) represent the component of \( u_i \) orthogonal to \( u_0 \). Consider the following randomized rounding algorithm that outputs a solution \( \{ \hat{x}_i : i \in [n] \} \) as follows:

\[
\forall i \in \{0, 1, \ldots, n\}, \quad \hat{x}_i := \langle u_i, u_0 \rangle + \langle u_i, g \rangle = \langle u_i, u_0 \rangle + \langle u_i^\perp, g \rangle, \quad \text{with} \quad g \sim N(0, \Pi^\perp),
\]

where \( \Pi^\perp \) is the projection matrix onto the subspace of \( \text{span}\{u_1, \ldots, u_n\} \) that is orthogonal to \( u_0 \). For convenience, we can assume without loss of generality that \( u_0 = e_0 \), where \( e_0 \) is a standard basis vector, and \( u_i \in \mathbb{R}^{n+1} \). Let \( e_0, e_1, \ldots, e_n \) represent an orthogonal basis for \( \mathbb{R}^{n+1} \). Then \( \forall i \in \{0, 1, \ldots, n\}, \)

\[
\hat{x}_i = \langle u_i, u_0 \rangle + \langle u_i^\perp, g \rangle, \quad \text{where} \quad \langle g, e_0 \rangle = 0 \quad \text{and} \quad \langle g, v \rangle \sim N(0, \|v\|_2^2) \quad \text{for every} \quad v \perp e_0,
\]

and \( \hat{x}_0 = 1 \). We now give the analysis of the algorithm.

We prove Theorem 11 by showing the following guarantee for the rounding algorithm.

**Lemma 17 (Same as Lemma 12)** There is a polynomial time randomized rounding algorithm that takes as input the solution of the SDP as defined in Equations 7, 8, and 9, and outputs a solution \( \hat{x} \) such that

\[
\mathbb{P}_{\hat{x}} \left[ g(\hat{x}) \geq \max_{\|x\|_\infty \leq \delta} g(x) \text{ and } \|\hat{x}\|_\infty \leq O\left(\sqrt{\log n} \cdot \delta\right) \right] \geq \frac{1}{O(\log n)}.
\]

Assuming (10), we can repeat the algorithm at least \( O(\log(1/\eta) \log n) \) times to get the guarantee of Theorem 11.

**Proof [Proof of Lemma 12]** We start with a simple observation that follows from the standard properties of spherical Gaussians. For any \( i, j \in [n] \), we have \( \mathbb{E}_g[\langle u_i^\perp, g \rangle \langle u_j^\perp, g \rangle] = (u_i^\perp)^T \Pi^\perp u_j^\perp = \langle u_i^\perp, u_j^\perp \rangle \). Hence we get the key observation that for \( \forall i, j \in \{0, \ldots, n\}, \)

\[
\mathbb{E} [\hat{x}_i \hat{x}_j] = \mathbb{E}_g \left[ \left( \langle u_i, u_0 \rangle + \langle u_i^\perp, g \rangle \right) \left( \langle u_j, u_0 \rangle + \langle u_j^\perp, g \rangle \right) \right] = \langle u_i, u_0 \rangle \langle u_j, u_0 \rangle + \mathbb{E}_g \left[ \langle u_i^\perp, g \rangle \langle u_j^\perp, g \rangle \right]
\]

\[
= \langle u_i, u_0 \rangle \langle u_j, u_0 \rangle + \langle u_i^\perp, u_j^\perp \rangle = \delta \langle u_i, u_j \rangle.
\]

Note that this also holds when \( i = j \). We now consider the expected value of \( g(\hat{x}) \). Using (16), \( \hat{x}_0 = 1 \) and since \( \mathbb{E}_g[\langle u_i^\perp, g \rangle] = 0 \), we have

\[
\mathbb{E}[g(\hat{x})] = \sum_{i,j=1}^{n} A_{ij} \mathbb{E}_g[\hat{x}_i \hat{x}_j] + \sum_{i=1}^{n} b_i \mathbb{E}_g[\hat{x}_i \hat{x}_0] + c \mathbb{E}_g[\hat{x}_0^2]
\]

\[
= \sum_{i,j=1}^{n} A_{ij} \langle u_i, u_j \rangle + \sum_{i=1}^{n} b_i \langle u_i, u_0 \rangle + c \left\| u_0 \right\|_2^2 = \text{SDP}_{val}.
\]

We now show that \( \hat{x}_i \leq O\left(\sqrt{\log n} \cdot \delta\right) \) w.h.p. For each fixed \( i \in \{1, \ldots, n\}, \langle u_i^\perp, g \rangle \) is distributed as a Gaussian with mean 0 and variance \( \|u_i^\perp\|_2^2 \leq \delta^2 \),

\[
|\hat{x}_i| \leq |\langle u_i, u_0 \rangle| + |\langle u_i^\perp, g \rangle| \leq \delta + |\langle u_i^\perp, g \rangle| \leq \sqrt{C \log n} \cdot \delta \quad \text{with probability at least} \quad 1 - 1/n^{C/2},
\]
using standard tail properties of Gaussians. Hence, using a union bound over all \(i \in [n]\), we have that
\[
\mathbb{E}[g(\hat{x})] \geq \max_{\|x\|_\infty \leq \delta} g(x), \quad \text{and } \mathbb{P}\left(\|\hat{x}\|_\infty \leq O(\sqrt{\log n}) \cdot \delta \right) \geq 1 - \frac{1}{n^2}, \quad (18)
\]
for \(C \geq 4\).

**Showing the high probability bound.** We now prove (10). Recall that \(\mathbb{E}[g(\hat{x})] \geq \text{SDP}_{\text{val}} \geq \max_{\|x\|_\infty \leq \delta} g(x)\). For any \(C > 0\), let \(\text{SDP}_{\text{val}}^{(C \log n)}\) be the value of the optimal SDP solution when constraint (8) is replaced by
\[
\|u_i\|^2 \leq (C \log n)\delta^2 \quad \forall i \in [n].
\]
Let \(v_0, v_1, \ldots, v_n\) represent the optimal SDP solution. Then
\[
\text{SDP}_{\text{val}}^{(C \log n)} - c = \sum_{i,j=1}^{n} A_{ij}\langle v_i, v_j \rangle + \sum_{i=1}^{n} b_i \langle v_i, v_0 \rangle.
\]
Note that \(\sum_{i} b_i \langle v_i, v_0 \rangle \geq 0\) (otherwise we can negate all the \(v_i\) to \(-v_i\) and increase the value). Let \(v'_i = \frac{v_i}{\sqrt{C \log n}}\) for all \(i \in \{1, \ldots, n\}\). As \(v_0, v'_1, \ldots, v'_n\) is a valid solution for the original SDP we have
\[
\text{SDP}_{\text{val}} - c \geq \sum_{i,j=1}^{n} A_{ij}\langle v'_i, v'_j \rangle + \sum_{i=1}^{n} b_i \langle v'_i, v_0 \rangle. \quad \text{Multiplying by } C \log n,
\]
\[
C \log n(\text{SDP}_{\text{val}} - c) \geq \sum_{i,j=1}^{n} A_{ij}\langle v_i, v_j \rangle + \sqrt{C \log n} \sum_{i=1}^{n} b_i \langle v_i, v_0 \rangle \geq \sum_{i,j=1}^{n} A_{ij}\langle v'_i, v'_j \rangle + \sum_{i=1}^{n} b_i \langle v'_i, v_0 \rangle \geq \text{SDP}_{\text{val}}^{(C \log n)} - c.
\]

Now consider the distribution over solutions \(\hat{x}\) output by the randomized rounding algorithm. With probability at least \(1 - n^{-C/2}\), \(\|\hat{x}\|_\infty \leq \delta \sqrt{C \log n}\); and in this case the maximum value that it takes
\[
\max_{\|\hat{x}\|_\infty \leq \delta \sqrt{C \log n}} g(\hat{x}) - c \leq \text{SDP}_{\text{val}}^{(C \log n)} - c \leq C \log n(\text{SDP}_{\text{val}} - c).
\]
Combining this with (17) we have
\[
\text{SDP}_{\text{val}} - c \leq \mathbb{E}\left[g(\hat{x}) - c\right] \leq \int_C \mathbb{P}\left(\|\hat{x}\|_\infty \leq \sqrt{C \log n} \delta; \, g(\hat{x}) \geq \text{SDP}_{\text{val}}\right) \times \left(\text{SDP}_{\text{val}}^{(C \log n)} - c\right) \, dC
\]
\[
\leq \mathbb{P}\left(\|\hat{x}\|_\infty \leq 4\sqrt{\log n}\delta; \, g(\hat{x}) \geq \text{SDP}_{\text{val}}\right) \times 16 \log n(\text{SDP}_{\text{val}} - c)
\]
\[
+ \int_{C>4} \frac{1}{n^{C/2}} \times C \log n(\text{SDP}_{\text{val}} - c) \, dC
\]
\[
\leq \log n(\text{SDP}_{\text{val}} - c) \cdot \left(16 \mathbb{P}\left(\|\hat{x}\|_\infty \leq 4\sqrt{\log n}\delta; \, g(\hat{x}) \geq \text{SDP}_{\text{val}}\right) + \frac{1}{n\pi}\right).
\]
This establishes (10).
Remark 18 Obtaining an admissibility factor of $O(\gamma)$ for the special case of homogeneous degree-2 polynomials $\sum_{i,j=1}^n a_{ij}x_ix_j$ with no diagonal entries ($a_{ii} = 0 \forall i \in [n]$) over $\|x\|_\infty \leq \delta$ is equivalent to obtaining a $O(\gamma^2)$-factor approximation algorithm for the problem called Quadratic Programming (QP) which maximizes $\sum_{i,j=1}^n a_{ij}x_ix_j$ over $x \in \{-1, 1\}^n$ (this is also called the Grothendieck problem on complete graphs). The best known approximation algorithm for Quadratic Programming (QP) gives an $O(\sum_{i<j}^n 2^i)$-factor approximation in polynomial time (Nesterov, 1998; Charikar and Wirth, 2004). Further Arora et al. (2005) showed that it is hard to approximate QP within a $O(\log^c n)$ for some universal constant $c > 0$ assuming NP does not have quasi-polynomial time algorithms. Moreover integrality gaps for SDP relaxations (Alon et al., 2006; Khot and O’Donnell, 2006) suggest that $O(\log n)$ factor maybe be tight for polynomial time algorithms. Hence even for the special case of homogeneous degree-2 polynomials, improving upon the bound of $\sqrt{\log n}$ in the admissibility factor seems unlikely.

Appendix C. Computational hardness of learning optimally robust classifiers

We recall the steps in the reduction. We reduce from the QP problem (Problem $\mathcal{QP}$) which is known to be NP hard.

1. Let $p(x) := x^TAx$ be the polynomial given by Problem $\mathcal{QP}$, and let $\beta, \delta$ be the given parameters. Set $\alpha := \delta^2/\beta + \delta$, $\rho := c_3\delta n^{3/2}/m$, for some sufficiently large constant $c_3 \geq 1$.

2. Using $A$ we generate $m$ points $(x^{(j)}, z^{(j)}) \in \mathbb{R}^{n+1}$ as follows. Sample point $x^{(j)}$ from $\mathbb{N}(0, \rho^2)^n$, then set $z^{(j)} = p(x^{(j)}) = (x^{(j)})^TAx^{(j)}$ for each $j \in [m]$.

3. Define $s^{(j)} = \text{sgn}(\nabla p(x^{(j)}))$ where the $\text{sgn}(x) \in \{-1, 1\}^n$ refers to a vector with entry-wise signs, and $\nabla p$ stands for the gradient of $p$ at $x^{(j)}$. From each $(x^{(j)}, z^{(j)})$ generate $\left(u^{(j)}, z_u^{(j)}\right) = (x^{(j)} - \delta s^{(j)}, z^{(j)} + \delta)$ with label $y_u^{(j)} = \text{sgn}(z_u^{(j)} - p(u^{(j)}))$ and $\left(v^{(j)}, z_v^{(j)}\right) = (x^{(j)} + \delta s^{(j)}, z^{(j)} - \delta)$ with label $y_v^{(j)} = \text{sgn}(z_v^{(j)} - p(v^{(j)}))$.

4. Generate $\alpha$ (depends on $\delta$ and $\beta$ from problem $\mathcal{QP}$) and input the $2m + 1$ points in $\mathbb{R}^{n+1} \times \{\pm 1\}$ given by $((u^{(j)}, z_u^{(j)}), y_u^{(j)}), ((v^{(j)}, z_v^{(j)}), y_v^{(j)}$ for each $j \in [m]$ and $(0, \alpha, +1)$ to the algorithm.

Figure 4: Reduction from the QP problem.

To argue the soundness and the completeness of our reduction, we will first analyze the robustness of degree-2 PTFs on the $2m$ added labeled examples $((u^{(\ell)}, z_u^{(\ell)}), y_u^{(\ell)})$ and $((v^{(\ell)}, z_v^{(\ell)}), y_v^{(\ell)})$. We will show that the “intended” PTF $\text{sgn}(z - p(x))$ is the unique degree-2 PTF (up to scaling) that is robust at all these $2m$ points. Note that a degree-2 PTF $\text{sgn}(q(x, z))$ on the $n + 1$ variables $(x, z)$ may not necessarily be of the form $\text{sgn}(z - g(x))$ for some degree-2 polynomial $g(x)$. We need to rule out the existence of any other degree-2 PTF of the form $\text{sgn}(q(x, z))$ that is $\delta$-robust at these points. Once we have established this, we will then show that the “intended” PTF $\text{sgn}(z - p(x))$ is $\delta$-robust at $((0, \alpha), +1)$ in the YES case, and not $\delta$-robust at $((0, \alpha), +1)$ in the NO case.
Figure 5: The figure shows the construction of a hard instance for the robust learning problem. First, points \((x^{(j)}, z^{(j)})\) are sampled randomly and satisfying \(z^{(j)} = p(x^{(j)})\). Each such point is then perturbed along the direction of the sign vector of the gradient at \((x^{(j)}, z^{(j)})\) to get two data points of the training set, one labeled as \(+1\), and the other labeled as \(-1\).

We proceed by first proving that the intended PTF \(\text{sgn}(z - p(x))\) is robust at the \(2m\) added examples. Recall that the points \(x^{(j)} \in \mathbb{R}^n\) are chosen according to a Gaussian distribution with variance \(\rho^2\) in every direction. The following lemma shows a property that holds w.h.p. for the points \(\{x^{(\ell)} : \ell \in [m]\}\) that will be key in proving the robustness of \(\text{sgn}(z - p(x))\) at the \(2m\) added points in Lemma 21.

**Lemma 19** There exists some universal constant \(C > 0\) such that for any \(\eta > 0\), assuming \(\rho \geq C\delta n^{3/2}m/\eta\) we have with probability at least \(1 - \eta\) that

\[
\forall \ell \in [m], \forall i \in [n], \quad \frac{|\langle A_i, x^{(\ell)} \rangle|}{\|A_i\|_1} > \delta, \tag{19}
\]

where \(A_i\) denotes the \(i\)th row of \(A\).

**Proof** The proof follows from the following standard anti-concentration fact about Gaussians.

**Fact 20** Let \(x^*\) be sampled from \(\mathcal{N}(0, \rho^2)^n\). Let \(a \in \mathbb{R}^n\). There exists a universal constant \(C > 0\) such that for any \(\eta' > 0\),

\[
P \left[ |\langle a, x^* \rangle| \leq C\|a\|_2 \eta' \right] \leq \eta'.
\]

Set \(\eta' := \eta/(mn)\). Fix \(\ell \in [m], i \in [n]\). Using Fact 20 we have with probability at least \(1 - \eta'\)

\[
|\langle A_i, x^{(j)} \rangle| \geq \|A_i\|_2 \eta' \geq \frac{\|A_i\|_1}{\sqrt{n}} \cdot \rho \cdot \frac{\eta}{mn} \geq \delta,
\]
from our assumption on $\rho$. The lemma follows from a union bound over all $\ell \in [m], i \in [n]$. ■

We now prove the $\delta$-robustness of the “intended” degree-2 PTF $\text{sgn}(z - p(x))$ at the $2m$ added points w.h.p.

**Lemma 21** There exists constant $C > 0$ such that for any $\eta > 0$, assuming $\rho \geq C\delta n^{3/2}m/\eta$, then with probability at least $1 - \eta$, the degree-2 PTF $\text{sgn}(z - p(x)) = \text{sgn}(z - x^T A x)$ is $\delta$-robust at all the $2m$ points $\{(u^\ell, z_u^\ell), (v^\ell, z_v^\ell) : \ell \in [m]\}$.

**Proof** Consider a fixed $\ell \in [m]$. For convenience let $x^*, z^*, u, v, z_u, z_v$ denote $x^\ell, z^\ell, u^\ell, v^\ell, z_u^\ell, z_v^\ell$ respectively, and let $s = \text{sgn}(\nabla p(x^\ell)) \in \{-1, 1\}^n$. Hence $z^* = x^* A x^*$, $(u, z_u) = (x^* - \delta s, z^* + \delta)$ and $(v, z_v) = (x^* + \delta s, z^* - \delta)$. We want to prove that the points $(u, z_u)$ and $(v, z_v)$ are $\delta$-robust i.e., these points are $\delta$ away in $\ell_\infty$ distance from the decision boundary of $\text{sgn}(z - p(x))$.

We now prove the following claim:

**Claim.** Any point $(x, z) \in B_{\infty}^{n+1}(u, z_u)$ is on the ‘positive’ side i.e., $z - x^T A x > 0$.

Note that $(u, z_u)$ itself lies inside the ball, and hence the claim will show that $\text{sgn}(z - x^T A x)$ is $\delta$-robust at $(u, z_u)$. An analogous proof also holds that $\delta$-robustness at $(v, z_v)$.

**Proof of Claim.** Let’s now define $\tilde{x} = x - x^*$, $\tilde{z} = z - z^*$. A simple observation is that $(x, z)$ lies on the opposite orthant with respect to $(x^*, z^*)$ as $s$, and we have (as shown in Figure 6)

$$\forall j \in [d], -2\delta \leq s(j)\tilde{x}(j) \leq 0, \quad \text{and} \quad \tilde{z} \geq 0.$$
Using \( z^* = p(x^*) \) and \( \bar{z} \geq 0 \), for all \((x, z) \in B^{n+1}((u, z_u), \delta)\) we have
\[
z - p(x) = z^* + \bar{z} - p(\bar{x} + x^*) = \bar{z} + p(x^*) - p(\bar{x} + x^*) = \bar{z} - \langle \nabla p, \bar{x} \rangle - \frac{1}{2} \bar{x}^T \nabla^2 p \bar{x}
\]
\[
\geq - \sum_{i=1}^{n} \bar{x}(i) \left( \sum_{j=1}^{n} a_{ij} x^*(j) \right) - \frac{1}{2} \sum_{i=1}^{n} \bar{x}(i) \left( \sum_{j=1}^{n} a_{ij} \bar{x}(j) \right)
\]
\[
= \sum_{i=1}^{n} (-\bar{x}(i)s(i)) \left| \sum_{j=1}^{n} a_{ij} x^*(j) \right| - \frac{1}{2} \sum_{i=1}^{n} \bar{x}(i) \sum_{j=1}^{n} a_{ij} \bar{x}(j)
\]
\[
\geq \sum_{i=1}^{n} |\bar{x}(i)| \left( \left| \sum_{j=1}^{n} a_{ij} x^*(j) \right| - \delta \sum_{j=1}^{n} |a_{ij}| \right),
\]
using the fact that \( \bar{x}(i)s(i) \in [-2\delta, 0] \) for each \( i \in [n] \). Applying Lemma 19 we see that with probability at least \((1 - \eta)\), (19) holds, and hence \(|\langle x^*, A_i \rangle| > \delta \| A_i \|_1 \) for each \( i \in [n] \) as required. This establishes the claim, and proves the lemma.

We now prove that the “intended” PTF \( \text{sgn}(z - p(x)) \) is only degree-2 that is robust at the added \( 2m \) examples.

**Lemma 22** Consider any degree-2 PTF \( \text{sgn}(q(x, z)) \) that is \( \delta \)-robust at the \( 2m \) labeled points \( \{ ((u^\ell, z_u^\ell), +1) : \ell \in [m] \} \) and \( \{ ((v^\ell, z_v^\ell), -1) : \ell \in [m] \} \) and is consistent with their labels. Then \( q(x, z) = C(z - p(x)) \) for some \( C \neq 0 \).

The proof of Lemma 22 follows immediately from the following two lemmas (Lemma 23 and Lemma 24).

**Lemma 23** Consider any degree-2 PTF on \( n + 1 \) variables \( \text{sgn}(q(x, z)) \) that satisfies the conditions of Lemma 22. Then \( q(x^\ell, z^\ell) = 0 \) for each \( \ell \in [m] \).

**Proof** Since \( \text{sgn}(q(u^\ell, z_u^\ell)) \neq \text{sgn}(q(v^\ell, z_v^\ell)) \), by the Intermediate Value Theorem,
\[
\exists \gamma \in [0, 1] \text{ s.t. } (\bar{x}, \bar{z}) = \gamma(u^\ell, z_u^\ell) + (1 - \gamma)(v^\ell, z_v^\ell) \text{ and } q(\bar{x}, \bar{z}) = 0.
\]

Also, since \( q \) is \( \delta \)-robust at \((u^\ell, z_u^\ell)\) and \((v^\ell, z_v^\ell)\), we must have that \((\bar{x}, \bar{z})\) is atleast \( \delta \) far away in \( \ell_\infty \) distance from both \((u^\ell, z_u^\ell)\) and \((v^\ell, z_v^\ell)\). Further by design two points are separated by exactly \( 2\delta \) in each co-ordinate (see Figure 6 for an illustration)! Hence it is easy to see that \( \gamma = 1/2 \) i.e., \((\bar{x}, \bar{z}) = (x^\ell, z^\ell)\) as required.

We now show that \( q(x, z) = z - p(x) \) is the only polynomial over \((n + 1)\) variables that evaluates to 0 on all points \( \{ (x^\ell, z^\ell) : \ell \in [m] \} \). Together with Lemma 23 this establishes the proof of Lemma 22.

**Lemma 24** Let \( m > (n+1)^2 \) and let \( q : \mathbb{R}^{n+1} \rightarrow \mathbb{R} \) be any degree-2 polynomial with \( q(x^\ell, z^\ell) = 0 \) for all \( \ell \in [m] \), where \( z^\ell = (x^\ell)^TA^*x^\ell \) and \( x^\ell \sim N(0, \rho^2) \) with \( \rho > 0 \). Then with probability 1, \( q(x, z) = C(z - x^T A^* x) \) for \( C \neq 0 \).
Proof We can represent a general degree-2 polynomial \( q : \mathbb{R}^{n+1} \to \mathbb{R} \) given by

\[
q(x, z) = x^T Ax + b_1^T x + c_1 + z b_2^T x + c_2 z^2 + c_3 z,
\]

where \( x \in \mathbb{R}^n, z \in \mathbb{R} \).

This polynomial is parameterized by a vector \( w = (A, b_1, c_1, b_2, c_2, c_3) \in \mathbb{R}^r \) where \( r = (n + 1)^2 + 2n + 3 \) (since \( A \) is symmetric). Now given a point \( (x^{(\ell)}, z^{(\ell)}) \), the equation \( q(x^{(\ell)}, z^{(\ell)}) = 0 \) is a linear equation over the coefficients \( w \) of \( q \). Hence, the set of conditions \( q(x^{(\ell)}, z^{(\ell)}) = 0 \) can be expressed as a systems of linear equations \( Mw = 0 \) over the (unknown) co-efficients \( w \). Hence any valid polynomial \( q \) corresponds to a solution of the linear system \( Mw = 0 \) and vice-versa. We now describe the matrix \( M \in \mathbb{R}^{m \times r} \). Define

\[
f(x, z) := (1) \oplus (x_1, \ldots, x_n) \oplus (x_j : i \leq j \in [n]) \oplus (x_1 z, \ldots, x_n z) \oplus (z^2), z \in \mathbb{R}^r,
\]

and \( M_\ell := f(x^{(\ell)}, z^{(\ell)}) \quad \forall \ell \in [m], \)

where \( u \oplus v \) refers to the concatenation of vectors \( u \) and \( v \), and \( M_\ell \) represents the row \( \ell \) of \( M \). In other words

\[
f(x, z) = (1, x_1, \ldots, x_n, x_1^2, \ldots, x_j x_k, \ldots, x_n^2, x_1 z, \ldots, x_j z, \ldots, x_n z, z^2, z),
\]

where \( x_j \) is the \( j \)th component of \( x \) and \( z = x^T A^T x \). Observe that the “intended” polynomial \( q^*(x, z) = z - x^T A^T x \) is a valid solution to this system of equations. Hence, it will suffice to prove that \( M \) has rank exactly \( r - 1 \) i.e., \( M \) has full column rank minus one. First observe that as polynomials over the formal variables \( x, z \), all but one of the columns of \( f(x, z) \) are linearly independent – in fact the only linear dependency in \( f(x, z) \) corresponds to the column \( z \) that can be expressed as a linear combination of degree-2 monomials \( \{ x_j x_k : i < j \} \) since \( z := x^T A^T x \) is a homogenous degree-2 polynomial. Further the columns \( \{ x_j z : j \in [n] \} \) have degree 3 and \( z^2 \) has degree 4. Hence excluding the column corresponding to \( z \), it is easy to see that the rest of the columns are linearly independent (either they correspond to different monomials, or the degrees are different). Now define \( g(x, z), M' \) analogously to \( f(x, z) \) and \( M \) respectively, without the last column that corresponds to \( z \) i.e.,

\[
g(x, z) := (1) \oplus (x_1, \ldots, x_n) \oplus (x_j : i \leq j \in [n]) \oplus (x_1 z, \ldots, x_n z) \oplus (z^2) \in \mathbb{R}^{r-1},
\]

and \( M'_\ell := g(x^{(\ell)}, z^{(\ell)}) \quad \forall \ell \in [m]. \)

From our earlier discussion, the columns of \( g(x, z) \) when seen as polynomials over the formal variables \( x, z \) are linearly independent. Hence, it suffices to prove the following claim:

Claim: \( M' \) has full column rank i.e., rank of \( M' \) is \( r \).

To see why the claim holds consider the first \( \ell \) rows of the matrix \( M' \) and look at their span \( S(R_\ell) \). If \( \ell \leq r - 1 \) then the space orthogonal to \( S(R_\ell) \) i.e., \( S(R_\ell)^\perp \) is non-empty. Consider any direction \( v \) in \( S(R_\ell)^\perp \).

\[
\langle v, M_{\ell+1}' \rangle = \hat{q}(x^{(\ell+1)}, z^{(\ell+1)}), \quad \text{where } \hat{q}(x, z) := \langle v, g(x, z) \rangle
\]

is a non-zero polynomial of degree 2 in \( x, z \) (it is not identically zero because the columns of \( g(x, z) \) are linearly independent as polynomials over \( x, z \)). Hence using a standard result about multivariate polynomials evaluated at randomly chose points (See Fact 25), we get that \( \hat{q}(x^{(\ell+1)}, z^{(\ell+1)}) \neq 0 \) and so \( \langle v, M_{\ell+1}' \rangle \neq 0 \) with probability 1. Taking a union bound over all the \( \ell \in \{ 1, \ldots, r \} \) completes the proof.
**Fact 25** A non-zero multivariate polynomial \( p : \mathbb{R}^n \rightarrow \mathbb{R} \) evaluated at a point \( x \sim N(0, \rho^2)^n \) with \( \rho > 0 \) evaluates to zero with zero probability.

We remark that the statement of Lemma 24 can also be made robust to inverse polynomial error by using polynomial anti-concentration bounds (e.g., Carbery-Wright inequality) instead of Fact 25; however this is not required for proving NP-hardness. We now complete the proof of Theorem 14.

**Proof** [Proof of Theorem 14] We start with the NP-hardness of \( QP \), and for the reduction in Figure 3, we will show that in the YES case, we will show that there is a \( \delta \)-robust degree-2 PTF (completeness), and in the NO case we will show that there is no \( \delta \) robust degree-2 PTF (soundness). As a reminder, the NP-hard problem \( QP \) is the following: given a symmetric matrix \( A \in \mathbb{R}^{n \times n} \) with zeros on diagonals, and \( \beta > 0 \) distinguish whether

**No Case**: there exists an assignment \( y^* \) with \( \|y^*\|_\infty \leq 1 \) such that \( q(y^*) = (y^*)^T Ay^* > \beta \),

**Yes Case**: \( \max_{\|y\|_\infty \leq 1} y^T Ay < \beta \).

**Completeness (Yes Case)**: Consider the degree-2 PTF given by \( \text{sgn}(z - p(x)) = \text{sgn}(z - x^T Ax) \). From Lemma 21, we have that it is \( \delta \) robust at the \( 2m \) points \( \{(u^{(\ell)}, z_u^{(\ell)}), y_u^{(\ell)} \} : \ell \in [m] \} \) and \( \{(v^{(\ell)}, z_v^{(\ell)}), y_v^{(\ell)} \} : \ell \in [m] \} \) with probability at least \( 1 - \eta \) (for \( \eta \) being any sufficiently small constant). Further, from multilinearity of \( p \) we have that,

\[
\max_{\|y\|_\infty \leq \delta} y^T Ay = \delta^2 \max_{\|y\|_\infty \leq 1} y^T Ay < \delta^2 \beta = \alpha - \delta.
\]

Hence \( (\alpha - \delta) - \max_{\|y\|_\infty \leq \delta} y^T Ay > 0 \),

which establishes robustness at \( ((0, \alpha), +1) \) for \( \text{sgn}(z - x^T Ax) \). Hence \( \text{sgn}(z - p(x)) \) is \( \delta \)-robust at the \( 2m + 1 \) points with probability at least \( 1 - \eta \) (for \( \eta \) being any sufficiently small constant).

**Soundness (No Case)**: From Lemma 22, we see that the degree-2 PTF given by \( \text{sgn}(z - p(x)) = \text{sgn}(z - x^T Ax) \) is the only degree-2 PTF that can potentially be robust at all the \( 2m + 1 \) points with probability 1. Again analyzing robustness at the example \( ((0, \alpha), +1) \), we see that from multilinearity of \( p \),

\[
\max_{\|y\|_\infty \leq \delta} y^T Ay = \delta^2 \max_{\|y\|_\infty \leq 1} y^T Ay > \delta^2 \beta = \alpha - \delta.
\]

Hence \( (\alpha - \delta) - \max_{\|y\|_\infty \leq \delta} y^T Ay < 0 \),

which shows that the degree-2 PTF \( \text{sgn}(z - p(x)) \) is not robust at \( (0, \alpha) \). Hence there is no \( \delta \)-robust degree-2 PTF at the \( 2m + 1 \) given points, with probability 1. This completes the analysis of the reduction, and establishes the theorem.

\[\blacksquare\]

**Stronger Hardness.** We now prove Theorem 16. As before, this will follow from the corresponding non-distributional version.

**Theorem 26** [Stronger Hardness] For every \( \delta > 0 \) and \( \varepsilon \in (0, \frac{\delta}{2}) \), assuming \( NP \neq RP \) there is no polynomial time algorithm that given a set of \( N = \text{poly}(n, 1/\varepsilon) \) labeled points \( \{(x^{(1)}, y^{(1)}), \ldots , (x^{(N)}, y^{(N)})\} \) in \( \mathbb{R}^{n+1} \times \{-1, 1\} \) such that there is a degree-2 PTF with \( \delta \)-robust empirical error of 0, can output a degree-2 PTF that has \( \delta \)-robust empirical error of at most \( \varepsilon \) on these \( N \) points.
Proof The proof of this theorem closely follows the proof of Theorem 14 (the $\varepsilon = 0$ case), so we only point out the differences here. The reduction uses the same gadget (Figure 3) used in Theorem 14. The main challenge is the soundness analysis (NO case), where we need to rule out the existence of degree-2 PTFs which are $\delta$-robust and consistent at all but an $\varepsilon$ fraction of the points. To handle this, we introduce “redundancy” by including more points (of both kinds) to ensure that even when an arbitrary $\varepsilon$ fraction of these points are ignored (the PTF makes errors on them), we can still use the arguments in the soundness analysis of Theorem 14.

Recall that our reduction (see Figure 3) generated two sets of points. We have one point of the form $(0, \alpha)$ (let us denote this type as Type A) and $m$ pairs of points \{ $(u^{(\ell)}, z_u^{(\ell)}), (v^{(\ell)}, z_v^{(\ell)}) : \ell \in [m]$ \} which are obtained by modifying $(x^{(\ell)}, z^{(\ell)} = p(x^{(\ell)}))$ with $x^{(\ell)}$ generated randomly (let us denote these $2m$ points as of Type B).

Set $N_1 := n^3, N_2 := 2n^3$. In our modified instance, we will have $N_1$ points of Type A i.e., $N_1$ identical points $(0, \alpha)$ (note that we can also perturb these points slightly so that they are all distinct, if required). Further, we will have $N_2$ points of Type B i.e., we will generate $N_2/2$ pairs of points \{ $(u^{(\ell)}, z_u^{(\ell)}) : \ell \in \lfloor N_2/2 \rfloor$ \} which are generated as described in Figure 3 after drawing $x^{(\ell)} \sim N(0, \rho^2)^n$ for $\ell \in \lfloor N_2/2 \rfloor$ (here a larger $\rho = O(\delta n^3/2 N_2)$ will suffice). Hence, we have in total $N = N_1 + N_2 = 3n^3$ points.

The completeness analysis (YES case) is identical to that of Theorem 14, as $\sgn(z - p(x))$ will be $\delta$-robust at all of the $N$ points (from Lemma 21 and our choice of $\alpha$).

We now focus on the soundness analysis (NO case). From $\varepsilon < \frac{1}{3}$ and our choice of $N_1$ and $N_2$.

\[
N_1 \geq \varepsilon(N_1 + N_2)
\]

\[
(1 - \varepsilon)(N_1 + N_2) > N_1 + \frac{N_2}{2} + (n + 1)^2
\]

From (21) and a pigeonhole argument, any subset of size $(1 - \varepsilon)(N_1 + N_2)$ is guaranteed to have $(n + 1)^2$ pairs of points of the form $(u^{(\ell)}, z_u^{(\ell)})$ and $(v^{(\ell)}, z_v^{(\ell)})$. This is because the LHS of (21) represents a lower bound on the number of points the candidate degree-2 PTF is robust on. The RHS of (21) represents the number of points needed to ensure that at least $(n + 1)^2$ pairs of points from Type B are picked. Hence using Lemma 22 along with a union bound over all the \{ $N_2 \choose (n+1)^2$ \} choices of the pairs (note that the failure probability in Lemma 24 is 0), the “intended” PTF $\sgn(z - p(x))$ is the only surviving degree-2 PTF.

Again from (20) and the pigeonhole principle, any $(1 - \varepsilon)$ fraction of the points will contain at least one point of the Type A i.e., $(0, \alpha)$. Hence in the NO case, the “intended” PTF $\sgn(z - p(x))$ is not $\delta$-robust. This completes the soundness analysis and establishes the theorem. ■