NORTHWESTERN UNIVERSITY

Learning under Adversarial Resilience

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Mathematics

By

Abhratanu Dutta

EVANSTON, ILLINOIS

December 2020

© Copyright by Abhratanu Dutta 2020

All Rights Reserved

ABSTRACT

Learning under Adversarial Resilience

Abhratanu Dutta

In this thesis we study two problems, one in unsupervised learning - k-means clustering and the other in a supervised learning setting with the presence of adversarial perturbations. We do a beyond-worst case style analysis and show that in either case instances that are resilient to adversarial perturbations are also tractable.

Resilience to adversarial perturbations imply margin conditions of different flavors. For the k-means problem, the resilience comes in the form of an assumption related to Bilu-Linial stability. We give an efficient algorithm for the k-means problem under the assumption that perturbing each point slightly does not alter the optimal clustering. This assumption implies that there exists an angular margin between any two clusters in an optimal clustering.

In the adversarial learning problem we assume realizability, i.e. there exists a perfect classifier that is resilient to adversarial perturbations. This in turn implies that this perfect classifier has a margin and we show that under this assumption we can achieve adversarial robustness.

We also show hardness of approximation in the adversarial learning setting. One of the perhaps surprising observation of this thesis is the role of the adversarial resilience assumption in making NP-hard problems tractable in both the supervised and the unsupervised setting.

Acknowledgements

A thesis is never a solo undertaking and I want to take the time to acknowledge everyone that made this document possible. First and foremost, I am greatly indebted to my advisor Aravindan Vijayaraghavan. I have always been awed at the breadth of his knowledge and the depth of his insight. Over the years he guided me through the research landscape patiently and with understanding. He has always encouraged me to find exciting new problems to work on and follow them through. I could not have asked for a better advisor and I'm grateful to be his first PhD student.

I would like to thank my collaborators Alex Wang and Pranjal Awasthi. It was really enjoyable working with Alex and he has become a great friend since. I have learnt a lot from working with Pranjal and he has become a mentor in my research journey. I would also like to thank Avrim Blum for numerous useful discussions regarding our clustering paper.

I am thankful to Aravindan, Pranjal and Kostya for serving in my PhD committee. I would like to thank the faculty at the Theory department at Northwestern over the years, particularly Jason, Anindya and Kostya for their encouragement and very constructive feedback in all my talks and presentations. I have learnt a lot from them. I would like to thank Jake Abernathy for inviting me to Georgia Tech to work on our project.

I would also like to thank Sam, Aleck, Yiding, Aravind, Liren, Madhav, Johes and all my other friends in the CS Department for making this journey that much more interesting and

enjoyable. Special thanks to Sushobhan, Khadija, and Vineet for sticking with me throughout the years and the very fun times. I am thankful to my friends in Ames. They gave me a sense of community and belonging.

I want to thank June Ghosh for being the awesome friend that she is and for being my support system. I feel blessed to have shared all the ups and downs of grad school and life with her.

I am thankful to my sister Priyanka and Agnivo for their affection and encouragement. Finally I would like to thank Ma and Bapi for their incredible love and support. This dissertation would not have been possible without them.

Table of Contents

List of Tables

List of Figures

The images in the first column represent the original images corresponding to three, the second column represents the perturbed images produced by the successful PGDattack, and perturbed images produced by the successful SDPattack. Visual inspection of these examples suggest that our method often produces sparse targeted perturbations. 110

CHAPTER 1

Introduction

In practice, we often care about problems that are NP-hard to solve, for which we do not expect to find worst-case efficient algorithms. While theoretically we have various tools like approximation algorithms to tackle these problems, in the real world these often fall short and we resort to heuristics. These heuristics while being very inefficient in the worst-case seem to work really well in real life. On the other hand, often we do not have the necessary theoretical techniques needed to explain why these heuristics work well in practice. A possible explanation to this is that the generic "worst-case" complexity view of the world is too bleak - we rarely ever face worst-case instances. We need a new way to measure performance of and distinguish between algorithms for hard problems and this is where beyond worst case complexity comes in.

Average case analysis also often provides brittle and non-robust solutions that depend crucially on the exact distributional assumptions. Beyond worst case analysis fills this gap between worst-case and purely average-case analysis. In the words of Tim Roughgarden [Roughgarden, 2018], beyond worst case analysis is about *"articulating properties of 'real-world' inputs, and proving rigorous and meaningful algorithmic guarantees for inputs with these properties.*

This thesis considers important questions in the frontiers of beyond worst-case analysis both in the unsupervised and the supervised regime. In particular it considers two NeurIPS

papers Dutta, Vijayaraghavan and Wang '17 [Dutta et al., 2017] and Awasthi, Dutta and Vijayaraghavan '19 [Awasthi et al., 2019b]. The first one explores beyond worst case analysis in the ubiquitous k-means clustering problem and comes up with a new algorithm with provable guarantees and experiments to validate the results. The second work explores the topical subject of adversarial learning and draws a perhaps surprising connection to polynomial optimization. This thesis does further experimentation and goes on to ask some important follow up questions on how to generalize these results to bring them closer to practice and applicability in the real world.

In the following section, we define a natural property that we expect real-world instances to have : instance stability, and give an efficient algorithm for k-means clustering for instances with such properties. Then we explore an intriguing connection between beyond worst case notions like stability and the realizability assumption in the adversarial learning setting.

The Instance Stability Assumption

The instance stability assumption was first introduced by Bilu and Linial [2010] in a seminal paper. They put forward the argument that in order to prove a problem like clustering is NP-hard we reduce say SAT to some clustering instances. This proves that an algorithm that solves these clustering instances also solves SAT. However all these hard instances may be of no practical interest at all. And this does not preclude the existence of an algorithm that solves all the practical instances of clustering that might crop up in the real world.

Bilu and Linial's assumption is a way to formalize the notion that the optimal solution does not change much with small changes in the input. Their original paper tackles the

Max-Cut problem which they think of as "clustering into two clusters". With respect to Max-Cut, they define γ stability as follows :

Definition 1.0.1 (Bilu and Linial)**.** Let W be an $n \times n$ symmetric, non-negative matrix. A γ-perturbation of W, for $\gamma \geq 1$, is an $n \times n$ matrix W' such that $\forall i, j = 1, \cdots, n, W_{i,j} \leq W'_{i,j} \leq \gamma \cdot W_{i,j}$. Let $(S, [n] \setminus S)$ be a maximal cut of W, i.e. a partition that maximizes $\sum_{i \in S, j \notin S} W_{i,j}$. The instance W (of the Max-Cut problem) is said to be γ-stable, if for every γ-perturbation W' of W, $(S, [n] \setminus S)$ is the unique maximal cut of W'.

To rule out spurious examples, they need another assumption that there are no small cuts in the graph. Under these assumptions, they find an efficient algorithm for solving Max-Cut instances. They however concede that this formulation is a first step and "it is of great interest to study more permissive notions of stability where a small perturbation can slightly modify the optimal solution." The additive perturbation notion of stability that we define in our work is inspired by this formulation and tries to maintain the spirit of the assumption.

There has been a lot of further work with the instance stability assumption on various NP-hard problems. These include works on MAP inference [Lang et al., 2018], Minimum Multiway Cut [Makarychev et al., 2013] and clustering [Awasthi et al., 2010]. We give a brief overview of the works on stability with respect to clustering in the next section.

1.1. Case Study : Stability and Clustering

Clustering is an unsupervised learning problem, where given a set of points in some high dimensional space, we have to partition them into *coherent groups*. This is quantified by defining an objective function that we want to minimize. In theory, most natural formulations

of the objective function are NP-hard to solve. However, in practice, heuristics like the Lloyd's algorithm consistently recover the intuitively correct solution, i.e. *ground truth*. How do we reconcile this difference between theory and practice?

Clustering is difficult only when it does not matter [Ackerman and Ben-David, 2009] has been posited as a possible explanation. That is, the hard instances of clustering does not appear organically and the real-world instances have additional structure that we should explore and exploit. Bilu and Linial [Bilu and Linial, 2010]'s instance stability is a way to formalize this notion. Broadly speaking, the idea is that the optimum solution to a problem shouldn't change too much if the input is perturbed a little. In the context of clustering, this has been studied in multiple papers as multiplicative perturbation stability [Awasthi et al., 2010] [Makarychev and Makarychev, 2016].

Definition 1.1.1. Multiplicative Perturbation Stability : For any $\gamma \geq 1$, a metric clustering instance (X, d) on point set $X \subset R^d$ and metric $d : X \times X \to R^+$ is said to be γ-factor (multiplicative) stable iff the (unique) optimal clustering $C_1, \cdots, C_k$ of X remains the optimal solution for any instance (X, d') where any (subset) of the the distances are increased by up to a γ factor i.e., $d(x, y) \leq d'(x, y) \leq \gamma d(x, y)$ for any $x, y \in X$.

1.2. Our Contributions for Stable Clustering

While this formulation has multiple advantages, it has a few disadvantages as well. Specifically, in practice the assumption often turns out to be too strong and the algorithms that we know even in these strong cases are often non-robust. In our 2017 work [Dutta et al., 2017], we formulated a more natural notion of instance stability for clustering in the Euclidean space. It starts by defining an ε additive perturbation.

Definition 1.2.1. ε-additive perturbation : Let $X = \{x_1, \cdots, x_n\}$ be a k-means clustering instance with unique optimal clustering $C_1, \cdots, C_k$ whose means are given by $\mu_1, \cdots, \mu_k$. Let $D = \max_{i,j} \|\mu_i - \mu_j\|$. We say that $X' = \{x'_1, \cdots, x'_n\}$ is an ε-additive perturbation of X if for all i, $\|x'_i - x_i\| \leq \varepsilon D$

Now we can define ε-additive perturbation stability as the property that the optimal clustering remains unchanged under any ε-additive perturbation.

Definition 1.2.2. (ε-additive perturbation stability). Let X be a k-means clustering instance with unique optimal clustering $C_1, C_2, \cdots, C_k$. We say that X is ε-additive perturbation stable (APS) if every ε-additive perturbation of X has an optimal clustering given by $C_1, C_2, \cdots, C_k$.

This is a natural interpretation of the crux of Bilu-Linial stability in Euclidean domains. For instance, it captures measurement errors which usually produces such additive errors. And we want our clustering to be well-defined enough so that it remains unchanged under small measurement errors.

For the theoretical results, we also required some ρ-amount of margin separation between the optimal clusters. We defined this set of assumptions as $(\rho, \Delta, \varepsilon)$-separation and gave an efficient algorithm that found the optimal k-means clustering.

We also extended the results to the robust setting, where we have the required guarantee on only a fraction of data points. We complement the theoretical results with experimental results on real world datasets where we compare our algorithm to k-means++ and show comparable performance .

1.3. Case Study : Adversarial Learning

The field of adversarial learning is quickly becoming very relevant as we deploy more machine learning systems in the real world. We often find that these systems are very non-robust to adversarial perturbations. This was first identified by [Szegedy et al., 2013]. In recent times a group of researchers at Keen Security Labs managed to confuse Tesla's self-driving system into driving onto oncoming traffic by just placing three stickers on the road. There are plenty of other real world examples - adversarial sunglasses that makes the wearer's face undetectable or intentionally misclassified, a 3D printed turtle that is recognized as a rifle from all angles and so on.

The setting is as follows : we first train a classifier on good training data. During test time, an adversary that has access to our classifier (white-box) makes a small perturbation on the test input from x to $x + \varepsilon$. We want our classifier f to return a correct answer on the perturbed input, i.e. $f(x + \varepsilon) = y$ where y is the correct label for x. In this work, there is a key assumption we make - *realizability*. That is, we assume that exists a classifier that is robust to ε adversaries. The question then becomes, how robust a classifier can we find efficiently?

Although the previous work and this work looks quite different at first sight, they have a few common themes. Somewhat surprisingly, the well known assumption of realizability is related to the assumption of stability in this case. In the first paper we proved that the assumption of stability implies the existence of an angular margin between the different clusters. On the other hand, the existence of a ε-adversarially robust classifier implies a margin separation of 2ε between any two classes. The first one is unsupervised, so we just

find a perfect clustering of the data. In the second work however, we have access to training data and so we try to find something stronger - an adversarially robust perfect classifier.

1.4. Our contributions for Adversarial Learning

In this work, we make four main contributions as follows : first, for degree 1 and degree 2 polynomial threshold functions(PTFs) we find the first provably efficient algorithms that finds an adversarial example when one exists.

Secondly, we use these algorithms designed above for finding adversarial examples to design the first optimal approximately robust polynomial time learning algorithms for the class of degree-1 and degree-2 polynomial threshold functions (PTFs).

We also show a matching hardness of approximation bound for the above result derived from the hardness of quadratic programming(QP) problem [Charikar and Wirth, 2004].

Theorem 1.4.1. *There exists $\delta > 0$, and a distribution D over $\mathbb{R}^n \times \{-1, +1\}$ and $\varepsilon > 0$, such that assuming $NP \neq RP$ there is no polynomial time algorithm that given a set of $poly(n, \frac{1}{\varepsilon})$ points from D labeled by a degree-2 PTF that has δ-robust error of 0 w.r.t. D, outputs a degree-2 PTF of $o(\sqrt{\eta_{approx}}\delta)$-robust error at most ε w.r.t. D, where η_{approx} is the hardness of approximation factor of the* QP *problem.*

Finally, we show that we can leverage the connection to polynomial optimization to generate adversarial attacks on neural networks. For 2-layer networks with ReLU activations, we show that given a network and a test input, the problem of finding an adversarial example corresponds to a natural optimization problem. We design a semi-definite programming (SDP) based polynomial time algorithm to generate an adversarial attack for such networks. We

show that under a natural condition on the structure of the SDP solution, our attack provably finds an adversarial example or certifies that none exists. We empirically show that condition holds true in practice and compare our attack to the state-of-the-art attack of Madry et al. [Madry et al., 2017] on the MNIST data set.

1.5. Organization of the Thesis

First we mention some preliminaries of Stable Clustering and Adversarial Learning in Chapter 2. We also provide our model and discuss some related work. The rest of the thesis is broadly split into two parts. In Chapter 3 we discuss k-means clustering under a Bilu-Linial stability assumption. We give algorithms for $k = 2$, general k and robust k-means setting. The second part is Chapters 4 and 5. In Chapter 4 we give algorithms for adversarial learning of degree-2 PTFs with theoretical guarantees and for two layer neural networks. In Chapter 5 we give a matching hardness of approximation proof degree-2 PTFs. We also give a lower bound for robust weak learning. We finally end the thesis with experimental results of both stable clustering and adversarial learning in Chapter 6.

CHAPTER 2

Background

2.1. Stable Clustering

2.1.1. Models and Preliminaries

In the k-means clustering problem, we are given n points $X = \{ x_1, \ldots, x_n \}$ in $\mathbb{R}^d$ and need to find k centers $\mu_1, \ldots, \mu_k \in \mathbb{R}^d$ minimizing

$$\sum_{x \in X} \min_{i \in [k]} \|x - \mu_i\|^2 .$$

A given choice of centers $\mu_1, \ldots, \mu_k$ determines an optimal clustering $C_1, \ldots, C_k$ where $C_i = \{ x \mid i = \arg\min_j \|x - \mu_j\| \}$. We can rewrite the objective as

$$\sum_{i \in [k]} \sum_{x \in C_i} \|x - \mu_i\|^2 .$$

On the other hand, a given choice for cluster C_i determines its optimal center as $\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$, the mean of the points in the set. Thus, we can reformulate the problem as minimizing over clusters $C_1, C_2, \ldots, C_k$ of $\{ x_i \}$ the objective

$$\sum_{i \in [k]} \sum_{y \in C_i} \left\| y - \left(\frac{1}{|C_i|} \sum_{x \in C_i} x \right) \right\|^2 .$$

k-means clustering is NP-hard for general Euclidean space $\mathbb{R}^d$ even in the case of $k = 2$ [Dasgupta, 2008].

2.1.2. Related Work

Awasthi et al. [2012] showed that γ-multiplicative perturbation stable instance also satisfied the notion of γ-center based stability (every point is a γ-factor closer to its center than to any other center). They showed that an algorithm based on the classic single linkage algorithm works under this weaker notion when $\gamma \geq 3$. This was subsequently improved by [Balcan and Liang, 2011], and the best result along these lines [Angelidakis et al., 2017] gives a polynomial time algorithm that works for $\gamma \geq 2$. A robust version of (γ, η)-perturbation resilience was explored for center-based clustering objectives [Balcan and Liang, 2011]. As such, the notions of additive perturbation stability, and $(\rho, \Delta, \varepsilon)$-separated instances are incomparable to the various notions of multiplicative perturbation stability. Further as argued in [Ben-David, 2015], we believe that additive perturbation stability is more realistic for Euclidean clustering problems.

Ackerman and Ben-David [2009] initiated the study of various deterministic assumptions for clustering instances. The measure of stability most related to this work is Center Perturbation (CP) clusterability (an instance is δ-CP-clusterable if perturbing the centers by a distance of δ does not increase the cost much). A subtle difference is their focus on obtaining solutions with small objective cost[Ackerman and Ben-David, 2009], while our goal is to recover the optimal clustering. However, the main qualitative difference is how the length scale is defined — this is crucial for additive perturbations. The run time of the algorithm in[Ackerman and Ben-David, 2009] is $n^{\mathrm{poly}(k, L(X)/\delta)}$, where the length scale

of the perturbations is $L(X)$, the diameter of the whole instance. Our notion of additive perturbations uses a much smaller length-scale of Δ (essentially the inter-mean distance; see Prop. 1.1 for a geometric interpretation), and Theorem 1.2 gives a run-time guarantee of $n^{\mathrm{poly}(\Delta/\delta)}$ for $k = 2$ (Theorem 1.2 is stated in terms of $\varepsilon = \Delta/\delta$). By using the largest inter-mean distance instead of the diameter as the length scale, our algorithmic guarantees can also handle unbounded clusters with arbitrarily large diameters and outliers.

The exciting results of Kumar and Kannan [2010] and Awasthi and Sheffet [2012] also gave a deterministic margin-separation condition, under which spectral clustering (PCA followed by k-means) [1] finds the optimum clusters under deterministic conditions about the data. Suppose $\sigma = \|X - C\|_{op}^2/n$ is the "spectral radius" of the dataset, where C is the matrix given by the centers. In the case of equal-sized clusters, the improved results of [Awasthi and Sheffet, 2012] proves approximate recovery of the optimal clustering if the margin ρ between the clusters along the line joining the centers satisfies $\rho = \Omega(\sqrt{k}\sigma)$. Our notion of margin ρ in $(\rho, \Delta, \varepsilon)$-separated instances is analogous to the margin separation notion used by the above results on spectral clustering [Awasthi and Sheffet, 2012; Kumar and Kannan, 2010]. In particular, we require a margin of $\rho = \Omega(\Delta/\varepsilon^2)$ where Δ is our scale parameter, with no extra $\sqrt{k}$ factor. However, we emphasize that the two margin conditions are incomparable, since the spectral radius σ is incomparable to the scale parameter Δ.

We now illustrate the difference between these deterministic conditions by presenting a couple of examples. Consider an instance with n points drawn from a mixture of k Gaussians in d dimensions with identical diagonal covariance matrices with variance 1 in the first $O(1)$ coordinates and roughly $1/d$ in the others, and all the means lying in the subspace spanned

[1] This requires appropriate initializers, that they can obtain in polynomial time.

by these first $O(1)$ co-ordinates. In this setting, the results of [Awasthi and Sheffet, 2012; Kumar and Kannan, 2010] require a margin separation of at least $\sqrt{k \log n}$ between clusters. On the other hand, these instances satisfy our geometric conditions with $\varepsilon = \Omega(1)$, $\Delta\ \sqrt{\log n}$ and therefore our algorithm only needs a margin separation of $\rho\sqrt{\log n}$ (hence, saving a factor of $\sqrt{k}$)[2]. However, if the n points were drawn from a mixture of spherical Gaussians in high dimensions (with $d \gg k$), then the margin condition required for [Awasthi and Sheffet, 2012; Kumar and Kannan, 2010] is weaker.

We note another strand of recent works show that convex relaxations for k-means clustering become integral under distributional assumptions about points and sufficient separation between the components [Awasthi et al., 2014; Mixon et al., 2016].

Finally, we mention some very recent work on hardness of multiplication stable clustering instances that assume a plausible PCP hypothesis [Friggstad et al., 2018] which was subsequently proved in [Paradise, 2020].

2.2. Adversarial Learning

2.2.1. Preliminaries and Model

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

[2]Further, while algorithms for learning GMM models may work here, adding some outliers far from the decision boundary will cause many of these algorithms to fail, while our algorithm is robust to such outliers.

Given a binary classifier $\text{sgn}(g(x))$, an input x^*, and a budget $\delta > 0$, we say that $x^* + z$ is an *adversarial example* (for input x^*) if $\text{sgn}(g(x^* + z)) \neq \text{sgn}(g(x^*))$ and that $\|z\|_\infty \leq \delta$. One could similarly define the notion of adversarial examples for other norms. For a classifier with multiple outputs, we say that $x^* + z$ is an adversarial example iff the largest co-ordinate of $g(x^* + z)$ differs from the largest co-ordinate of $g(x^*)$. We now define the notion of robust error of a classifier.

Definition 2.2.1 (δ-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 δ-robust error of f with respect to D as $err_{\delta,D}(f) = \mathbb{E}_{(x,y)\sim D}\big[\sup_{z\in B^n_\infty(0,\delta)} \ell(f(x+z), y)\big]$. Here $B^n_\infty(0, \delta)$ denotes the ℓ_∞ ball of radius δ, i.e., $B^n_\infty(0, \delta) = \{x \in \mathbb{R}^n : \|x\|_\infty \leq \delta\}$.

Analogous to empirical error in PAC learning, we denote $\hat{err}_{\delta,S}(f)$ to be the δ-robust empirical error of f, i.e., the robust error computed on the given sample S. To bound generalization gap, we will use the notion of adversarial VC dimension as introduced in [Cullina et al., 2018]. Next we define robust learning for PTFs.

Definition 2.2.2 (γ-approximately robust learning)**.** Let $\mathcal{F}$ be the class of degree-d PTFs from $\mathbb{R}^n \mapsto \{-1, 1\}$ of VC dimension $\Delta = O(n^d)$. For $\gamma \geq 1$, an algorithm $\mathcal{A}$ γ-approximately robustly learns $\mathcal{F}$ 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 $\mathcal{F}$ contains a function f^* such that $err_{\delta,D}(f^*) = 0$, then with probability at least $1 - \eta$, $\mathcal{A}$ runs in time polynomial in m and outputs $f \in \mathcal{F}$ such that $err_{\delta/\gamma,D}(f) \leq \varepsilon$. If $\mathcal{F}$ admits such an algorithm then we say that $\mathcal{F}$ is γ-approximately robustly learnable. Here γ quantifies the price of achieving computationally efficient robust learning, with $\gamma = 1$ implying optimal learnability.

A note about the model and the realizability assumption. Our definition of an adversarial example requires that $\text{sgn}(g(x^* + z)) \neq \text{sgn}(g(x^*))$, whereas for robust learning we require a classifier that satisfies $\text{sgn}(g(x^* + z)) \neq y$, where y is the given label of x^*. This might create two sources of confusion to the reader: a) In general the two requirements might be incompatible, and b) It might happen that initially $\text{sgn}(g(x^*))$ predicts the true label incorrectly but there is a perturbation z such that $\text{sgn}(g(x^* + z))$ predicts the true label correctly. In this case one may not count z as an adversarial example. To address (a) we would like to stress that all our guarantees hold under the realizability assumption, i.e., we assume that there is true function c^* such that for all examples x in the support of the distribution and all perturbations of magnitude upto δ, $\text{sgn}(c^*(x^* + z)) = \text{sgn}(c^*(x^*))$. Hence, there will indeed be a target concept for which no adversarial example exists and as a result will have zero robust error. To address (b) we would like to point out that in Section 4.2 where we use the subroutine for finding adversarial examples to learn a good classifier $\text{sgn}(g)$, we always enforce the constraint that on the training set $\text{sgn}(g(x^*)) = \text{sgn}(c^*(x^*))$ and g is as robust as possible. Hence when we find an adversarial example for a point x^* in our training set, it will indeed satisfy that $\text{sgn}(g(x^* + z)) \neq \text{sgn}(c^*(x))$ and correctly penalize g for the mistake. More generally, we could also define an adversarial example as one where given pair (x^*, y) the goal is to find a z such that $\text{sgn}(g(x^* + z)) \neq y$. All of our guarantees from Section 4.1 apply to this definition as well. Finally, in the non-realizable case, the distinction between defining adversarial robustness as either $\text{sgn}(g(x^* + z)) \neq \text{sgn}(g(x^*))$, or $\text{sgn}(g(x^* + z)) \neq y$, or even $\text{sgn}(g(x^* + z)) \neq \text{sgn}(c^*(x))$ matters and has different computational and statistical implications [Diochnos et al., 2018; Gourdeau et al., 2019]. Understanding when one can

achieve computationally efficient robust learning in the non-realizable case is an important direction for future work.

The definition of γ-approximately robustly learnability has the realizability assumption built into it. So, when we prove that a class $\mathcal{F}$ is γ-approximately robustly learnable, we find an approximate robust learner from $\mathcal{F}$ under the realizability assumption on $\mathcal{F}$ i.e. for a set of points from the distribution, the algorithm guarantees to return an approximate robust learner only if there exists a perfect robust learner in the class $\mathcal{F}$ of learners.

The work of [Cullina et al., 2018] defines the notion of adversarial VC dimension to bound the generalization gap for robust empirical risk minimization. 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 2.2.3. *Let $\mathcal{F}$ be a class of degree-d polynomial threshold functions from $\mathbb{R}^n \mapsto \{-1, 1\}$ of VC dimesion $\Delta = O(n^d)$. 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 $\sup_{f \in \mathcal{F}} |err_{\delta,D}(f) - \hat{err}_{\delta,S}(f)| \leq 2\sqrt{2\Delta \log m/m} + \sqrt{\log(1/\eta)/(2m)}$.*

2.2.2. 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 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 and Mannor, 2012; Xu et al., 2009] 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 [Attias et al., 2018; Feige et al., 2015] 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 [Diochnos et al., 2018; Mahloujifar and Mahmoody, 2018; Mahloujifar 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 [Raghunathan et al., 2018; Sinha et al., 2017; Wong and Kolter, 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.

The work of Bubeck et al. [Bubeck et al., 2018a,b] provides a cryptographic lower bound by designing 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 γ-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.

CHAPTER 3

Stable Clustering

In this chapter, in Section 3.1 we formally state our notion of stability and define parameters that capture this notion. We also provide some geometric intuition and explain what this notion entails. Then we move on to stable clustering algorithms and theoretical guarantees for k means (k = 2 and general k) in Sections 3.2 and 3.3. We finally end the chapter with a robust k-means algorithm and theoretical guarantees in Section 3.4.

3.1. Stability definitions and geometric properties

3.1.1. Balance parameter

We define an instance parameter, β, capturing how balanced a given instance's clusters are.

Definition 3.1.1 (Balance parameter)**.** Given an instance X with optimal clustering $C_1, \ldots, C_k$, we say X satisfies balance parameter $\beta \geq 1$ if for all $i \neq j$, $\beta|C_i| > |C_j|$.

3.1.2. Additive perturbation stability

Definition 3.1.2 (ε-additive perturbation)**.** Let $X = \{ x_1, \ldots, x_n \}$ be a k-means clustering instance with unique optimal clustering $C_1, C_2, \ldots, C_k$ whose means are given by $\mu_1, \mu_2, \ldots, \mu_k$. Let $D = \max_{i,j} \|\mu_i - \mu_j\|$. We say that $X' = \{ x'_1, \ldots, x'_n \}$ is an ε-additive perturbation of X if for all i, $\|x'_i - x_i\| \leq \varepsilon D$.

Definition 3.1.3 (ε-additive perturbation stability)**.** Let X be a k-means clustering instance with unique optimal clustering $C_1, C_2, \ldots, C_k$. We say that X is ε-additive perturbation stable (APS) if every ε-additive perturbation of X has an optimal clustering given by $C_1, C_2, \ldots, C_k$.

Intuitively, the difficulty of the clustering task increases as the stability parameter ε decreases. For example, when $\varepsilon = 0$ the set of ε-APS instances contains any instance with a unique solution. In the following we will only consider $\varepsilon > 0$.

3.1.3. Geometric implication of ε-APS

Let X be an ε-APS k-means clustering instance such that each cluster has at least 4 points. Fix $i \neq j$ and consider clusters C_i, C_j with means μ_i, μ_j. We fix the following notation.

- Let $D_{i,j} = \|\mu_i - \mu_j\|$ and let $D = \max_{i',j'} \|\mu_{i'} - \mu_{j'}\|$.
- Let $u = \frac{\mu_i - \mu_j}{\|\mu_i - \mu_j\|}$ be the unit vector in the intermean direction. Let $V = u^{\perp}$ be the space orthogonal to u. For $x \in \mathbb{R}^d$, let $x_{(u)}$ and $x_{(V)}$ be the projections x onto u and V.
- Let $p = \frac{\mu_i + \mu_j}{2}$ be the midpoint between μ_i and μ_j.

Clusters in the optimal solution of an ε-APS instance satisfy a natural geometric condition — there is an "angular separation" between every pair of clusters.

Proposition 3.1.4 (Geometric Implication of ε-APS)**.** *Let X be an ε-APS instance and let C_i, C_j be two clusters in its optimal solution. Any point $x \in C_i$ lies in a cone whose axis is along the direction $(\mu_i - \mu_j)$ with half-angle $\arctan(1/\varepsilon)$. Hence if u is the unit vector along*

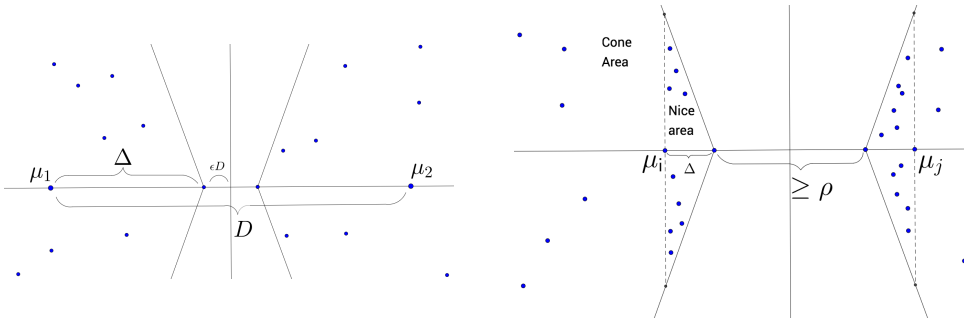

Figure 3.1. **a.** An ε-APS instance. The means are separated by a distance D, the half-angle of each cone is $\arctan(1/\varepsilon)$ and the distance between μ_1 and the apex of the cone $\Delta \leq D/2$. **b.** A $(\rho, \Delta, \varepsilon)$-separated instance with scale parameter Δ. The half-angle of each cone is $\arctan(1/\varepsilon)$ and the distance between the apexes of the cones is at least ρ.

$\mu_i - \mu_j$ *then*

$$\forall x \in C_i, \ \frac{|\langle x - \frac{\mu_i+\mu_j}{2}, u\rangle|}{\|x - \frac{\mu_i+\mu_j}{2}\|_2} > \frac{\varepsilon}{\sqrt{1+\varepsilon^2}}. \tag{3.1}$$

The distance between μ_i and the apex of the cone is $\Delta = (\frac{1}{2} - \varepsilon)D$. We will call Δ the *scale parameter* of the clustering. See Figure 3.1a for an illustration.

We can establish geometric conditions that X must satisfy by considering different perturbations. As an example, one could move all points in C_i and C_j towards each other in the intermean direction a distance of εD; by assumption no point has crossed the separating hyperplane and thus we can conclude the existence of a margin of width $2\varepsilon D$.

A careful choice of a family of perturbations allows us to prove Proposition 3.1.4. Consider the perturbation which moves μ_i and μ_j in opposite directions orthogonal to u while moving a single point towards the other cluster parallel to u (see figure 3.2). The following lemma establishes Proposition 3.1.4.

Lemma 3.1.5. *For any* $x \in C_i \cup C_j$, $\|(x-p)_{(V)}\| \leq \frac{1}{\varepsilon}\left(\|(x-p)_{(u)}\| - \varepsilon D_{i,j}\right)$.

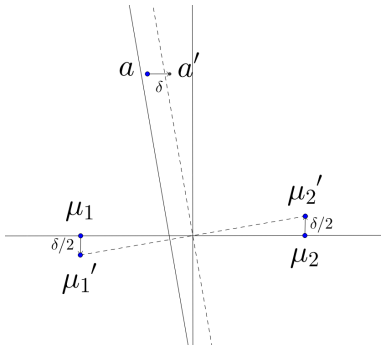

Figure 3.2. An example from the family of perturbations considered by Lemma 3.1.5. Here v is in the upwards direction. If a is to the right of the diagonal solid line, then a' will be to the right of the slanted dashed line and will lie on the wrong side of the separating hyperplane.

Proof. Let $v \in V$ be a unit vector perpendicular to u. Without loss of generality, let $a \in C_i$ (taking u or $-u$ does not change the inequality). Let $b, c, d \in C_i$ such that $a, b, c, d \in C_i$ are distinct. Let $\delta = \varepsilon D_{i,j} \leq \varepsilon D$ and consider the ε-additive perturbation X' given by the union of

$$\{ a - \delta u, b + \delta u, c - \delta v, d - \delta v \} \cup \{ x - \tfrac{\delta}{2} v \mid x \in C_i \setminus \{ a, b, c, d \} \} \cup \{ x + \tfrac{\delta}{2} v \mid x \in C_j \}$$

and an unperturbed copy of $X \setminus (C_i \cup C_j)$.

By assumption, $\{ C_i, C_j \}$ remain optimal clusters in X'. We have constructed X' such that the new means of C_i, C_j are $\mu_i' = \mu_i - \frac{\delta}{2} v$ and $\mu_j' = \mu_j + \frac{\delta}{2} v$, and the midpoint between the means is $p' = p$. The halfspace containing μ_i' given by the linear separator between μ_i' and μ_j' is $\langle x - p', \mu_i' - \mu_j' \rangle \geq 0$. Hence, as a' is classified correctly by the ε-APS assumption,

$$\begin{aligned} \langle a' - p', \mu_i' - \mu_j' \rangle &= \langle a - p - \delta u, D_{i,j} u - \delta v \rangle \\ &= D_{i,j} (\langle a - p, u \rangle - \varepsilon \langle a - p, v \rangle - \delta) \geq 0 \end{aligned}$$

Then noting that $\langle a-p, u\rangle \geq 0$, we have that $\langle a-p, v\rangle \leq \frac{1}{\varepsilon}\left(\|(a-p)_{(u)}\| - \delta\right)$. □

This geometric property follows from perturbations which only affect two clusters at a time. Our results follow from this weaker notion.

3.1.4. $(\rho, \Delta, \varepsilon)$-separation

Motivated by Lemma 3.1.5, we define a geometric condition where the angular separation and margin separation are parametrized separately. These separations are implied by a stronger stability assumption where any pair of clusters is ε-APS with scale parameter Δ even after being moved towards each other a distance of ρ.

We say that a pair of clusters is $(\rho, \Delta, \varepsilon)$-separated if their points lie in cones with axes along the intermean direction, half-angle $\arctan(1/\varepsilon)$, and apexes at distance Δ from their means and at least ρ from each other (see figure 3.1b). Formally, we require the following.

Definition 3.1.6 (Pairwise $(\rho, \Delta, \varepsilon)$-separation)**.** Given a pair of clusters C_i, C_j with means μ_i, μ_j, let $u = \frac{\mu_i - \mu_j}{\|\mu_i - \mu_j\|}$ be the unit vector in the intermean direction and let $p = (\mu_i + \mu_j)/2$. We say that C_i and C_j are $(\rho, \Delta, \varepsilon)$-separated if $D_{i,j} \geq \rho + 2\Delta$ and for all $x \in C_i \cup C_j$,

$$\|(x-p)_{(V)}\| \leq \frac{1}{\varepsilon}\left(\|(x-p)_{(u)}\| - (D_{i,j}/2 - \Delta)\right).$$

Definition 3.1.7 ($(\rho, \Delta, \varepsilon)$-separation)**.** We say that an instance X is $(\rho, \Delta, \varepsilon)$-separated if every pair of clusters in the optimal clustering is $(\rho, \Delta, \varepsilon)$-separated.

3.2. k-means clustering for $k = 2$

In this section, we give an algorithm that is able to cluster 2-means ε-APS instances correctly.

Theorem 3.2.1. *There exists a universal constant $c \geq 1$ such that for any fixed $\varepsilon > 0$, there exists an $n^{O((1/\varepsilon)^c)}d$ time algorithm that correctly clusters all ε-APS 2-means instances.*

The algorithm is inspired by work in Blum and Dunagan [2002] showing that the perceptron algorithm runs in poly-time with high probability in the smoothed analysis setting.

3.2.1. Review of perceptron algorithm

Suppose $y_1, \ldots, y_n$ is a sequence of labeled $\{+1, -1\}$-samples consistent with a linear threshold function, i.e., there exists vector w^* such that the labeling function $\ell(y_i)$ is consistent with $sgn(\langle y_i, w^* \rangle)$. At time $t = 0$, the perceptron algorithm sets $w_0 = 0$. At each subsequent time step, the algorithm sees sample y_t, outputs $sgn(\langle y_t, w_{t-1} \rangle)$ as its guess for $\ell(y_t)$, sees the true label $\ell(y_t)$, and updates w_t. On a correct guess, $w_t = w_{t-1}$, and on a mistake $w_t = w_{t-1} + \ell(y_t)y_t / \|y_t\|$.

The following well-known theorem Block [1962] bounds the number of total mistakes the perceptron algorithm can make in terms of the sequence's angular margin.

Theorem 3.2.2. *The number of mistakes made by the perceptron algorithm is bounded above by $(1/\gamma)^2$ for*

$$\gamma = \min_{i \in [n]} \frac{|\langle y_i, w^* \rangle|}{\|y_i\| \, \|w^*\|}.$$

For a universe U of elements and a function $f : U \to \mathbb{Z}_{\geq 0}$, we will denote by (U, f) the multiset where $u \in U$ appears in the multiset $f(u)$-many times. The size of a multiset is $\sum_{u \in U} f(u)$. The next lemma is an immediate consequence of the above theorem.

Lemma 3.2.3. *There exists a multiset $M = (\{ y_1, \ldots, y_n \}, f)$ of size at most $(1/\gamma)^2$ such that $\sum_{y \in M} \ell(y) \frac{y}{\|y\|}$ correctly classifies all of $\{ y_1, \ldots, y_n \}$.*

Proof. Let $r = (1/\gamma)^2 + 1$. Consider the performance of the perceptron algorithm on r consecutive runs of the $y_1, \ldots, y_n$, i.e., let the input be

$$\underbrace{\overbrace{y_1, \ldots, y_n}^{1 \text{ run}}, y_1, \ldots, y_n, \ldots, y_1, \ldots, y_n}_{r \text{ runs}}.$$

A mistake can only be made on a given run if mistakes were made on every previous run. Suppose the perceptron algorithm makes a mistake on the rth run, then the algorithm must have made at least $(1/\gamma)^2 + 1$ mistakes, a contradiction. Hence the direction of w after $r - 1$ runs correctly classifies all of $\{ y_1, \ldots, y_n \}$. The value of w is $\sum_{i \in [n]} f(y_i) \ell(y_i) \frac{y_i}{\|y_i\|}$ where $f(y_i)$ is the number of times y_i was misclassified. □

3.2.2. A perceptron-based clustering algorithm

Fix the following notation: let $X = \{ x_1, \ldots, x_n \} \subseteq \mathbb{R}^d$ be an ε-APS 2-means clustering instance with optimal clusters C_1, C_2 such that each cluster has at least 4 points. Let $D = \|\mu_1 - \mu_2\|$, $u = \frac{\mu_1 - \mu_2}{\|\mu_1 - \mu_2\|}$, $p = \frac{\mu_1 + \mu_2}{2}$. Without loss of generality, assume that $\sum_i x_i = 0$.

Lemma 3.1.5 gives a lower bound for γ in the correctly-centered set $\{ x_1 - p, \ldots, x_n - p \}$. Thus Lemma 3.2.3 might suggest a simple algorithm: for each multiset of bounded size and each of its possible labels, compute the cost of the associated clustering, then output the

clustering of minimum cost. However, a difficulty arises as the clusters C_1, C_2 may not be linearly separable (in particular the separating hyperplane may not pass through the origin). Note that the guarantees of the perceptron algorithm, and hence Lemma 3.2.3, do not hold in this case. Instead, we will apply the above idea to an instance Y, constructed from X, in which C_1, C_2 are linearly separable and we can efficiently lower bound γ.

Consider the following algorithm.

Algorithm 3.2.4.

Input: $X = \{ x_1, \ldots, x_n \}$, ε

1: If necessary, translate X such that $\sum x_i = 0$

2: **for all** pairs a, b of distinct points in $\{ x_i \}$ **do**

3: Let $\delta = \|a - b\|$

4: Let $Y_{a,b} = \{ y_1, \ldots, y_n \}$ be an instance given by $y_i = (x_i, \delta) \in \mathbb{R}^{d+1}$

5: **for all** multisets M of size at most $c_1^{-2}\varepsilon^{-8}$ and assignments $\ell : M \to \{ \pm 1 \}$ **do**

6: Let $w = \sum_{y \in M} \ell(y) \frac{y}{\|y\|}$

7: Calculate k-means cost of $C_1 = \{ x_i \mid \langle w, y_i \rangle \geq 0 \}, C_2 = \{ x_i \mid \langle w, y_i \rangle < 0 \}$.

8: Return clustering with smallest k-means objective found above

3.2.3. Overview of proof of Theorem 3.2.1

Each new instance $Y_{a,b}$ constructed in the algorithm has labeling consistent with some linear threshold function: $\ell(y_i) = \ell(x_i) = sgn(\langle x_i - p, u \rangle) = sgn(\langle x_i, u \rangle + \langle -p, u \rangle)$. Then taking $w^* = (u, \langle -p, u \rangle / \delta)$, we have that $\ell(y_i) = sgn(\langle y_i, w^* \rangle)$.

We will lower bound γ for a particular instance $Y_{a,b}$ in which a, b have nice properties. The following lemma states that on one of the iterations of its outer for loop, Algorithm 3.2.4 will pick such points.

Lemma 3.2.5. *There exist points $a \in C_1$, $b \in C_2$ such that $\langle a - p, u\rangle \leq \Delta/2$ and $\langle b - p, -u\rangle \leq \Delta/2$.*

The geometric conditions implied by ε-APS allow us to bound $\delta = \|a - b\|$ in terms of ε, D. In particular, using this handle on δ, it is possible to prove the following lower bound on γ.

Lemma 3.2.6. *There exists constant c_1 such that for any a, b satisfying Lemma 3.2.5, the corresponding instance $Y_{a,b}$ has*

$$\gamma = \min_{i \in [n]} \frac{|\langle y_i, w^* \rangle|}{\|y_i\| \, \|w^*\|} \geq c_1 \varepsilon^4.$$

The correctness of Algorithm 3.2.4 for all ε-APS 2-means clustering instances in which each cluster has at least 4 points then follows from Lemmas 3.2.3, 3.2.5, and 3.2.6. On the other hand, the optimal 2-means clustering where one of the clusters has at most 3 points can be calculated in $O(n^4 d)$ time. An algorithm that returns the better of these two solutions thus correctly clusters all ε-APS 2-means instances, completing the proof of Theorem 3.2.1.

3.2.4. Proof of Lemmas 3.2.5, 3.2.6

We state two lemmas that follow immediately from Lemma 3.1.5 and will be useful for the proofs in this section.

Lemma 3.2.7. *For any $x \in X$,*

$$\|\langle x - p, u\rangle\| \geq \varepsilon D.$$

In particular, for $x \in C_1$, $\langle x - p, u\rangle \geq \varepsilon D$ and for $x \in C_2$, $\langle x - p, u\rangle \leq -\varepsilon D$.

Lemma 3.2.8. *For any $x \in X$,*

$$\frac{|\langle x - p, u\rangle|}{\|x - p\|} \geq \sqrt{\frac{\varepsilon^2}{1 + \varepsilon^2}}.$$

Lemma 3.2.5. We restate and prove Lemma 3.2.5 below.

Lemma. *There exist points $a \in C_1$, $b \in C_2$ such that $\langle a - p, u\rangle \leq \Delta/2$ and $\langle b - p, -u\rangle \leq \Delta/2$.*

Proof of Lemma 3.2.5. Note that $\langle \mu_1 - p, u\rangle = \frac{1}{|C_1|}\sum_{x \in C_1}\langle x - p, u\rangle$. As $\langle \mu_1 - p, u\rangle = \Delta/2$, there must be some $a \in C_1$ such that $\{\, a - p, u \,\} \leq \Delta/2$. The second assertion is proved similarly. □

Lemma 3.2.6. Note that Lemmas 3.2.7 and 3.2.5 together imply that we *cannot* have an instance with $\varepsilon > 1/2$.

Lemma 3.2.9. *There is no ε-APS k-means clustering instance for $\varepsilon > 1/2$.*

The following lemma bounds $\delta = \|a - b\|$ in terms of ε, D.

Lemma 3.2.10. *Let $a, b \in X$ be points satisfying Lemma 3.2.5. Then,*

$$(2\varepsilon)D \leq \|a - b\| \leq \left(\sqrt{\frac{1 + \varepsilon^2}{\varepsilon^2}}\right) D.$$

Proof. For the first inequality, $\|a-b\| \geq |\langle u, a-b\rangle| = |\langle u, a-p\rangle - \langle u, b-p\rangle|$. Then by Lemma 3.2.7, $\|a-b\| \geq 2\varepsilon D$.

For the second inequality, $\|a-b\| \leq \|a-p\| + \|p-b\|$. By assumption, $\langle a-p, u\rangle \leq \Delta/2$. Then by Lemma 3.2.8, $\|a-p\| \leq \sqrt{(1+\varepsilon^2)/\varepsilon^2}D/2$. Similarly, $\|b-p\| \leq \sqrt{(1+\varepsilon^2)/\varepsilon^2}D/2$.

□

Finally, we restate and prove Lemma 3.2.6 below.

Lemma. *There exists constant c_1 such that for any a, b satisfying Lemma 3.2.8, the corresponding instance $Y_{a,b}$ has*

$$\gamma = \min_{i\in[n]} \frac{|\langle y_i, w^*\rangle|}{\|y_i\| \|w^*\|} \geq c_1\varepsilon^4.$$

Proof. We bound each term in the minimization individually. Let $i \in [n]$, then

$$\frac{|\langle y_i, w^*\rangle|}{\|y_i\| \|w^*\|} = \frac{|\langle x_i - p, u\rangle|}{\sqrt{\|x_i\|^2 + \delta^2}\sqrt{1 + \left(\frac{\langle p,u\rangle}{\delta}\right)^2}}.$$

We first observe the following facts.

- From Lemma 3.2.8, $|\langle x_i - p, u\rangle| \geq \sqrt{\frac{\varepsilon^2}{1+\varepsilon^2}} \|x_i - p\| \geq \frac{\varepsilon}{1+\varepsilon} \|x_i - p\|$
- By Lemma 3.2.8, $\|x_i\|^2 \leq 2\|x_i - p\|^2 + 2\|p\|^2 \leq 2\|x_i - p\|^2 + \frac{1}{2}\frac{1+\varepsilon^2}{\varepsilon^2}D^2$
- From Lemma 3.2.10, $\delta^2 \leq \frac{1+\varepsilon^2}{\varepsilon^2}D^2$
- As p and the origin both lie on the line between μ_1 and μ_2, $|\langle p, u\rangle| \leq \frac{D}{2} \leq \frac{\delta}{4\varepsilon}$
- From Lemma 3.2.7, $\|x_i - p\| \geq \varepsilon D$

Making each of the substitutions above,

$$\begin{aligned}\frac{|\langle y_i, w^* \rangle|}{\|y_i\| \|w^*\|} &\geq \varepsilon \frac{\|x_i - p\|}{(1+\varepsilon)\sqrt{2\|x_i - p\|^2 + \frac{3}{2}\frac{1+\varepsilon^2}{\varepsilon^2}D^2}\sqrt{1+\frac{1}{16\varepsilon^2}}} \\ &\geq \varepsilon \frac{1}{(1+\varepsilon)\sqrt{2 + \frac{3}{2}\frac{1+\varepsilon^2}{\varepsilon^2}\left(\frac{D}{\|x_i - p\|}\right)^2}\sqrt{1+\frac{1}{16\varepsilon^2}}} \\ &\geq \varepsilon \frac{1}{(1+\varepsilon)\sqrt{2 + \frac{3}{2\varepsilon^2} + \frac{3}{2\varepsilon^4}}\sqrt{1+\frac{1}{16\varepsilon^2}}}.\end{aligned}$$

Then, completing both squares,

$$\begin{aligned}\frac{|\langle y_i, w^* \rangle|}{\|y_i\| \|w^*\|} &\geq \varepsilon \frac{1}{(1+\varepsilon)\left(\sqrt{2} + \frac{\sqrt{3/2}}{\varepsilon^2}\right)\left(1 + \frac{1/4}{\varepsilon}\right)} \\ &= \varepsilon^4 \frac{1}{(1+\varepsilon)\left(\sqrt{2}\varepsilon^2 + \sqrt{3/2}\right)(\varepsilon + 1/4)}\end{aligned}$$

As $\varepsilon \leq 1/2$ by Lemma 3.2.9, we can bound the fraction below by some constant $c_1 \approx 0.563$. □

3.3. k-means clustering for general k

For general k, we will require the stronger $(\rho, \Delta, \varepsilon)$-separation. Consider the following algorithm.

Algorithm 3.3.1.

Input: $X = \{ x_1, \ldots, x_n \}$, k.

1: **for all** pairs a, b of distinct points in $\{ x_i \}$ **do**

2: Let $r = \|a - b\|$ be our guess for ρ

3: **procedure** `INITIALIZE`

4: Create graph G on vertices $\{ x_1, \ldots, x_n \}$ where x_i and x_j have an edge iff $\|x_i - x_j\| < r$

5: Let $a_1, \ldots, a_k \in \mathbb{R}^d$ where a_i is the mean of the ith largest connected component of G

6: **procedure** `ASSIGN`

7: Let $C_1, \ldots, C_k$ be the clusters obtained by assigning each point in X to the closest a_i

8: Calculate the k-means objective of $C_1, \ldots, C_k$

9: Return clustering with smallest k-means objective found above

---

Theorem 3.3.2. *Algorithm 3.3.1 recovers $C_1, \ldots, C_k$ for any $(\rho, \Delta, \varepsilon)$-separated instance with $\rho = \Omega\left(\frac{\Delta}{\varepsilon^2} + \frac{\beta\Delta}{\varepsilon}\right)$ and can be implemented in $\widetilde{O}(n^2kd)$ time.*

This running time can be achieved by inserting edges into a dynamic graph in order, maintaining connected components and their means using a union-find data structure, and noting that the number of connected components can change at most n times.

In particular, note that this algorithm does not need any prior knowledge of the stability parameters and its running time has no dependence on ρ, Δ, or ε.

Define the following regions of $\mathbb{R}^d$ for every pair i, j. Given i, j, let C_i, C_j be the corresponding clusters with means μ_i, μ_j. Let $u = \frac{\mu_i - \mu_j}{\|\mu_i - \mu_j\|}$ be the unit vector in the inter-mean direction.

Definition 3.3.3.

- $S_{i,j}^{(\text{cone})} = \{\, x \in \mathbb{R}^d \mid \|(x - (\mu_i - \Delta u))_{(V)}\| \leq \frac{1}{\varepsilon}\langle x - (\mu_i - \Delta u), u\rangle \,\}$,
- $S_{i,j}^{(\text{nice})} = \{\, x \in S_{i,j}^{(\text{cone})} \mid \langle x - \mu_i, u\rangle \leq 0 \,\}$,
- $S_i^{(\text{good})} = \bigcap_{j \neq i} S_{i,j}^{(\text{nice})}$.

See Figure 3.1b. for an illustration.

It suffices to prove the following two lemmas. Lemma 3.3.4 states that the initialization returned by the `INITIALIZE` subroutine satisfies certain properties when we guess $r = \rho$ correctly. As ρ is only used as a threshold on edge lengths, testing the distances between all pairs of data points i.e. $\{\, \|a - b\| : a, b \in X \,\}$ suffices. Lemma 3.3.5 states that the `ASSIGN` subroutine correctly clusters all points given an initialization satisfying these properties.

Lemma 3.3.4. *For a $(\rho, \Delta, \varepsilon)$-separated instance with balance parameter β and $\rho = \Omega(\beta\Delta/\varepsilon)$, the `INITIALIZE` subroutine finds a set $\{\, a_1, \ldots, a_k \,\}$ where $a_i \in S_i^{(good)}$ when $r = \rho$.*

Lemma 3.3.5. *For a $(\rho, \Delta, \varepsilon)$-separated instance with $\rho = \Omega(\Delta/\varepsilon^2)$, the `ASSIGN` subroutine recovers $C_1, C_2, \cdots C_k$ correctly when initialized with k points $\{\, a_1, a_2, \ldots, a_k \,\}$ where $a_i \in S_i^{(good)}$.*

3.3.1. Proof of Lemma 3.3.4.

Suppose $r = \rho$ and consider the graph constructed by Algorithm 3.3.1. We start by defining the *core region* of each cluster.

Definition 3.3.6 ($S^{(\text{core})}$)**.** Let $S_i^{(\text{core})} = \{\, x \in \mathbb{R}^d \mid \|x - \mu_i\| \leq \Delta/\varepsilon \,\}$.

The core regions are defined in such a way that for each cluster C_i, all points in $C_i \cap S_i^{(\text{core})}$ belong to a single connected component. Although $S_i^{(\text{core})}$ may not contain too many points on its own, the connected component containing $S_i^{(\text{core})}$ will contain most (at least $\beta/(1+\beta)$ fraction) of the points in C_i. Hence, the k largest components will be the connected components containing the k different core regions. Finally, since the connected component containing $S_i^{(\text{core})}$ contains most of the points in C_i, the geometric conditions of $(\rho, \Delta, \varepsilon)$-separation ensure that the empirical mean of the connected component lies in $S_i^{(\text{good})}$. The following lemma states some properties of the connected components in our graph.

Lemma 3.3.7.

(1) Any connected component only contains points from a single cluster.

(2) For all i, j, $S_i^{(core)} \supseteq S_{i,j}^{(nice)}$. There is a point $x \in C_i$ such that $x \in S_i^{(core)} \cap S_{i,j}^{(nice)}$.

(3) For all i, j, let $A_{i,j} = \{\, x \in C_i \mid \langle x - \mu_i, u \rangle \leq \beta\Delta \,\}$. Then, $|A_{i,j}| \geq \frac{\beta}{1+\beta}|C_i|$.

(4) For all i, $S_i^{(core)} \cap X$ is connected in G.

(5) For all i, j, $A_{i,j}$ is connected in G.

(6) The largest component, K_i, in each cluster contains $A_{i,j}$ for each $j \neq i$. In particular, $|K_i| \geq \frac{\beta}{1+\beta}|C_i|$, and K_i contains $S_i^{(core)} \cap X$.

Proof.

(1) Let $x \in C_i$ and $y \in C_j$. Then $\|x - y\| \geq |\langle x - y, u \rangle| \geq \rho$, thus no edge connecting points in different clusters is added to G.

(2) For $x \in S_{i,j}^{(\text{nice})}$, $\|(x - \mu_i)_{(V)}\| \leq \frac{1}{\varepsilon}(\Delta - \|(x - \mu_i)_{(u)}\|)$, hence $\|x - \mu_i\| \leq \Delta/\varepsilon$. Recall μ_i is the mean of the points in cluster C_i. By an averaging argument, $S_{i,j}^{(\text{nice})} \cap$

$X = \{\, x \in C_i \mid \langle x - (\mu_i - \Delta u), u\rangle \le \Delta \,\}$ is nonempty and hence $S_i^{(\text{core})} \cap S_{i,j}^{(\text{nice})}$ is nonempty.

(3) μ_i is the mean of the points in cluster C_i. By an averaging argument, $|A_{i,j}|\Delta - (|C_i| - |A_{i,j}|)\beta\Delta \ge 0$. Rearranging, $|A_{i,j}| \ge \frac{\beta}{1+\beta}|C_i|$.

(4) For $x, y \in S_i^{(\text{core})}$, $\|x - y\| \le 2\Delta/\varepsilon$. Thus for $\rho = \Omega(\Delta/\varepsilon)$, the points $S_i^{(\text{core})} \cap X$ are connected.

(5) From 2 above, $S_{i,j}^{(\text{nice})} \cap X$ is nonempty; fix such a point x. For $y \in A_{i,j}$, $\|x - y\|^2 = \|(x-y)_{(u)}\|^2 + \|(x-y)_{(V)}\|^2 \le ((\beta+1)\Delta)^2 + ((\beta+1)\Delta/\varepsilon)^2$. Thus for $\rho = \Omega(\beta\Delta/\varepsilon)$, all of $A_{i,j}$ is connected through x.

(6) Let K_i be the component containing $S_i^{(\text{core})} \cap X$. By 2 above, for all j there exists a point $x_{(j)} \in S_i^{(\text{core})}$ such that $x_{(j)} \in S_{i,j}^{(\text{nice})} \subseteq A_{i,j}$. Then as $A_{i,j}$ is connected, K_i must also contain $A_{i,j}$. As $|K_i| \ge |A|$ and $\beta \ge 1$, part 3 above tells us that K_i is the largest connected component in C_i.

□

Lemma 3.3.8 states that the k largest components (and hence $\{\, a_1, \ldots, a_k \,\}$) must belong to different clusters while Lemma 3.3.9 states that each a_i lie inside a good region. Together, they imply Lemma 3.3.4, i.e. each a_i comes from a different good region.

Lemma 3.3.8. *The set of k largest components of G contains the largest component of each cluster.*

Proof. Let K_i be the largest component in C_i and let K'_j be a component in C_j that is not the largest. Then by the β parameter, $|K_i| \ge \frac{\beta}{1+\beta}|C_i| > \frac{1}{1+\beta}|C_j| \ge |K'_j|$. It follows that the k largest connected components are $K_1, K_2, \ldots, K_k$. □

Lemma 3.3.9. *The mean of points in K_i lies in $S_i^{(good)}$.*

Proof. Let a_i be the mean of the points in K_i. As $K_i \subseteq S_{i,j}^{(\mathrm{cone})}$ is a convex set, $a_i \in S_{i,j}^{(\mathrm{cone})}$. As $K_i \supseteq S_i^{(\mathrm{core})} \cap X \supseteq S_{i,j}^{(\mathrm{nice})} \cap X$, the points $x \in C_i$ not contained in K_i have $\langle x - \mu_i, u\rangle > 0$. Noting that $\sum_{x \in C_i} \langle x - \mu_i, u\rangle = 0$, it follows that $\langle a_i - \mu_i\rangle \leq 0$. Hence, $a_i \in S_{i,j}^{(\mathrm{nice})}$. As this holds for each $j \neq i$, $a_i \in S_i^{(\mathrm{good})}$. □

3.3.2. Proof of Lemma 3.3.5.

We will show that for any $a_i \in S_{i,j}^{(\mathrm{nice})}$, $a_j \in S_{j,i}^{(\mathrm{nice})}$, and $x \in C_i$, x is closer to a_i than to a_j. The following lemma states some properties of the perpendicular bisector between a_i and a_j. These statements follow from the definitions of the nice regions and the angular separation.

Lemma 3.3.10. *Suppose $\rho = \Omega(\Delta/\varepsilon^2)$. Then, for $a_i \in S_{i,j}^{(nice)}$ and $a_j \in S_{j,i}^{(nice)}$, we have*

(1) $\|(a_i - a_j)_{(u)}\| \geq \frac{\|(a_i - a_j)_{(V)}\|}{\varepsilon}$,

(2) $\langle \frac{a_i + a_j}{2} - p, u\rangle \leq \frac{\Delta}{2}$, *and*

(3) $\left\| \left(\frac{a_i + a_j}{2} - p\right)_{(V)} \right\| \leq \Delta/\varepsilon$.

Proof.

(1) We have $\|(a_i - a_j)_{(V)}\| \leq 2\Delta/\varepsilon$. On the other hand, $\rho \leq \|(a_i - a_j)_{(u)}\|$. Thus the inequality holds for $\rho \geq 2\Delta/\varepsilon^2$.

(2) $\langle a_i + a_j - 2p, u\rangle = \langle a_i - p, u\rangle + \langle a_j - p, u\rangle \leq D_{i,j}/2 + (-D_{i,j}/2 + \Delta) = \Delta$. Multiplying by $1/2$ gives the desired inequality.

(3) $\|(a_i + a_j - 2p)_{(V)}\| \leq \|(a_i - p)_{(V)}\| + \|(a_j - p)_{(V)}\| \leq 2\Delta/\varepsilon$. Multiplying by $1/2$ gives the desired inequality.

□

To prove Lemma 3.3.5, we rewrite the condition $\|x - a_i\| \leq \|x - a_j\|$ as $\langle x - p - (\frac{1}{2}(a_i + a_j) - p), a_i - a_j \rangle \geq 0$. Then we write each vector in terms of their projection on u and V and use the above lemma to bound each of the terms.

Proof of Lemma 3.3.5. It suffices to show that for any $a_i \in S_{i,j}^{(\text{nice})}$, $a_j \in S_{j,i}^{(\text{nice})}$, and $x \in C_i$, $\|x - a_i\| \leq \|x - a_j\|$. Then by Lemma 3.3.10 above,

$$\begin{aligned}
\left\langle (x-p) - \left(\frac{a_i + a_j}{2} - p\right), a_i - a_j \right\rangle &= \left\langle (x-p)_{(u)}, (a_i - a_j)_{(u)} \right\rangle + \left\langle (x-p)_{(V)}, (a_i - a_j)_{(V)} \right\rangle \\
&\quad - \left\langle (\tfrac{1}{2}(a_i + a_j) - p)_{(u)}, (a_i - a_j)_{(u)} \right\rangle \\
&\quad - \left\langle (\tfrac{1}{2}(a_i + a_j) - p)_{(V)}, (a_i - a_j)_{(V)} \right\rangle \\
&\geq \|(x-p)_{(u)}\| \|(a_i - a_j)_{(u)}\| - \frac{1}{\varepsilon}\left(\|(x-p)_{(u)}\| - \rho/2\right)\varepsilon \|(a_i - a_j)_{(u)}\| \\
&\quad - \frac{\Delta}{2}\|(a_i - a_j)_{(u)}\| - \frac{\Delta}{\varepsilon}\varepsilon\|(a_i - a_j)_{(u)}\| \\
&= \left(\frac{\rho}{2} - \frac{3}{2}\Delta\right)\|(a_i - a_j)_{(u)}\| \geq 0
\end{aligned}$$

where the first inequality follows because of equality on the first term and Cauchy-Schwarz on the rest. So, for all $a_i \in S_{i,j}^{(\text{nice})}$, $a_j \in S_{j,i}^{(\text{nice})}$, and $x \in C_i$, x is closer to a_i than a_j. □

3.4. Robust k-means

A simple extension of algorithm 3.3.1 does well even in the presence of adversarial noise for instances with $(\rho, \Delta, \varepsilon)$-separation for large enough ρ. Specifically, we consider the following model.

Let $X = \{ x_1, \ldots, x_n \} \subset \mathbb{R}^d$ be a k-means clustering instance with optimal clustering $C_1, \ldots, C_k$. We call X the set of *pure* points. An additional set of at most ηn-many *impure* points $Z \subset \mathbb{R}^d$ is added by an adversary. Our goal is to find a clustering of $X \cup Z$ that agrees with $C_1, \ldots, C_k$ on the pure points.

Let $w_{\max} = \max|C_i|/n$ and let $w_{\min} = \min|C_i|/n$ be the maximum and minimum weight of clusters. We will assume that $\eta < w_{\min}$.

Algorithm 3.4.1.

Input: $X \cup Z$, r, t

1: **procedure** `INITIALIZE`

2: Create graph G on $X \cup Z$ where vertices u and v have an edge iff $\|u - v\| < r$

3: Remove vertices with vertex degree $< t$

4: Let $a_1, \ldots, a_k \in \mathbb{R}^d$ where a_i is the mean of the ith largest connected component of G

5: **procedure** `ASSIGN`

6: Let $C_1, \ldots, C_k$ be the clusters obtained by assigning each point in $I \cup Z$ to the closest a_i

Theorem 3.4.2. *Given $X \cup Z$ where X satisfies $(\rho, \Delta, \varepsilon)$-separation for*

$$\rho = \Omega\left(\frac{\Delta}{\varepsilon^2}\left(\frac{w_{\max} + \eta}{w_{\min} - \eta}\right)\right),$$

$|X| = n$ and $|Z| \leq \eta n$ for $\eta < w_{\min}$, there exists values of r, t such that Algorithm 3.4.1 returns a clustering consistent with $C_1, \ldots, C_k$ on X. Algorithm 3.4.1 can be implemented in $\widetilde{O}(n^2 kd)$ time.

Proof. Fix the following parameters.

$$\alpha = 2\left(\frac{w_{\max} + \eta}{w_{\min} - \eta}\right), \qquad r = \Delta(\alpha + 1)(1 + 2/\varepsilon), \qquad t = w_{\min} n \frac{\alpha}{\alpha + 1}.$$

Define the following extended and robust versions of the regions defined in Section 3.3. Given i, j, let C_i, C_j be the corresponding clusters with means μ_i, μ_j. Let $u = \frac{\mu_i - \mu_j}{\|\mu_i - \mu_j\|}$ be the unit vector in the inter-mean direction.

Definition 3.4.3.

- $S_{i,j}^{(\text{e nice})} = \{\, x \in S_{i,j}^{(\text{cone})} \mid \langle x - \mu_i, u \rangle \leq \alpha\Delta \,\}$,
- $S_{i,j}^{(\text{r e nice})} = \{\, x \in \mathbb{R}^d \mid d(x, S_{i,j}^{(\text{e nice})}) \leq r \,\}$,
- $S_i^{(\text{r good})} = \bigcap_{j \neq i} S_{i,j}^{(\text{r e nice})}$.

Again, it suffices to prove the following two lemmas. Lemma 3.4.4 states that the initialization returned by the `INITIALIZE` subroutine satisfies certain properties when given r, t. As in the case of Algorithm 3.3.1, this algorithm uses r and t as thresholds. Hence, it is possible to guess r from the $\binom{n}{2}$ pairwise edge lengths and t from $[n]$ if necessary. Lemma 3.4.5 states that the `ASSIGN` subroutine correctly clusters all points given an initialization satisfying these properties.

Lemma 3.4.4. *Given $X \cup Z$ where X is a $(\rho, \Delta, \varepsilon)$-separated instance with $\rho = \Omega(\alpha\Delta/\varepsilon^2)$ and $\eta < w_{\min}$, for the choices of r and t as above, the* `INITIALIZE` *subroutine finds a set $\{ a_1, \ldots, a_k \}$ where $a_i \in S_i^{(r\ good)}$*

Lemma 3.4.5. *Given $X \cup Z$ where X is a $(\rho, \Delta, \varepsilon)$-separated instance with $\rho = \Omega(\alpha\Delta/\varepsilon^2)$ and $\eta < w_{\min}$, the* `ASSIGN` *subroutine finds a clustering consistent with $C_1, \ldots, C_k$ on X when initialized with k points $\{ a_1, \ldots, a_k \}$ where $a_i \in S_i^{(r\ good)}$.*

3.4.1. Proof of Lemma 3.4.4

Consider the graph constructed by Algorithm 3.4.1. The following lemma states some properties of the connected components in our graph.

Lemma 3.4.6.

(1) For any $i \neq j$, the set of vertices $S_{i,j}^{(e\ nice)} \cap X$ forms a clique and the size of this clique is greater than t. In particular, no vertex in $S_{i,j}^{(e\ nice)}$ is deleted.

(2) Fix i. For all $j \neq i$, the vertices $S_{i,j}^{(e\ nice)} \cap X$ belong to a single connected component. Let K_i be this connected component.

(3) Before vertex deletion (and after), no vertex is adjacent to pure points from different clusters.

(4) After vertex deletion, every remaining point lies in $S_i^{(r\ good)}$ for some i. Hence by part 2, every connected component contains pure points from at most a single cluster. In particular, $K_1, \ldots, K_k$ are distinct.

Proof.

(1) The diameter of $S_{i,j}^{(\text{e nice})}$ is $L(S_{i,j}^{(\text{e nice})}) \leq (\alpha+1)2\Delta/\varepsilon < r$. Thus every pair of points in this region is connected. Recall that μ_i is the mean of the pure points in cluster C_i. By an averaging argument, $|S_{i,j}^{(\text{e nice})} \cap X|\Delta - (|C_i| - |S_{i,j}^{(\text{e nice})} \cap X|)\alpha\Delta \geq 0$. Rearranging, $|S_{i,j}^{(\text{e nice})} \cap X| \geq \frac{\alpha}{\alpha+1}|C_i| \geq \frac{\alpha}{\alpha+1} n w_{\min} = t$.

(2) Fix i. Let $j \neq i$. Recall $S_{i,j}^{(\text{nice})} \cap X$ is nonempty; let $x \in S_{i,j}^{(\text{nice})} \cap X$. Then $\|x - \mu_i\| \leq \Delta/\varepsilon$. We show that for any $j' \neq i$, the connected component containing x contains $S_{i,j'}^{(\text{e nice})} \cap X$. Let $y \in S_{i,j'}^{(\text{e nice})} \cap X$. Then $\|y - x\| \leq \|y - \mu_i\| + \|x - \mu_i\| \leq (\alpha+1)\Delta/\varepsilon + \alpha\Delta + \Delta/\varepsilon < \Delta(\alpha+1)(1+2/\varepsilon) = r$.

(3) Pure points in different clusters are at distance at least ρ whereas two vertices sharing a neighbor must be at distance less than $2r$. Thus the inequality holds for $\rho \geq \Omega(\alpha\Delta/\varepsilon)$.

(4) Let x be a point not in $\bigcup_i S_i^{(\text{r good})}$. By part 3 above, x can only be connected to pure points in a single cluster. Suppose it is connected to pure points in cluster C_i. By assumption, there exists a j such that $x \notin S_{i,j}^{(\text{r e nice})}$. We bound the degree of x above by the number of points in $X \setminus S_{i,j}^{(\text{e nice})}$ and the ηn-many impure points, i.e., $\deg(x) \leq \eta n + \frac{|C_i|}{\alpha+1} \leq n(\eta + \frac{w_{\max}}{\alpha+1})$. By our choice of t, we have that $\deg(x) < t$. Thus x is deleted and all remaining points lie in $\bigcup_i S_i^{(\text{r good})}$.

For any i, j, the minimum distance between $S_i^{(\text{r good})}$ and $S_j^{(\text{r good})}$ is at least $\rho - 2r$. For some $\rho \geq \Omega(\alpha\Delta/\varepsilon)$ then, the distance between these regions is greater than $\rho - 2r > r$ and no connected component contains pure points from multiple clusters.

□

Lemma 3.4.7 state that the k largest components contain pure points corresponding to different clusters while Lemma 3.4.8 states that each a_i lies inside a robust good region. Together, they imply Lemma 3.4.4, i.e. each a_i lies in a different robust good region.

Lemma 3.4.7. *Let K_i be defined as above. For any arbitrary connected component K not in $K_1, \ldots, K_k$, $|K_i| > |K|$. In particular, the k largest components of G are $K_1, \ldots, K_k$.*

Proof. As in part 2 above, the size of K_i is bounded below by the averaging argument $|K_i| \geq \frac{\alpha}{\alpha+1}|C_i|$. By part 3 above, K contains pure points from at most a single cluster C_j. By part 5 above, the size of the connected component K is bounded above by the number of remaining points after K_j is removed and the ηn-many impure points, i.e., $|C_j| \leq \frac{1}{\alpha+1}|C_j| + \eta n$. Then by our choice of α, $|K| < |K_i|$. □

Lemma 3.4.8. *The mean of K_i lies in $S_i^{(r\ good)}$.*

Proof. By above, $K_i \subseteq S_i^{(\text{r good})}$. As $S_i^{(\text{r good})}$ is convex, the mean of K_i also lies in $S_i^{(\text{r good})}$. □

3.4.2. Proof of Lemma 3.4.5

We will show that for any $a_i \in S_{i,j}^{(\text{r e nice})}$, $a_j \in S_{j,i}^{(\text{r e nice})}$ and $x \in C_i$, x is closer to a_i than a_j. The following lemma states some properties of the perpendicular bisector between a_i and a_j.

Lemma 3.4.9. *Suppose $\rho = \Omega(\alpha\Delta/\varepsilon^2)$. Then, for $a_i \in S_{i,j}^{(r\ e\ nice)}$ and $a_j \in S_{j,i}^{(r\ e\ nice)}$, we have*

(1) $\|(a_i - a_j)_{(u)}\| \geq \frac{\|(a_i - a_j)_{(V)}\|}{\varepsilon}$,

(2) $\langle \frac{a_i+a_j}{2} - p, u\rangle \leq (\alpha+1)\Delta/2 + r$,

(3) $\left\| \left(\frac{a_i+a_j}{2} - p\right)_{(V)} \right\| \leq (\alpha+1)\Delta/\varepsilon + r$.

Proof.

(1) By triangle inequality, $\|(a_i - a_j)_{(V)}\| \leq 2((\alpha+1)\Delta/\varepsilon + r)$. On the other hand, $\|(a_i - a_j)_{(u)}\| \geq \rho - 2r$. Thus the inequality holds for $\rho \geq 2r + \frac{2}{\varepsilon}((\alpha+1)\Delta/\varepsilon + r)$.

(2) $\langle a_i + a_j - 2p, u\rangle = \langle a_i - p, u\rangle + \langle a_j - p, u\rangle \leq (D_{i,j}/2 + \alpha\Delta + r) + (-D_{i,j}/2 + \Delta + r) = (\alpha+1)\Delta + 2r$. Multiplying by $1/2$ gives the desired inequality.

(3) $\|(a_i + a_j - 2p)_{(V)}\| \leq \|(a_i - p)_{(V)}\| + \|(a_j - p)_{(V)}\| \leq 2((\alpha+1)\Delta/\varepsilon + r)$. Multiplying by $1/2$ gives the desired inequality.

□

To prove Lemma 3.4.5, we rewrite the condition $\|x - a_i\| \leq \|x - a_j\|$ as $\langle (x-p) - (\frac{1}{2}(a_i + a_j) - p), a_i - a_j\rangle \geq 0$. Then we write each vector in terms of their projection on u and V and use the above lemma to bound each of the terms.

PROOF OF LEMMA 3.4.5. It suffices to show that for any $a_i \in S_{i,j}^{(\text{r e nice})}$, $a_j \in S_{j,i}^{(\text{r e nice})}$ and $x \in C_i$, $\|x - a_i\| \leq \|x - a_j\|$. Then by Lemma 3.4.9 above,

$$\begin{aligned}
\left\langle (x-p) - \left(\frac{a_i + a_j}{2} - p\right), a_i - a_j \right\rangle &= \left\langle (x-p)_{(u)}, (a_i - a_j)_{(u)} \right\rangle + \left\langle (x-p)_{(V)}, (a_i - a_j)_{(V)} \right\rangle \\
&\quad - \tfrac{1}{2} \left\langle (a_i + a_j - 2p)_{(u)}, (a_i - a_j)_{(u)} \right\rangle \\
&\quad - \tfrac{1}{2} \left\langle (a_i + a_j - 2p)_{(V)}, (a_i - a_j)_{(V)} \right\rangle \\
&\geq \|(x-p)_{(u)}\| \|(a_i - a_j)_{(u)}\| - \frac{1}{\varepsilon} \left(\|(x-p)_{(u)}\| - \rho/2 \right) \varepsilon \|(a_i - a_j)_{(u)}\| \\
&\quad - \left((\alpha+1)\Delta/2 + r \right) \|(a_i - a_j)_{(u)}\| \\
&\quad - \left((\alpha+1)\Delta/\varepsilon + r \right) \varepsilon \|(a_i - a_j)_{(u)}\| \\
&= \left(\frac{\rho}{2} - \left(\frac{3}{2}(\alpha+1)\Delta + (1+\varepsilon)r \right) \right) \|(a_i - a_j)_{(u)}\|
\end{aligned}$$

where the inequality follows because of equality on the first term and Cauchy-Schwarz on the rest. So, when $\rho = \Omega(\alpha\Delta/\varepsilon^2)$, for all $a_i \in S_{i,j}^{(\text{r e nice})}$, $a_j \in S_{j,i}^{(\text{r e nice})}$, and $x \in C_i$, x is closer to a_i than a_j. □

□

CHAPTER 4

Adversarial Learning : Upper Bound

In this chapter, we introduce a broad class of polynomial optimization problems and show a connection between them and designing adversarial examples for depth 2 neural networks with ReLU gates.

4.1. Finding Adversarial Examples Using Polynomial Optimization

The following proposition shows the crucial connection between finding adversarial examples and polynomial optimization. It proves the existence of an algorithm that finds adversarial examples if given access to an algorithm that optimizes polynomials. While our theory is written in terms of deterministic binary classification, it extends fairly easily to multiclass classification and randomized algorithms.

Proposition 4.1.1. *Let $\gamma \geq 1$. There is an efficient algorithm that given a classifier $sgn(f(x))$ and a point x^*, and budget $\delta > 0$, guarantees to either (a) find an adversarial example in $B^n_\infty(x^*, \gamma\delta)$, or (b) certify the absence of any adversarial example in $B^n_\infty(x^*, \delta)$, given access to an efficient optimization algorithm that takes x^* and a polynomial $g(z) \in \{ f(x^* + z), -f(x^* + z) \}$ as input and finds a $\widehat{z}$ such that $g(\widehat{z}) \geq \max_{\|z\|_\infty \leq \delta} g(z)$ with $\|\widehat{z}\|_\infty \leq \gamma\delta$.*

Proof of Proposition 4.1.1. Let ALG_γ be the optimization algorithm. Suppose there exists an adversarial example $x^* + z^*$ with $\|z^*\|_\infty \leq \delta$, and let $y^* := sgn(f(x^*))$ be the

label for the point x^*. Then we have that $\max_{z:\|z\|_\infty \le \delta}(-y^*)f(x^*+z) \ge (-y^*)f(x^*+z^*) > 0$. Now for $g(z) = -y^* f(x^*+z)$ (a polynomial in z), we get that ALG_γ finds a point $\widehat{z}$ with $\|\widehat{z}\|_\infty \le \gamma\delta$ that also satisfies $(-y^*)f(x^*+\widehat{z}) > 0$ i.e., $sgn(f(x^*)) \ne sgn(f(x^*+\widehat{z}))$, as required. Furthermore, if ALG_γ fails, i.e., outputs a $\hat{z}$ such that $(-y^*)f(x^*+\widehat{z}) < 0$, then from the guarantee of the algorithm we know that $\max_{z:\|z\|_\infty \le \delta}(-y^*)f(x^*+z) < 0$ and hence no adversarial example exists within a δ ball around x^*. □

While the proof of the proposition only requires that the algorithm returns $\widehat{z}$ with $g(\widehat{z}) > 0$, it effectively requires that $\widehat{z}$ attains at least as large an objective value because the constant term can be arbitrary. When the classifier is a degree-d PTF of the form $sgn(f)$, it leads to the following approximate optimization problem: given as input a degree d polynomial $g : \mathbb{R}^n \to \mathbb{R}$ (potentially different from f) and any $\eta, \delta > 0$, find in time $\mathrm{poly}(n, \log(\frac{1}{\eta}))$ and w.p. at least $1-\eta$ a point $\hat{x}$ s.t.

$$g(\hat{x}) \ge \max_{x \in B^n_\infty(0,\delta)} g(x) \text{ and } \widehat{x} \in B^n_\infty(0, \gamma\delta). \tag{4.1}$$

Given a n-variate polynomial g, consider the following basic polynomial optimization problem

$$val^* := \max_{x \in B^n_\infty(0,\delta)} g(x). \tag{4.2}$$

This simple version of polynomial optimization problem is NP-hard for polynomials of degree-2 or more (see Section 5.1 for example). We study a natural approximation variant of this problem, that asks, *given a polynomial $g(x)$ such that $\max_{x \in B^n_\infty(0,\delta)} g(x) = val^*$, can one output in polynomial time, a point $\hat{x}$ in a larger $\|\|_\infty$ ball such that $g(\hat{x}) \ge val^*$?* The above proposition proves that if there exists an algorithm that can solve this approximate version of

the maximization problem for a particular class of polynomials, then we can find adversarial examples in the relaxed ball or certify their absence for the corresponding class of PTFs.

As a warm-up, using this framework, we prove that we can find adversarial examples for linear separators.

Claim 4.1.2. *There is a deterministic linear-time algorithm that given any linear threshold function $sgn(b^T x + c)$, a point x^* and $\delta > 0$, provably finds an adversarial example in the ℓ_∞ ball of δ around x^* when it exists.*

Proof. We use Proposition 4.1.1 and give a corresponding polynomial maximization algorithm for linear functions. For linear function $g(x)$ represented by $g(x) := b^T x + c$ where $b \in \mathbb{R}^n, c \in \mathbb{R}$, we can easily find a solution $\widehat{x} \in B^n_\infty(0, \delta)$ such that $g(\widehat{x}) = \max_{x \in B^n_\infty(0,\delta)} g(x)$. This is because the linear form $b^T x + c$ is maximized within $B^n_\infty(0, \delta)$ by setting each variable x_i to be δ if the corresponding $b_i \geq 0$, and $-\delta$, otherwise. □

Maximizing a degree-1 polynomial is easy, so in the case of linear threshold functions we can exactly find adversarial examples when they exist. The following theorem is the main theoretical result that gives an algorithm to provably find adversarial examples of degree-2 PTFs.

Theorem 4.1.3. *For any $\delta, \eta > 0$, there is a polynomial time algorithm that given a degree-2 PTF $sgn(f(x))$ and an example $(x^*, sgn(f(x^*)))$, guarantees at least one of the following holds with probability at least $(1 - \eta)$: (a) finds an adversarial example $(x^* + \widehat{z})$ i.e., $sgn(f(x^*)) \neq sgn(f(x^* + \widehat{z}))$, with $\|\widehat{z}\|_\infty \leq C\delta\sqrt{\log n}$, or (b) certifies that $\forall z : \|z\|_\infty \leq \delta$, $sgn(f(x^*)) = sgn(f(x^* + z))$ for some constant $C > 0$.*

(1) Given (A, b, c) that defines the polynomial $g(z) := z^T Az + b^T z + c$.
(2) Solve the SDP given by following vector program:
max $\sum_{i,j} A_{ij}\langle u_i, u_j\rangle + \sum_i b_i\langle u_i, u_0\rangle + c$ subject to $\|u_i\|_2^2 \leq \delta^2\ \forall i \in [n]$, $\|u_0\|_2^2 = 1$.
(3) Let $u_i^\perp$ represent the component of u_i orthogonal to u_0. Draw $\zeta \sim N(0, I)$ a standard Gaussian vector, and set $\widehat{z}_i := \langle u_i, u_0\rangle + \langle u_i^\perp, \zeta\rangle$ for each $i \in \{\, 0, 1, \ldots, n \,\}$.
(4) Repeat rounding $O(\log(1/\eta))$ random choices of ζ and pick the best choice.

Figure 4.1. The SDP-based algorithm for the degree-2 optimization problem.

To establish the above theorem using Proposition 4.1.1, we need to design a polynomial time algorithm that given any degree-2 polynomial $g(x) = x^T Ax + b^T x + c$ with $A \in \mathbb{R}^{n\times n}, b \in \mathbb{R}^n, c \in \mathbb{R}$, finds a solution $\widehat{x}$ with $\|\widehat{x}\|_\infty \leq O(\sqrt{\log n}) \cdot \delta$ such that $g(\widehat{x}) \geq \max_{\|x\|_\infty \leq \delta} g(x)$.

We use a semi-definite programming (SDP) based algorithm shown in Figure 4.1, that is directly inspired by the SDP-based algorithm for quadratic programming (QP) by [Charikar and Wirth, 2004; Nesterov, 1998]. 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 ℓ_∞ length of the vector. Secondly, unlike QP where the diagonal terms are all 0, 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 $\{\, \pm 1 \,\}$. 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 approximation factor can be improved even for the special case of homogeenous degree-2

polynomials, based on the current state of the approximability of Quadratic Programming (see Remark 4.1.5 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).

$$\max_{\{ u_0, u_1, \ldots, u_n \}} \sum_{i,j=1}^{n} A_{ij} \langle u_i, u_j \rangle + \sum_{i=1}^{n} b_i \langle u_i, u_0 \rangle + c \tag{4.3}$$

$$\text{s.t. } \|u_i\|_2^2 \leq \delta^2 \ \forall i \in \{ 1, 2, \ldots, n \}, \text{ and } \|u_0\|_2^2 = 1. \tag{4.4}$$

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 $x \in [-\delta, \delta]^n$, there is a corresponding SDP solution with the same objective value, given by $\big(u_i = x_i u_0 : i \in [n]\big)$ for any unit vector u_0. Hence $\text{SDP}_{val} \geq \max_{\|x\|_\infty \leq \delta} g(x)$. Moreover, when the SDP value SDP_{val} is negative, then $\max_{\|x\|_\infty \leq \delta} g(x)$ is negative which means that the classifier is robust around the given sample x^*.

We prove Theorem 4.1.3 by designing a polynomial time rounding algorithm that takes the SDP solution and obtains a valid $\hat{z}$ satisfying the requirements of the theorem.

Gaussian 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 $\{ \widehat{x}_i : i \in [n] \}$:

$$\forall i \in \{ 0, 1, \ldots, n \}, \ \widehat{x}_i := \langle u_i, u_0 \rangle + \langle u_i, \zeta \rangle = \langle u_i, u_0 \rangle + \langle u_i^\perp, \zeta \rangle, \text{ with } \zeta \sim N\Big(0, \Pi^\perp\Big), \tag{4.5}$$

where $\Pi^{\perp}$ is the projection matrix onto the subspace of $span(\{ u_1^{\perp}, \ldots, u_n^{\perp} \})$. 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

$$\forall i \in \{ 0, 1, \ldots, n \}, \ \widehat{x}_i = \langle u_i, u_0 \rangle + \langle u_i^{\perp}, \zeta \rangle \text{ where } \langle \zeta, e_0 \rangle = 0, \ \langle \zeta, v \rangle \sim N(0, \|v\|_2^2) \text{ for every } v \perp e_0, \tag{4.6}$$

and $\widehat{x}_0 = 1$. The rounding algorithm just tries $O(\log(1/\eta))$ independent random draws for ζ, and picks the best of these solutions.

We now give the analysis of the algorithm. We prove Theorem 4.1.3 by showing the following guarantee for the rounding algorithm.

Lemma 4.1.4. *There is a polynomial time randomized rounding algorithm that takes as input the solution of the SDP as defined in Equations* (4.3)*, and 4.4, and outputs a solution* $\widehat{x}$ *given by Equation 4.6 such that*

$$\mathop{\mathbb{P}}_{\widehat{x}} \left[g(\widehat{x}) \geq \max_{\|x\|_\infty \leq \delta} g(x) \text{ and } \|\widehat{x}\|_\infty \leq O(\sqrt{\log n}) \cdot \delta \right] \geq \Omega(1). \tag{4.7}$$

Assuming (4.7), we can repeat the algorithm at least $O(\log(1/\eta))$ times to get the guarantee of Theorem 4.1.3.

Proof of Lemma 4.1.4. We start with a simple observation that follows from the standard properties of spherical Gaussians. For any $i, j \in [n]$, we have $\mathbb{E}_\zeta[\langle u_i^{\perp}, \zeta \rangle \langle u_j^{\perp}, \zeta \rangle] =$

$(u_i^\perp)^T \Pi^\perp u_j^\perp = \langle u_i^\perp, u_j^\perp \rangle$. Hence we get the key observation that for $\forall i, j \in \{0, \ldots, n\}$,

$$\mathbb{E}\left[\widehat{x}_i \widehat{x}_j\right] = \mathop{\mathbb{E}}_{\zeta}\left[\Big(\langle u_i, u_0\rangle + \langle u_i^\perp, \zeta\rangle\Big)\Big(\langle u_j, u_0\rangle + \langle u_j^\perp, \zeta\rangle\Big)\right] = \langle u_i, u_0\rangle\langle u_j, u_0\rangle + \mathop{\mathbb{E}}_{\zeta}\left[\langle u_i^\perp, \zeta\rangle\langle u_j^\perp, \zeta\rangle\right]$$
$$= \langle u_i, u_0\rangle\langle u_j, u_0\rangle + \langle u_i^\perp, u_j^\perp\rangle = \langle u_i, u_j\rangle. \tag{4.8}$$

Note that this also holds when $i = j$. We now consider the expected value of $g(\widehat{x})$. Using (4.8), $\widehat{x}_0 = 1$ and since $\mathbb{E}_\zeta[\langle u_i^\perp, \zeta\rangle] = 0$, we have

$$\mathbb{E}[g(\widehat{x})] = \sum_{i,j=1}^{n} A_{ij} \mathop{\mathbb{E}}_{\zeta}\left[\widehat{x}_i \widehat{x}_j\right] + \sum_{i=1}^{n} b_i \mathop{\mathbb{E}}_{\zeta}[\widehat{x}_i \widehat{x}_0] + c \mathop{\mathbb{E}}_{\zeta}[\widehat{x}_0^2]$$
$$= \sum_{i,j=1}^{n} A_{ij}\langle u_i, u_j\rangle + \sum_{i=1}^{n} b_i \langle u_i, u_0\rangle + c\|u_0\|_2^2 = \mathrm{SDP}_{val}. \tag{4.9}$$

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

$$|\widehat{x}_i| \le |\langle u_i, u_0\rangle| + |\langle u_i^\perp, \zeta\rangle| \le \delta + |\langle u_i^\perp, \zeta\rangle| \le \sqrt{C \log n} \cdot \delta \text{ with probability at least } 1 - 1/n^{C/2},$$

using standard tail properties of Gaussians. Hence, using a union bound over all $i \in [n]$, we have that

$$\mathbb{E}[g(\widehat{x})] \ge \max_{\|x\|_\infty \le \delta} g(x), \quad \text{and } \mathbb{P}\left[\|\widehat{x}\|_\infty \le O(\sqrt{\log n}) \cdot \delta\right] \ge 1 - \frac{1}{n^2}. \tag{4.10}$$

for $C \ge 4$. Further note that $g(\widehat{x})$ can be expressed a degree-d polynomial of the Gaussian vector ζ. Hence using hypercontractivity of low-degree polynomials [O'Donnell, 2014, Theorem

10.23], we have

$$\mathop{\mathbb{P}}_{\zeta}\left[g(\widehat{x}) \geq \mathop{\mathbb{E}}_{\zeta} g(\widehat{x})\right] \geq \Omega(1).$$

Hence (4.7) follows. □

Remark 4.1.5. Obtaining an approximation factor of $O(\gamma)$ in the ℓ_∞ norm of $\hat{z}$, even 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 $O(\log n)$-factor approximation in polynomial time [Charikar and Wirth, 2004; Nesterov, 1998]. 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 approximation factor seems unlikely.

4.2. From Adversarial Examples to Robust Learning Algorithms

In this section we will show how to leverage the algorithms for finding adversarial examples to design polynomial time robust learning algorithms for various sub-classes of Polynomial Threshold Functions (PTF). 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 polynomial maximization problem studied in Section 4.1.This is formalized in the definition below:

Definition 4.2.1 (γ-factor admissibility)**.** For $\gamma \geq 1$, we say that a sub-class $\mathcal{F}$ of PTFs is γ-factor admissible if $\mathcal{F}$ has the following properties:

(1) For any $a, b, c \in \mathbb{R}$, $sgn(f(x)), sgn(g(x)) \in \mathcal{F}$, $sgn(af(x) + bg(x) + c) \in \mathcal{F}$.

(2) For any $b \in \mathbb{R}^n$ and $sgn(g(x)) \in \mathcal{F}$, we have that $sgn(g(x + b)) \in \mathcal{F}$.

(3) There is a γ-admissible approximation for $\{\, g : sgn(g) \in \mathcal{F} \,\}$.

The first two conditions above are natural and are satisfied by many sub-classes of PTFs. The third condition in the above definition concerns the optimization problem studied in Section 4.1. 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 4.2.2. *Let $\mathcal{F}$ be a sub-class of PTFs that is γ-factor admissible for $\gamma \geq 1$. Then $\mathcal{F}$ is γ-approximately robustly learnable.*

Remark 4.2.3. While we state our upper bounds for perturbations measured in the ℓ_∞ norm, we would like to point out that one can define analogously γ-factor admissibility for any ℓ_p norm and the above theorem will still hold true with the new definition.

To learn a $g \in \mathcal{F}$ we formulate robust empirical risk minimization as a convex program, shown in Figure 4.2. Here we use the fact that the value of any polynomial g of degree d at a given point x can be expressed as the inner product between the co-efficient vector of g (denoted by coeff$(g) \in \mathbb{R}^D$) and an appropriate vector $\psi(x) \in \mathbb{R}^D$ where $D = \binom{n+d-1}{d}$. Our

goal is to find a polynomial $g \in \mathcal{F}$ that correctly classifies all the training examples (x_i, y_i). This corresponds to the constraint $y_i g(x_i) > 0$ expressed as $y_i \langle \text{coeff}(g), \psi(x) \rangle > 0$, a linear constraint in the unknown coefficients coeff(g) of the polynomial g. For example, if $g(x)$ is a degree-2 polynomial of the form $x^T Ax + b^T x + c$, 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 ith 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 4.2, where the r_i's are additional variables apart from the coefficients of g. Note that the set of all g is convex because of condition 1 of Definition 4.2.1. While constraints in (4.12) are linear in the variables and easy to implement, (4.13) 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 $\mathcal{F}$ is γ-factor admissible to design an approximate separation oracle for the type of constraints enforced in (4.13). 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; Bertsimas and Sim, 2004; El Ghaoui and Lebret, 1997]. 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 of Theorem 4.2.2. 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 $\mathcal{B}$ be an algorithm

that achieves the γ-factor admissibility for the class $\mathcal{F}$. Given S, we will run the Ellipsoid algorithm on the convex program in Figure 4.2. 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 $sgn(r_i - y_i g(x_i))$, and the algorithm proceeds to the next iteration. If all the constraints in (4.12) 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 ith data point. Furthermore, we will set η', the failure probability of $\mathcal{B}$, to be equal to $\eta/(mT(m,n))$ and set δ' that is input to $\mathcal{B}$ to be δ/γ. From the guarantee of $\mathcal{B}$ we get that if there exists an i such that

$$r_i < \sup_{z \in B_\infty^n(0, \frac{\delta}{\gamma})} y_i \Big(g(x_i) - g(x_i + z) \Big), \tag{4.11}$$

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

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

This gives us a separating hyperplane of the form $sgn(y_i(g(x_i) - g(x_i + \hat{z})) - r_i)$, 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 (4.12) and (4.11) are satisfied. This means that we would have the empirical robust error

(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 $g \in \mathcal{F}$ that satisfies

$$y_i g(x_i) > r_i, \quad \forall i \in [m] \tag{4.12}$$

$$r_i \geq \sup_{z \in B_\infty^n(0,\delta)} y_i \Big(g(x_i) - g(x_i + z) \Big), \quad \forall i \in [m] \tag{4.13}$$

Figure 4.2. The convex program for finding a polynomial $g \in \mathcal{F}$ with zero robust empirical error.

$\hat{err}_{\delta/\gamma,S}(sgn(g)) = 0$. Hence, by Lemma 2.2.3, we get that

$$err_{\delta/\gamma,D}(sgn(g)) \leq 2\sqrt{\frac{2\Delta \log m}{m}} + \sqrt{\frac{\log \frac{1}{\eta}}{2m}},$$

where Δ is the VC dimension of $\mathcal{F}$. Choosing $m = c\frac{\Delta + \log(1/\eta)}{\varepsilon^2}$, makes $err_{\delta/\gamma,D}(sgn(g)) \leq \varepsilon$. $\square$

It is easy to check that for any fixed $d \in \mathbb{N}$, general degree-d PTFs satisfy conditions 1 and 2 of Definition 4.2.1 (however homogeneous degree d polynomials do not satisfy condition 2). We conclude the section by stating the following corollaries about robust learnability of general degree-1 and degree-2 PTFs.We begin with the following claim about admissibility and hence robust learnability of degree-1 PTFs.

Corollary 4.2.4. *The class of degree-*1 *PTFs is optimally robustly learnable.*

The proof just follows from Claim 4.1.2 and since any linear combination or shift of a linear function is also linear. Similarly, the following corollary about degree-2 PTFs is immediate from Theorem 4.1.3.

Corollary 4.2.5. *The class of degree-*2 *PTFs is* $O(\sqrt{\log n})$*-approximately robustly learnable.*

4.3. Finding Adversarial Examples for Two Layer Neural Networks

Next we use the framework in Section 4.1 to design new algorithms for finding adversarial examples in two layer neural networks with ReLU activations. The description that follows applies to binary classification and can be easily extended to multiclass classification. The binary classifier corresponding to the network is $sgn(f_1(x) - f_2(x)) = sgn(v^T\sigma(Wx))$ where $v = v_1 - v_2$. The optimization problem that arises is the following: given an instance with $A \in \mathbb{R}^{m_1 \times n}, \beta \in \mathbb{R}^{m_2}, B \in \mathbb{R}^{m_2 \times n}, c_1 \in \mathbb{R}^n, c_2 \in \mathbb{R}^{m_1}, c_0 \in \mathbb{R}$, the goal is to find $opt(A, B, \beta, c)$, defined as :

$$\begin{aligned} opt(A, B, \beta, c) &:= \max_{z:\|z\|_\infty \leq \delta} \|c_2 + Az\|_1 + c_1^T z - \|\beta + Bz\|_1 + c_0 \\ &= \max_{z:\|z\|_\infty \leq \delta} \max_{y:\|y\|_\infty \leq 1} y^T Az + c_1^T z + c_2^T y - \sum_{j=1}^{m_2} |\beta_j + B_j^T z|. \end{aligned} \tag{4.14}$$

Here B_j is the jth row of B. Let c denote (c_0, c_1, c_2), and let $opt(A, B, \beta, c)$ be the optimal value of the above problem.

To see the connection to polynomial optimization, notice that if $B = 0$, then the above problem is exactly the one we considered in Section 4.1 in the context of degree-2 PTFs. Furthermore, if $A = 0$, then 4.14 is a linear program. However, the presence of both the terms involving A and B make 4.14 a challenging optimization problem. Next we discuss how the problem is related to finding adversarial examples for 2-layer neural networks. A two layer neural network with ReLU gates is given by parameters (v_1, v_2, W) and outputs $f_1(x) = v_1^T\sigma(Wx), f_2(x) = v_2^T\sigma(Wx)$ where $x \in \mathbb{R}^n, v_1, v_2 \in \mathbb{R}^k$ and $W \in \mathbb{R}^{k \times n}$. Here $\sigma : \mathbb{R}^m \to \mathbb{R}^m$

(1) Given instance $I = (A, B, \beta, c)$ of (4.14), solve SDP with parameter $\eta \in (0, 1)$:

$$sdp = \max \sum_{j\in[m_1], i\in[n]} A_{j,i}\langle v_j, u_i\rangle + \sum_{i=1}^{n} c_1(i)\langle u_i, u_0\rangle + \sum_{j=1}^{m_1} c_2(j)\langle u_0, v_j\rangle - \sum_{j\in[m_2]} r_j + c_0$$

$$\text{s.t.} \forall j \in [m_1]\ \|v_i\|^2 \le 1,\ \forall i \in \{1, \ldots, n\}\ \|u_i\|^2 \le \delta^2, \text{ and } \|u_0\|^2 = 1$$

$$\forall j \in [k_2]\ r_j \ge (\beta_j + \sum_j B_{j,i}\langle u_i, u_0\rangle), \text{ and } r_j \ge -(\beta_j + \sum_j B_{j,i}\langle u_i, u_0\rangle).$$

(2) Let $u_i^\perp, v_j^\perp$ represent the components of u_i, v_j orthogonal to u_0. Let $\varepsilon \in (0, 1)$ with $\varepsilon = \Omega(1)/\sqrt{\log m_1}$. Let $\zeta \sim N(0, I)$ be a Gaussian vector; set $\forall i \in \{0, 1, \ldots, n\}$, $\widehat{z}_i := \langle u_i, u_0\rangle + \frac{1}{\varepsilon}\langle u_i^\perp, \zeta\rangle$, $\widehat{y}_j := \langle v_j, u_0\rangle + \varepsilon\langle v_j^\perp, \zeta\rangle$.

(3) Repeat rounding with $poly(n)$ random choices of ζ and pick the best choice.

Figure 4.3. The SDP-based algorithm for Problem (4.14).

is a co-ordinate wise non-linear operator $\sigma(y)_i = \max\{0, y_i\}$ for each $i \in [m]$. The classifier corresponding to the network is $sgn(f_1(x) - f_2(x)) = sgn((v_1 - v_2)^T\sigma(Wx)) = sgn(v^T\sigma(Wx))$.

Our algorithm for solving (4.14) given in Figure 4.3 is inspired by Algorithm 4.1 for polynomial optimization. However, the rounding algorithm differs because the variables y_j and variables z_i serve different purposes in (4.14), and we need to simultaneously satisfy different constraints on them to produce a valid perturbation. Moreover when the SDP is negative, then this gives a certificate of robustness around x.

We remark that one can obtain provable guarantees similar to Theorem 4.2.2 for Algorithm 4.3 under certain regularity conditions about the SDP solution. However, this is unsatisfactory as this depends on the SDP solution to the given instance, as opposed to an explicit structural property of the instance. Obtaining provable guarantees of the latter kind is an interesting open question. The following proposition holds in a more general setting where there can be an extra linear term as described below.

Proposition 4.3.1. *Let $\gamma \geq 1$. Suppose there is an algorithm that given an instance of problem (4.14) finds a solution $\widehat{z}, \widehat{y}$ with $\|\widehat{z}\|_\infty \leq \gamma\delta, \|\widehat{y}\|_\infty \leq 1$ such that $\widehat{y}^T A\widehat{z} + c_1^T\widehat{z} + c_2^T\hat{y} - \|\beta + B\widehat{z}\|_1 + c_0 > 0$ when $opt(A, B, \beta, , c) > 0$, then there is a polynomial time algorithm that given a classifier $sgn(f(x))$ corresponding to a two layer neural net where $f(x) := v^T\sigma(Wx) + (v')^T x$ and an example x^*, guarantees to either (a) find an adversarial example in the ℓ_∞ ball of $\gamma\delta$ around x^*, or (b) certify the absence of any adversarial example in the ℓ_∞ ball of δ.*

Proof. Let $\ell(x^*) = sgn(f(x^*))$. We first observe that $\sigma(y_j) = \frac{1}{2}(|y_j| + y_j)$, and $\sigma(Wx)_j = \frac{1}{2}(|\langle W_j, x\rangle| + \langle W_j, x\rangle)$, where W_j is the jth row of W. We want to find a $\widehat{z}$ with $\|\widehat{z}\|_\infty \leq \gamma\delta$, such that $(-\ell(x^*))f(x^* + \widehat{z}) > 0$, or certify that there is no such $\widehat{z}$ with $\|\widehat{z}\|_\infty \leq \delta$.

Let $S_+ = \{\, j \in [k] : -\ell(x^*)v_j \geq 0 \,\}$ and $S_- = [k] \setminus S_+$ and let $k_1 = |S_+|$. We now split the rows of W into two (A and B) as follows: for every $j \in S_+$, define the row $A_j := \frac{1}{2}|v_j|W_j$; otherwise let $B_j := \frac{1}{2}|v_j|W_j$.

$$\begin{aligned} -\ell(x^*)f(x^* + z) &= \tfrac{1}{2}\sum_{j\in S_+} |v_j||\langle W_j, x^* + z\rangle| + \tfrac{1}{2}\langle v^T W, x^* + z\rangle - \tfrac{1}{2}\sum_{j\in S_-} |v_j||\langle W_j, x^* + z\rangle| \\ &= \max_{y\in\{\,-1,1\,\}^{k_1}} \sum_{j\in S_+} y_j\langle A_j, x^* + z\rangle - \sum_{j\in S_-} |\langle B_j, x^* + z\rangle| + c_1^T z + c_0, \end{aligned}$$

where $c_1^T = \frac{1}{2}v^T W + (v')^T$ and $c_0 = \frac{1}{2}v^T W x^*$ are constants. Since the dependence on y is linear we also get by substituting $c_2 := Ax^*$, $\beta := Bx^*$,

$$\max_{\|z\|_\infty\leq\delta} (-\ell(x^*))f(x^* + z) = \max_{\|z\|_\infty\leq\delta} \max_{y:\|y\|_\infty\leq 1} \sum_{j\in S_+} y_j\langle A_j, z\rangle + c_2^T y + c_1^T z - \sum_{j\in S_-} |\beta_j + \langle B_j, z\rangle| + c_0,$$

as required. Now the proposition follows from the same argument as in Proposition 4.1.1. □

CHAPTER 5

Adversarial Learning : Lower Bound

5.1. Computational Intractability of Learning Robust Classifiers

In this section, we leverage the connection to polynomial optimization to complement our upper bound with the following nearly matching lower bound.We give a reduction from *Quadratic Programming (QP)* where given a polynomial $p(x) = \sum_{i<j} a_{ij} x_i x_j$, and a value s, the goal is to distinguish whether $max_{x \in \{-1,1\}^n} p(x) < s$ or whether exists an x such that $p(x) > s\eta_{approx}$. It is known that the distinguishing problem is hard for $\eta_{approx} = O(\log^c n)$ for some constant $c > 0$, see Arora et al. [2005]; moreover the state-of-the-art algorithms give a $\eta_{approx} = O(\log n)$ factor approximation, see Charikar and Wirth [2004] and improving upon this factor is a major open problem. By appropriately scaling the instance, this immediately implies the hardness of checking whether a given degree-2 PTF is robust around a given point.

However, this does not suffice for hardness of learning, since given a distribution supported at a single point, there is a trivial constant classifier that robustly classifies the instance correctly. More generally, there could exist a different degree-2 PTF that could be easy to certify for the given point. Instead, given a degree-2 PTF $sgn(p(x))$, we carefully construct a set of $O(n^2)$ points such that any classifier that is robust on an instance supported on the set will have to be close to the given polynomial p. Having established this, we can distinguish

between the two cases of the *QP* problem by whether the learning algorithm is able to output a robust classifier or not. This is formalized below.

Theorem 5.1.1. *There exists $\delta, \varepsilon > 0$, such that assuming $NP \neq RP$ there is no algorithm that given a set of $N = poly(n, \frac{1}{\varepsilon})$ samples from a distribution D over $\mathbb{R}^n \times \{-1, +1\}$, runs in time* poly$(N)$ *and distinguishes between the following two cases for any $\delta' = o(\sqrt{\eta_{approx}}\delta)$:*

- YES: *There exists a degree-2 PTF that has δ-robust error of* 0 *w.r.t. D.*
- NO: *There exists no degree-2 PTF that has δ'-robust error at most ε w.r.t. D.*

Here η_{approx} is the hardness of approximation factor of the QP *problem.*

Remark 5.1.2. The above theorem proves that any polynomial time algorithm that always outputs a robust classifier (or declares failure if it does not find one) will have to incur an extra factor of $\Omega(\sqrt{\eta_{approx}})$ in the robustness parameter δ. Our upper bound in Section 4.2 on the other hand matches this bound.

While our lower bound applies to algorithms that output a classifier of low error, in Section 5.2 (see Theorem 5.2.1) we also prove a more robust lower bound that rules out the possibility of an efficient robust learner that incurs an error less than 1/4.

We will represent an instance of *Quadratic Programming (QP)* by a polynomial $p(x) = x^T Ax$ where A is a symmetric matrix with zeros on the diagonal, and $A_{ij} = A_{ji} = a_{ij}/2$. Formally, the NP-hard problem $\mathcal{QP}$ [Arora et al., 2005; Garey and Johnson, 2002] is the following: given $\beta > 0$ and a polynomial $p(x) = x^T Ax$ distinguish whether

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

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

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 δ robust at all the points or that no degree-2 PTF is $\eta\delta$ robust for $\eta = \Omega(1/\sqrt{\eta_{approx}})$.

Theorem 5.1.3. *[Hardness] There exists $\delta > 0$, such that 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 distinguish between the following two cases*

YES Case: *There exists a degree-2 PTF that has δ-robust empirical error of 0 on these N points.*

NO Case: *No degree-2 PTF is $\eta\delta$-robust on these points for $\eta = \Omega(1/\sqrt{\eta_{approx}})$.*

Theorem 5.1.1 follows from Theorem 5.1.3 and 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 $\varepsilon > 0$, then one could use it on the uniform distribution over the instance from Theorem 5.1.3 with $\varepsilon = \frac{1}{2N}$ to determine whether there exists a degree-2 PTF that has δ-robust empirical error of 0 on the points in the instance. Hence our main goal is to prove Theorem 5.1.3.

5.1.1. Warm up Lower Bound

Before we prove Theorem 5.1.3 we state and prove a simpler version of the Theorem where we show hardness of approximation of all degree-2 homogeneous PTFs. In this case we only

have to deal with the homogeneous second order term and the degree-1 z term, and thus do not have to control the other terms.

Note : In what follows, we slightly abuse terminology and refer to a polynomial of the form $p(x, z) = x^T Ax - z$ as homogeneous even though it has the extra z term. For the sake of simplicity, we do not define a new term.

Theorem 5.1.4. *[Simpler Version of Theorem 5.1.3] There exists $\delta > 0$ such that 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 distinguish between the following two cases*

YES Case: *There exists a* ***homogeneous*** *degree-2 PTF that has δ-robust empirical error of 0 on these N points.*

NO Case: *No* ***homogeneous*** *degree-2 PTF (of the form $q(x, z) = x^T Ax - z$) is $\eta\delta$-robust on these points for $\eta = \Omega(1/\sqrt{\eta_{approx}})$.*

Proof. We carefully define the point set S'. We want to achieve two things. First, in the **YES** case, the polynomial $p(x, z) = x^T Ax - z$ should be δ robust to all points in S'. Second, the correctness of any other deg-2 homogeneous polynomial $q(x, z) = x^T A'x - z$ on S' will force it to be close to $p(x, z)$ in terms of its parameters. This will ensure that in the **NO** case, if $q(x, z)$ is $\eta\delta$ robust to S' then $x^T A'x$ will be upper bounded around $\mathbf{0}$ and so will $x^T Ax$, thereby leading to a contradiction. Formally, for any $\varepsilon, \delta < 1/10$ let

$$S' = \{((0, 2\delta) - 1)\} \bigcup \{((\mathbf{e}_i, \gamma), -1), ((\mathbf{e}_i, -\gamma), 1) \forall i \in [n]\}$$

$$\bigcup \{((\mathbf{e}_{i,j}, 2), -1), ((2\mathbf{e}_{i,j}, 1), \text{sgn}(a_{i,j})) \forall i \neq j \in [n]\}$$

where $\mathbf{e}_i$ is the vector $(0, 0, \ldots, \tau, 0, \ldots, 0)$ and $\mathbf{e}_{i,j}$ is the vector $(0, 0, \ldots, \frac{1}{\sqrt{2(\varepsilon + |a_{i,j}|)}}, 0, \ldots, \frac{1}{\sqrt{2(\varepsilon + |a_{i,j}|)}}, 0, \ldots, 0)$. We now state the two cases

YES case: $\max_{x \in \{-1,1\}^n} x^T A x < s$

NO case: $\max_{x \in \{-1,1\}^n} x^T A x > s\eta_{approx}$

The reduction is as follows :

(1) Scale the entries of A such that each non zero entry is greater than 10. Scale s by the same factor. Set $\delta = 1/s$ and $\varepsilon = 200/n^2$.
(2) Generate the labeled point set S' in $\mathbb{R}^{n+1}$ with $\tau = \Omega(\frac{n}{\varepsilon}) \max(1, 1/(\varepsilon + \min_{i \neq j} |a_{i,j}|))$, $\gamma = 4n\tau$.

Figure 5.1. Reduction from the QP problem.

Under the **NO Case** we analyse soundness of the reduction :

Claim 5.1.5. *There does not exist an $\eta\delta$-robust degree-2 polynomial on S for $\eta = \Omega(1/\sqrt{\eta_{approx}})$*

Proof. First we show that if $q(x, z) = x^T A' x - z$ correctly classifies S' then A' is entrywise close to A. First we show that the diagonal entries of A' are less than ε. From correctness at $(\mathbf{e}_i, \gamma)$ and $(\mathbf{e}_i, -\gamma)$ we get

$$-\gamma < \tau^2 a'_{i,i} < \gamma \tag{5.1}$$

$$|a'_{i,i}| < \gamma/\tau^2 < \varepsilon/10 \tag{5.2}$$

From correctness at $(\mathbf{e}_{i,j}, 2)$ and $(\mathbf{e}_{i,j}, -2)$ we get :

$$\frac{a'_{i,i}}{2\tilde{a}_{i,j}} + \frac{a'_{j,j}}{2\tilde{a}_{i,j}} + \frac{a'_{i,j}}{\tilde{a}_{i,j}} < 2 \tag{5.3}$$

$$2\frac{a'_{i,i}}{\tilde{a}_{i,j}} + 2\frac{a'_{j,j}}{\tilde{a}_{i,j}} + 4\frac{a'_{i,j}}{\tilde{a}_{i,j}} > 1 \tag{5.4}$$

Combining them we get

$$\frac{1}{4} - \delta - \frac{\varepsilon}{4} < \frac{a'_{i,j}}{\tilde{a}_{i,j}} < 2 + 4\delta + \varepsilon \tag{5.5}$$

This gives us the necessary parameter closeness between A and A' because it implies that

$$\max_{x \in B^n_\infty(0,\eta\delta)} x^T A x = \Theta(\max_{x \in B^n_\infty(0,\eta\delta)} x^T A' x)$$

. Now we prove by contradiction. Let $q(x, z)$ be $\eta\delta$-robust on S'. The fact that $q(x, z)$ is $\eta\delta$-robust on $(0, 2\delta)$ gives us :

$$\max_{x \in B_\infty^n(0,\eta\delta)} x^T A' x < 2\delta \tag{5.6}$$

$$\max_{x \in B_\infty^n(0,\delta)} x^T A' x < \frac{2\delta}{\eta^2} \tag{5.7}$$

$$\max_{x \in B_\infty^n(0,\delta)} x^T A x < \frac{5\delta}{\eta^2} \tag{5.8}$$

where the last inequality follows from our earlier observation. However since we are in the NO case, we have :

$$\max_{x \in B_\infty^n(0,\delta)} x^T A x > \delta^2 s \eta_{approx} = \delta \eta_{approx} \tag{5.9}$$

This contradicts the fact $\eta = \Omega(1/\eta_{approx})$ for some appropriately chosen constant. □

Under the **YES Case** we analyse completeness by showing δ-robustness of $p(x, z)$ on S' :

Claim 5.1.6. *The polynomial $p(x, z) = x^T A x - z$ is δ-robust on S'*

The proof of this is the same as in the main Theorem since S' is a subset of S. See 5.1.9.

□

5.1.2. Main Lower Bound

For the proof of the main theorem , we use a similar idea. As before, our set of points will have the property that in the YES case of the QP instance, the polynomial $x^T A x - z$ will be δ robust at all the points (see Claim 5.1.9). Furthermore, the points will enforce the property

that any other degree-2 PTF that classifies the points correctly will have to be very close to $x^TAx - z$ in terms of the parameters. In this case, apart from the parameters in the quadratic term, we will also control the linear and constant terms. This will help us rule out the existence of an $\eta\delta$ robust classifier in the NO case, since if one exists, it must be close to $x^TAx - z$, thereby implying an upper bound on the value of x^TAx around the neighborhood of zero. This is established in the following key lemma.

Lemma 5.1.7. *Let $p(x, z) = x^TAx - z$ be a given polynomial where A is a symmetric matrix with zeros on the diagonal. For any $\varepsilon, \delta < 1/10$, consider the labeled set $S = S_1 \cup S_2 \cup S_3 \cup S_4 \cup S_5$ where,*

$$S_1 = \{((\mathbf{0}, 1), -1), ((\mathbf{0}, -1), +1), ((\mathbf{0}, \tau'), -1), ((\mathbf{0}, -\tau'), +1), ((\mathbf{0}, 2\delta), -1), ((\mathbf{0}, -2\delta), +1)\},$$

$$S_2 = \{((\mathbf{e}_i, \gamma), -1), ((\mathbf{e}_i, -\gamma), +1), ((-\mathbf{e}_i, \gamma), -1), ((-\mathbf{e}_i, -\gamma), +1)\},\ \forall i \in [n],$$

$$S_3 = \{((\mathbf{e}_{i,j}, 2), -1), ((\mathbf{e}_{-i,j}, 2), -1), ((\mathbf{e}_{i,-j}, 2), -1), ((\mathbf{e}_{-i,-j}, 2), -1)\},\ \forall i \neq j \in [n],$$

$$S_4 = \{((2\mathbf{e}_{i,j}, 1), sgn(a_{i,j})), ((2\mathbf{e}_{-i,j}, 1), -sgn(a_{i,j})), ((2\mathbf{e}_{i,-j}, 1), -sgn(a_{i,j})),$$
$$((2\mathbf{e}_{-i,-j}, 1), sgn(a_{i,j}))\},\ \forall i \neq j \in [n],$$

and

$$S_5 = \{((\mathbf{e}_{i,j}, -2), +1), ((\mathbf{e}_{-i,j}, -2), +1), ((\mathbf{e}_{i,-j}, -2), +1), ((\mathbf{e}_{-i,-j}, -2), +1)\},\ \forall i \neq j \in [n],$$

Here $\mathbf{e}_i$ *is the vector* $(0, 0, \ldots, \tau, 0, \ldots, 0)$ *and* $\mathbf{e}_{i,j}$ *is the vector* $(0, 0, \ldots, \frac{1}{\sqrt{2(\varepsilon+|a_{i,j}|)}}, 0, \ldots, \frac{1}{\sqrt{2(\varepsilon+|a_{i,j}|)}}, 0, \ldots, 0)$. *For every general degree 2 polynomial* $q'(x, z)$ *with the coefficient of* $z = c_z$, *such that* $sgn(q')$ *has zero error on* S, *we must have* $c_z \neq 0$. *Moreover, let* $q(x, z) = \frac{1}{-c_z} q'(x, z) = x^T A' x + c_1^T x + c_2 z^2 - z + c_4 + \sum_i \beta_i z x_i$, *where* A' *be a symmetric matrix. Then we must have that*

$$\max(|c_2|, \|\beta\|_\infty, |a'_{i,i}|) \leq \varepsilon,$$

$$|c_4| \leq 4\delta,$$

$$|c_{1,i}| \leq \min_{j \neq i} 8\delta \sqrt{\varepsilon + |a_{i,j}|},$$

and

$$\frac{1}{4} - \delta - \frac{\varepsilon}{4} \leq \max\left(\frac{|a'_{i,j}|}{\varepsilon + |a_{i,j}|}\right) \leq 2 + 4\delta + \varepsilon$$

provided $\tau' = \Omega(\frac{n^2}{\varepsilon}) \max(1, 1/(\varepsilon + \min_{i \neq j} |a_{i,j}|)), \tau = \Omega(\frac{n}{\varepsilon}) \max(1, 1/(\varepsilon + \min_{i \neq j} |a_{i,j}|))$, $\gamma = 4n\tau$.

We first prove Theorem 5.1.3 assuming the lemma above and finally end the section with the proof of the lemma. This proof relies on the fact stated above that if any degree-2 PTF correctly classifies all these points, then it must be close to $x^T Ax - z$ in parameter space. This implies that in the **NO** case, since $x^T Ax - z$ is bounded away from 0 any degree-2 PTF close to the polynomial must also be bounded away from 0. This however contradicts the $\eta\delta$ robustness of the polynomial to S. The **YES** case of the proof is relatively simple; we just have to verify that $x^T Ax - z$ correctly classifies and is δ robust to all the points in S.

PROOF OF THEOREM 5.1.3. Given an $n \times n$ symmetric matrix A with zeros on diagonals and given $s > 100$, we assume that the following cases are hard to distinguish for some $\eta_{approx} > 1$,

YES Case: $\max_{x \in \{-1,1\}^n} x^T A x < s$.

NO Case: $\max_{x \in \{-1,1\}^n} x^T A x > s\eta_{approx}$. The reduction from the instance of the QP problem is sketched below. Next we establish completeness and soundness of the reduction.

(1) Scale the entries of A such that each non zero entry is greater than 10. Scale s by the same factor. Set $\delta = 1/s$ and $\varepsilon = 200/n^2$.
(2) Generate the labeled point set S in $\mathbb{R}^{n+1}$ as specified in Lemma 5.1.7 with $\tau' = \Omega(\frac{n^2}{\varepsilon}) \max(1, 1/(\varepsilon + \min_{i \neq j} |a_{i,j}|))$, $\tau = \Omega(\frac{n}{\varepsilon}) \max(1, 1/(\varepsilon + \min_{i \neq j} |a_{i,j}|))$, $\gamma = 4n\tau$.

Figure 5.2. Reduction from the QP problem.

NO Case: The following claim captures the soundness analysis of the reduction.

Claim 5.1.8. *There does not exist an $\eta\delta$-robust degree-2 polynomial on S for $\eta = \Omega(1/\sqrt{\eta_{approx}})$.*

PROOF. We will prove by contradiction. Let $q(x, z) = x^T A' x + c_1^T x + c_2 z^2 - z + c_4 + \sum_i \beta_i z x_i$ be an $\eta\delta$-robust polynomial on S.[1] The fact that q is correct on $(\mathbf{0}, 2\delta)$ gives us

$$4c_2\delta^2 - 2\delta + c_4 < 0 \tag{5.10}$$

Furthermore, the fact that q is $\eta\delta$-robust on $(\mathbf{0}, 2\delta)$ gives us that

$$\max_{x \in B_\infty^n(0, \eta\delta), z \in (2\delta - \eta\delta, 2\delta + \eta\delta)} q(x, z) < 0 \tag{5.11}$$

[1] We can always scale q to get it into this form.

From Lemma 5.1.7 this implies that

$$\max_{x \in B_\infty^n(0,\eta\delta)} x^T A' x < \eta\delta\|c_1\|_1 + (2\delta + \eta\delta) + 12\delta + \varepsilon(2\delta + \eta\delta)^2 + n\varepsilon\eta\delta(2\delta + \eta\delta) \tag{5.12}$$

We now need to bound $\|c_1\|_1$.

$$\begin{aligned}
\|c_1\|_1 &< 8\delta(n\sqrt{\varepsilon} + \sum_i \min_{j \neq i} \sqrt{|a_{i,j}|}) \\
&< 8\delta(n\sqrt{\varepsilon} + \sum_i \min_{j \neq i} |a_{i,j}|) \\
&< c'\delta + 16\delta \frac{\sum_{i,j} |a_{i,j}|}{n} \\
&< c'\delta + 16\delta s = c'\delta + 16
\end{aligned}$$

Substituting the value of ε in 5.12 we get that

$$\max_{x \in B_\infty^n(0,\eta\delta)} x^T A' x < 20\delta. \tag{5.13}$$

Again using Lemma 5.1.7 we get that

$$\max_{x \in B_\infty^n(0,\delta)} x^T A x < \frac{50\delta}{\eta^2}. \tag{5.14}$$

But since we are in the NO case we also know that

$$\max_{x \in B_\infty^n(0,\delta)} x^T A x > \delta^2 s \eta_{approx} = \delta \eta_{approx}. \tag{5.15}$$

This contradicts the fact that $\eta = \Omega(1/\sqrt{\eta_{approx}})$. □

YES Case: The analysis of the YES case relies on the following claim which establishes δ-robustness of the PTF given by $p(x, z)$ on the point in S.

Claim 5.1.9. *The polynomial $p(x, z) = x^T Ax - z$ is δ-robust on S.*

Proof. It is fairly straightforward to check that $sgn(x^T Ax - z)$ classifies all of S correctly.

Robustness at $((\mathbf{0}, 2\delta), -1)$. Follows from the fact that we are in the YES case and hence $\max_{x \in B^n_\infty(0,\delta)} x^T Ax < \delta^2 s = \delta$.

Robustness at $((\mathbf{0}, 1), -1), ((\mathbf{0}, \tau'), -1), ((\mathbf{0}, -1), +1), ((\mathbf{0}, -\tau'), +1)$. Follows from the fact that we are in the YES case and hence $\max_{x \in B^n_\infty(0,\delta)} x^t Ax < \delta^2 s = 1/s < 1/100$ and that $\tau' > n/(20\delta) > 5n$.

Robustness at $((\mathbf{0}, 2\delta), -1), ((\mathbf{0}, -2\delta), +1)$. Follows from the fact that we are in the YES case and hence $\max_{x \in B^n_\infty(0,\delta)} x^T Ax < \delta^2 s = \delta$ and that $\varepsilon n/10 = 20\delta$.

Robustness at $((\mathbf{e}_i, \gamma), -1), ((\mathbf{e}_i, -\gamma), +1), ((-\mathbf{e}_i, \gamma), -1), ((-\mathbf{e}_i, -\gamma), +1)$. Let's argue robustness at $((\mathbf{e}_i, \gamma), -1)$ and the other calculations are similar. The maximum value of $x^T Ax$ in a δ-ball around $\mathbf{e}_i$ is at most

$$(\tau + \delta)\delta \sum_j |a_{i,j}| + \delta^2 s.$$

Hence to establish robustness we need that

$$(\tau + \delta)\delta \sum_j |a_{i,j}| + \delta^2 s \leq \gamma - \delta. \tag{5.16}$$

Substituting the value of δ and noticing that γ, τ are much larger than $\delta = 1/s < 1/100$ we get that it is enough for the following to hold

$$2\tau\delta \sum_j |a_{i,j}| \leq \frac{\gamma}{2}. \tag{5.17}$$

In other words we need that

$$\frac{\gamma}{\tau} \geq 4\delta \sum_j |a_{i,j}| \tag{5.18}$$

Substituting the values of γ, τ we get that

$$n \geq \delta \sum_j |a_{i,j}| \tag{5.19}$$

This is true since $\delta = 1/s$ and the fact that $s \geq \frac{1}{n}\sum_{i,j} |a_{i,j}| > \frac{1}{n}\sum_j |a_{i,j}|$ where the first inequality is from [Charikar and Wirth, 2004].

Robustness at $((\mathbf{e}_{i,j}, 2), -1), ((\mathbf{e}_{-i,j}, 2), -1), ((\mathbf{e}_{i,-j}, 2), -1), ((\mathbf{e}_{-i,-j}, 2), -1)$. Let's argue robustness at $((\mathbf{e}_{i,j}, 2), -1)$ and the other calculations are similar. The maximum value of $x^T Ax$ in a δ-ball around $\mathbf{e}_{i,j}$ is at most

$$\frac{2\delta \max_i \sum_j |a_{i,j}|}{\sqrt{2(\varepsilon + |a_{i,j}|)}} + \delta^2 s + 1$$

Hence to establish robustness we need that

$$\frac{2\delta \max_i \sum_j |a_{i,j}|}{\sqrt{2(\varepsilon + |a_{i,j}|)}} + \delta^2 s + 1 \leq 2 - \delta. \tag{5.20}$$

Noticing that $\delta = 1/s$ and much smaller than $1/100$, we get that it is enough for the following to hold

$$\frac{\delta \max_i \sum_j |a_{i,j}|}{\sqrt{2(\varepsilon + |a_{i,j}|)}} \leq \frac{1}{4}. \tag{5.21}$$

This is again true since $\delta = 1/s$ and by our assumption $|a_{i,j}| \geq 4$ for non-zero entries of A.

Robustness at $((2\mathbf{e}_{i,j}, 1), sgn(a_{i,j})), ((2\mathbf{e}_{-i,j}, 1), -sgn(a_{i,j})), ((2\mathbf{e}_{i,-j}, 1), -sgn(a_{i,j})), ((2\mathbf{e}_{-i,-j}, 1), sgn(a_{i,j}))$. We'll argue robustness at $((2\mathbf{e}_{i,j}, 1), +1)$ and the other calculations are similar. Also for simplicity, assume $sgn(a_{i,j}) > 0$. The other case is similar. The minimum value of $x^T Ax$ in a δ-ball arond $\mathbf{e}_{i,j}$ is at least

$$2 - \frac{2\delta \max_i \sum_j |a_{i,j}|}{\sqrt{2(\varepsilon + |a_{i,j}|)}} - \delta^2 s$$

So for robustness, we need

$$2 - \frac{2\delta \max_i \sum_j |a_{i,j}|}{\sqrt{2(\varepsilon + |a_{i,j}|)}} - \delta^2 s > 1 + \delta$$

This is true since we have

$$\frac{\delta \max_i \sum_j |a_{i,j}|}{\sqrt{2(\varepsilon + |a_{i,j}|)}} \leq \frac{1}{4}.$$

□

This ends the proof of Theorem 5.1.3. All that is left is the proof of Lemma 5.1.7. □

Bounding coefficients of q(x,z) : Lemma 5.1.7 bounds all the coefficients of any degree-2 polynomial that is correct on S. The point set S consists of five sets of points. Set S_1 is used to bound the constant terms. Set S_2 is used to get a bound on the diagonal terms and β_i.

Set S_3 gives an upper bound on the ratio between the entries of A and A' while set S_4 gives a lower bound. Sets S_3 and S_5 also bounds the linear term c_1.

Proof of Lemma 5.1.7. We now prove the key lemma that is used in the analysis of our reduction.

Proof. First we prove that if $q'(x, z)$ has zero error on S then c_z must be non zero. Then it is clear that if $q'(x, z)$ has zero error on S, then so does $q(x, z)$. Consider the case when $c_z = 0$. Now $q'(x, z)$ classifies S_1 correctly. More specifically, it classifies the two points $((\mathbf{0}, 1), -1)$ and $((\mathbf{0}, -1), 1)$ correctly. This gives us the following equations

$$c_2 + c_4 < 0$$

$$c_2 + c_4 > 0$$

and hence we get a contradiction. Moving on to the main part of the proof about the coefficients of $q(x, z)$, the constraints at $(\mathbf{0}, 1), (\mathbf{0}, -1), (\mathbf{0}, \tau'), (\mathbf{0}, -\tau')$ give us

$$c_2 - 1 + c_4 < 0 \tag{5.22}$$

$$c_2 + 1 + c_4 > 0 \tag{5.23}$$

$$\tau'^2 c_2 - \tau' + c_4 < 0 \tag{5.24}$$

$$\tau'^2 c_2 + \tau' + c_4 > 0 \tag{5.25}$$

From (5.22) and (5.23) we get that

$$-1 < c_2 + c_4 < 1 \tag{5.26}$$

Similarly, from (5.24) and (5.25) we get that

$$-\tau' < \tau'^2 c_2 + c_4 < \tau' \tag{5.27}$$

This implies that $|c_2| < 1/(\tau' - 1) < \varepsilon/10$ for $\tau' = \Omega(1/\varepsilon)$.

The constraints at $((\mathbf{0}, 2\delta), -1), ((\mathbf{0}, -2\delta)$ gives us that

$$4c_2\delta^2 - 2\delta + c_4 < 0$$

$$4c_2\delta^2 + 2\delta + c_4 > 0$$

From the above equations we get that

$$|c_4| \leq c_2(2\delta)^2 + 2\delta < 4\delta. \tag{5.28}$$

The constraints at $(\mathbf{e}_i, \gamma), (-\mathbf{e}_i, \gamma), (\mathbf{e}_i, -\gamma), (-\mathbf{e}_i, -\gamma)$ give us

$$\tau^2 a'_{i,i} + \tau c_{1,i} + c_2\gamma^2 - \gamma + c_4 + \tau\gamma\beta_i < 0 \tag{5.29}$$

$$\tau^2 a'_{i,i} - \tau c_{1,i} + c_2\gamma^2 - \gamma + c_4 - \tau\gamma\beta_i < 0 \tag{5.30}$$

$$\tau^2 a'_{i,i} + \tau c_{1,i} + c_2\gamma^2 + \gamma + c_4 - \tau\gamma\beta_i > 0 \tag{5.31}$$

$$\tau^2 a'_{i,i} - \tau c_{1,i} + c_2\gamma^2 + \gamma + c_4 + \tau\gamma\beta_i > 0 \tag{5.32}$$

From (5.29) and (5.32) we get that

$$\tau c_{1,i} < \gamma \tag{5.33}$$

Similarly, from (5.30) and (5.31) we get that

$$\tau c_{1,i} > -\gamma \tag{5.34}$$

Plugging back into the equations above we get that

$$-(4\delta + 2\gamma + \frac{\gamma^2}{\tau' - 1}) < \tau^2 a'_{i,i} + \tau\gamma\beta_i < (4\delta + 2\gamma + \frac{\gamma^2}{\tau' - 1}) \tag{5.35}$$

and

$$-(4\delta + 2\gamma + \frac{\gamma^2}{\tau' - 1}) < \tau^2 a'_{i,i} - \tau\gamma\beta_i < (4\delta + 2\gamma + \frac{\gamma^2}{\tau' - 1}) \tag{5.36}$$

This implies that

$$|a'_{i,i}| \leq \frac{1}{\tau^2}(4\delta + 2\gamma + \frac{\gamma^2}{\tau' - 1}) \leq \varepsilon/10$$

for $\tau' = \Omega(\frac{n^2}{\varepsilon}) \max(1, 1/\min_{i,j} |a_{i,j}|), \tau = \Omega(\frac{n}{\varepsilon}) \max(1, 1/\min_{i,j} |a_{i,j}|)$, $\gamma = 4n\tau$. We also get that

$$|\beta_i| \leq \frac{1}{\tau\gamma}(4\delta + 2\gamma + \frac{\gamma^2}{\tau' - 1}) \leq \varepsilon/10$$

for $\tau' = \Omega(\frac{n^2}{\varepsilon}) \max(1, 1/\min_{i,j} |a_{i,j}|), \tau = \Omega(\frac{n}{\varepsilon}) \max(1, 1/\min_{i,j} |a_{i,j}|)$, $\gamma = 4n\tau$.

The constraints at $(\mathbf{e}_{i,j}, 2), (\mathbf{e}_{-i,j}, 2), (\mathbf{e}_{i,-j}, 2), (\mathbf{e}_{-i,-j}, 2)$ give us

$$\frac{a'_{i,i}}{2\tilde{a}_{i,j}} + \frac{a'_{j,j}}{2\tilde{a}_{i,j}} + \frac{a'_{i,j}}{\tilde{a}_{i,j}} + \frac{c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} + \frac{c_{1,j}}{\sqrt{2\tilde{a}_{i,j}}} + 4c_2 - 2 + c_4 + \frac{2\beta_i}{\sqrt{2\tilde{a}_{i,j}}} + \frac{2\beta_j}{\sqrt{2\tilde{a}_{i,j}}} < 0 \tag{5.37}$$

$$\frac{a'_{i,i}}{2\tilde{a}_{i,j}} + \frac{a'_{j,j}}{2\tilde{a}_{i,j}} - \frac{a'_{i,j}}{\tilde{a}_{i,j}} - \frac{c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} + \frac{c_{1,j}}{\sqrt{2\tilde{a}_{i,j}}} + 4c_2 - 2 + c_4 - \frac{2\beta_i}{\sqrt{2\tilde{a}_{i,j}}} + \frac{2\beta_j}{\sqrt{2\tilde{a}_{i,j}}} < 0 \tag{5.38}$$

$$\frac{a'_{i,i}}{2\tilde{a}_{i,j}} + \frac{a'_{j,j}}{2\tilde{a}_{i,j}} - \frac{a'_{i,j}}{\tilde{a}_{i,j}} + \frac{c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} - \frac{c_{1,j}}{\sqrt{2\tilde{a}_{i,j}}} + 4c_2 - 2 + c_4 + \frac{2\beta_i}{\sqrt{2\tilde{a}_{i,j}}} - \frac{2\beta_j}{\sqrt{2\tilde{a}_{i,j}}} < 0 \tag{5.39}$$

$$\frac{a'_{i,i}}{2\tilde{a}_{i,j}} + \frac{a'_{j,j}}{2\tilde{a}_{i,j}} + \frac{a'_{i,j}}{\tilde{a}_{i,j}} - \frac{c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} - \frac{c_{1,j}}{\sqrt{2\tilde{a}_{i,j}}} + 4c_2 - 2 + c_4 - \frac{2\beta_i}{\sqrt{2\tilde{a}_{i,j}}} - \frac{2\beta_j}{\sqrt{2\tilde{a}_{i,j}}} < 0 \tag{5.40}$$

where $\tilde{a}_{i,j} = \varepsilon + |a_{i,j}|$. Combining (5.37) and (5.40) we get

$$\frac{a'_{i,i}}{2\tilde{a}_{i,j}} + \frac{a'_{j,j}}{2\tilde{a}_{i,j}} + \frac{a'_{i,j}}{\tilde{a}_{i,j}} + 4c_2 - 2 + c_4 < 0 \tag{5.41}$$

From this we get that

$$\frac{a'_{i,j}}{\tilde{a}_{i,j}} < 2 + 4\delta + 4\frac{\varepsilon}{10} + \frac{4\delta + 2\gamma + \frac{\gamma^2}{\tau'-1}}{\tau^2 \min_{i,j} |a_{i,j}|} < 2 + 4\delta + \varepsilon \tag{5.42}$$

for large enough τ. Similarly, combining (5.38) and (5.39) we get

$$\frac{a'_{i,i}}{2\tilde{a}_{i,j}} + \frac{a'_{j,j}}{2\tilde{a}_{i,j}} - \frac{a'_{i,j}}{\tilde{a}_{i,j}} + 4c_2 - 2 + c_4 < 0 \tag{5.43}$$

From this we get that

$$\frac{a'_{i,j}}{\tilde{a}_{i,j}} > -2 - 4\delta - \varepsilon. \tag{5.44}$$

The constraints at $(\mathbf{e}_{i,j}, -2), (\mathbf{e}_{-i,j}, -2), (\mathbf{e}_{i,-j}, -2), (\mathbf{e}_{-i,-j}, -2)$ give us

$$\frac{a'_{i,i}}{2\tilde{a}_{i,j}} + \frac{a'_{j,j}}{2\tilde{a}_{i,j}} + \frac{a'_{i,j}}{\tilde{a}_{i,j}} + \frac{c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} + \frac{c_{1,j}}{\sqrt{2\tilde{a}_{i,j}}} + 4c_2 + 2 + c_4 + \frac{2\beta_i}{\sqrt{2\tilde{a}_{i,j}}} + \frac{2\beta_j}{\sqrt{2\tilde{a}_{i,j}}} > 0 \tag{5.45}$$

$$\frac{a'_{i,i}}{2\tilde{a}_{i,j}} + \frac{a'_{j,j}}{2\tilde{a}_{i,j}} - \frac{a'_{i,j}}{\tilde{a}_{i,j}} - \frac{c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} + \frac{c_{1,j}}{\sqrt{2\tilde{a}_{i,j}}} + 4c_2 + 2 + c_4 - \frac{2\beta_i}{\sqrt{2\tilde{a}_{i,j}}} + \frac{2\beta_j}{\sqrt{2\tilde{a}_{i,j}}} > 0 \tag{5.46}$$

$$\frac{a'_{i,i}}{2\tilde{a}_{i,j}} + \frac{a'_{j,j}}{2\tilde{a}_{i,j}} - \frac{a'_{i,j}}{\tilde{a}_{i,j}} + \frac{c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} - \frac{c_{1,j}}{\sqrt{2\tilde{a}_{i,j}}} + 4c_2 + 2 + c_4 + \frac{2\beta_i}{\sqrt{2\tilde{a}_{i,j}}} - \frac{2\beta_j}{\sqrt{2\tilde{a}_{i,j}}} > 0 \tag{5.47}$$

$$\frac{a'_{i,i}}{2\tilde{a}_{i,j}} + \frac{a'_{j,j}}{2\tilde{a}_{i,j}} + \frac{a'_{i,j}}{\tilde{a}_{i,j}} - \frac{c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} - \frac{c_{1,j}}{\sqrt{2\tilde{a}_{i,j}}} + 4c_2 + 2 + c_4 - \frac{2\beta_i}{\sqrt{2\tilde{a}_{i,j}}} - \frac{2\beta_j}{\sqrt{2\tilde{a}_{i,j}}} > 0 \tag{5.48}$$

Combining (5.37) and (5.46) we get

$$\frac{a'_{i,j}}{\tilde{a}_{i,j}} + \frac{c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} - 2 + \frac{2\beta_i}{\sqrt{2\tilde{a}_{i,j}}} < 0 \tag{5.49}$$

From this we get that

$$c_{1,i} < (4\delta + \varepsilon)\sqrt{2\tilde{a}_{i,j}} \tag{5.50}$$

for large enough τ. Similarly, from (5.47) and (5.39) we get

$$c_{1,i} > -(4\delta + \varepsilon)\sqrt{2\tilde{a}_{i,j}}. \tag{5.51}$$

Finally, the constraints at $(2\mathbf{e}_{i,j}, 1), (2\mathbf{e}_{-i,j}, 1), (2\mathbf{e}_{i,-j}, 1), (2\mathbf{e}_{-i,-j}, 1)$ give us

$$2\frac{a'_{i,i}}{\tilde{a}_{i,j}} + 2\frac{a'_{j,j}}{\tilde{a}_{i,j}} + 4\frac{a'_{i,j}}{\tilde{a}_{i,j}} + \frac{2c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} + \frac{2c_{1,j}}{\sqrt{2\tilde{a}_{i,j}}} + c_2 - 1 + c_4 + \frac{4\beta_i}{\sqrt{2\tilde{a}_{i,j}}} + \frac{4\beta_j}{\sqrt{2\tilde{a}_{i,j}}} > 0 \tag{5.52}$$

$$2\frac{a'_{i,i}}{\tilde{a}_{i,j}} + 2\frac{a'_{j,j}}{\tilde{a}_{i,j}} - 4\frac{a'_{i,j}}{\tilde{a}_{i,j}} - \frac{2c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} + \frac{2c_{1,j}}{\sqrt{2\tilde{a}_{i,j}}} + c_2 - 1 + c_4 - \frac{4\beta_i}{\sqrt{2\tilde{a}_{i,j}}} + \frac{4\beta_j}{\sqrt{2\tilde{a}_{i,j}}} < 0 \tag{5.53}$$

$$2\frac{a'_{i,i}}{\tilde{a}_{i,j}} + 2\frac{a'_{j,j}}{\tilde{a}_{i,j}} - 4\frac{a'_{i,j}}{\tilde{a}_{i,j}} + \frac{2c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} - \frac{2c_{1,j}}{\sqrt{2\tilde{a}_{i,j}}} + c_2 - 1 + c_4 + \frac{4\beta_i}{\sqrt{2}\tilde{a}_{i,j}} - \frac{4\beta_j}{\sqrt{2\tilde{a}_{i,j}}} < 0 \tag{5.54}$$

$$2\frac{a'_{i,i}}{\tilde{a}_{i,j}} + 2\frac{a'_{j,j}}{\tilde{a}_{i,j}} + 4\frac{a'_{i,j}}{\tilde{a}_{i,j}} - \frac{2c_{1,i}}{\sqrt{2\tilde{a}_{i,j}}} - \frac{2c_{1,j}}{\sqrt{2\tilde{a}_{i,j}}} + c_2 - 1 + c_4 - \frac{4\beta_i}{\sqrt{2\tilde{a}_{i,j}}} - \frac{4\beta_j}{\sqrt{2\tilde{a}_{i,j}}} > 0 \tag{5.55}$$

Combining (5.52) and (5.55) we get

$$2\frac{a'_{i,i}}{\tilde{a}_{i,j}} + 2\frac{a'_{j,j}}{\tilde{a}_{i,j}} + 4\frac{a'_{i,j}}{\tilde{a}_{i,j}} + c_2 - 1 + c_4 > 0 \tag{5.56}$$

From this we get that

$$\frac{a'_{i,j}}{\tilde{a}_{i,j}} > \frac{1}{4} - \delta - \frac{\varepsilon}{4} \tag{5.57}$$

for large enough τ. Similarly, combining (5.53) and (5.54) we get

$$2\frac{a'_{i,i}}{\tilde{a}_{i,j}} + 2\frac{a'_{j,j}}{\tilde{a}_{i,j}} - 4\frac{a'_{i,j}}{\tilde{a}_{i,j}} + c_2 - 1 + c_4 < 0 \tag{5.58}$$

From this we get that

$$\frac{a'_{i,j}}{\tilde{a}_{i,j}} > -\frac{1}{4} - \delta - \frac{\varepsilon}{4} \tag{5.59}$$

for large enough τ. □

5.2. A Lower Bound for Weak Robust Learning

In this section we prove a robust lower bound that rules out the possibility of weak robust learning with $\gamma = 1$. This hardness result 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 δ robust error of 0, it is computationally hard to output a degree-2 PTF that has δ-robust error of $\varepsilon \leq \frac{1}{4}$.

Theorem 5.2.1. *[Stronger Distributional Hardness] For every $\delta > 0$ and $\varepsilon \in (0, \frac{1}{4})$, assuming $NP \neq RP$ there is no polynomial time algorithm that given a set of $N = poly(n, \frac{1}{\varepsilon})$ samples from a distribution D over $\mathbb{R}^n \times \{-1, +1\}$ can distinguish between the following two cases:*

- YES: *There exists a degree-2 PTF that has δ-robust error of 0 w.r.t. D.*
- NO: *There exists no degree-2 PTF that has δ-robust error at most ε w.r.t. D.*

In this section we prove Theorem 5.1.3, which in turns uses the non-distributional hardness in Theorem 5.2.11. But to begin with we first prove an alternate NP hardness result. Although

weaker than the hardness result of the previous section, this will help us prove the more robust bound. More formally, we will prove that

Theorem 5.2.2. *[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 δ-robust empirical error of 0 on these N points.*

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

Corollary 5.2.3. *[Distributional Hardness] For every $\delta > 0$, there exists an $\varepsilon > 0$ such that assuming $NP \neq RP$ there is no algorithm that given a set of $N = poly(n, \frac{1}{\varepsilon})$ samples from a distribution D over $\mathbb{R}^n \times \{-1, +1\}$, runs in time* $\mathrm{poly}(N)$ *and distinguishes between the following two cases:*

- YES: *There exists a degree-2 PTF that has δ-robust error of 0 w.r.t. D.*
- NO: *There exists no degree-2 PTF that has δ-robust error at most ε w.r.t. D.*

We again reduce from the QP problem (Problem $\mathcal{QP}$) which is known to be NP hard. The reduction is sketched below.

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 $sgn(z - p(x))$ is the *unique* degree-2 PTF (up to scaling) that is robust at all these $2m$ points. Note that a degree-2 PTF $sgn(q(x, z))$ on the $n + 1$ variables (x, z) may *not* necessarily be of the form $sgn(z - g(x))$

(1) Let $p(x) := x^T Ax$ be the polynomial given by Problem $\mathcal{QP}$, and let β, δ 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 $z^{(j)} = p(x^{(j)}) = (x^{(j)})^T Ax^{(j)}$ for each $j \in [m]$.
(3) Define $s^{(j)} = sgn(\nabla p(x^{(j)}))$ where the $sgn(x) \in \{-1, 1\}^n$ refers to a vector with entry-wise signs, and ∇p stands for the gradient of p at $x^{(j)}$. From each $(x^{(j)}, z^{(j)})$ generate $(u^{(j)}, z_u^{(j)}) = (x^{(j)} - \delta s^{(j)}, z^{(j)} + \delta)$ with label $y_u^{(j)} = sgn(z_u^{(j)} - p(u^{(j)}))$ and $(v^{(j)}, z_v^{(j)}) = (x^{(j)} + \delta s^{(j)}, z^{(j)} - \delta)$ with label $y_v^{(j)} = sgn(z_v^{(j)} - p(v^{(j)}))$.
(4) Generate α (depends on δ and β from problem $\mathcal{QP}$) and input the $2m + 1$ points in $\mathbb{R}^{n+1} \times \{\pm 1\}$ given by $((u^{(j)}, z_u^{(j)}), y_u^{(j)})$, $((v^{(j)}, z_v^{(j)}), y_u^{(j)})$ for each $j \in [m]$ and $(0, \alpha, +1)$ to the algorithm.

Figure 5.3. Reduction from the QP problem.

for some degree-2 polynomial $g(x)$. We need to rule out the existence of any other degree-2 PTF of the form $sgn(q(x, z))$ that is δ-robust at these points. Once we have established this, we will then show that the "intended" PTF $sgn(z - p(x))$ is δ-robust at $((0, \alpha), +1)$ in the YES case, and not δ-robust at $((0, \alpha), +1)$ in the NO case.

We proceed by first proving that the intended PTF $sgn(z-p(x))$ 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 ρ^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 $sgn(z - p(x))$ at the $2m$ added points in Lemma 5.2.6.

Lemma 5.2.4. *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*

$$\forall \ell \in [m],\ \forall i \in [n],\ \frac{|\langle A_i, x^{(\ell)} \rangle|}{\|A_i\|_1} > \delta, \tag{5.60}$$

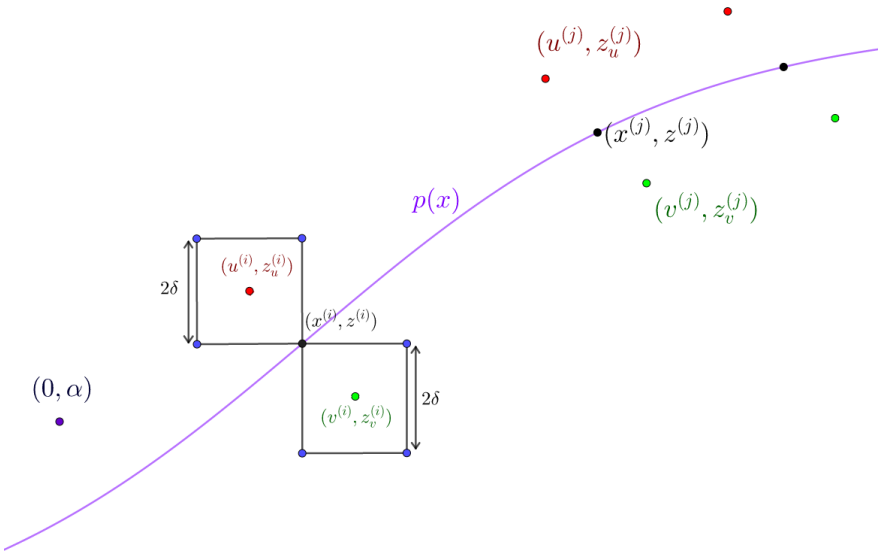

Figure 5.4. *The figure shows the construction of a hard instance for the robust learning problem. First, points* $(x^{(j)}, z^{(j)})$ *are sampled randomly and staisfying* $z^{(j)} = p(x^{(j)})$*. Each such point is then perturbed along the direction of the sign vector of the gradient at* $(x^j, z^{(j)})$ *to get two data points of the training set, one labeled as* $+1$*, and the other labeled as* -1*.*

where A_i *denotes the* i*th row of* A*.*

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

Fact 5.2.5. *Let* x^* *be sampled from* $\mathbb{N}(0, \rho^2)^n$*. Let* $a \in \mathbb{R}^n$*. There exists a universal constant* $C > 0$ *such that for any* $\eta' > 0$*,*

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

Set $\eta' := \eta/(mn)$. Fix $\ell \in [m], i \in [n]$. Using Fact 5.2.5 we have with probability at least $1 - \eta'$

$$|\langle A_i, x^{(j)} \rangle| \geq \|A_i\|_2 \rho \eta' \geq \frac{\|A_i\|_1}{\sqrt{n}} \cdot \rho \cdot \frac{\eta}{mn} \geq \delta,$$

from our assumption on ρ. The lemma follows from a union bound over all $\ell \in [m], i \in [n]$. $\square$

We now prove the δ-robustness of the "intended" degree-2 PTF $sgn(z - p(x))$ at the $2m$ added points w.h.p.

Lemma 5.2.6. *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 $sgn(z - p(x)) = sgn(z - x^T Ax)$ is δ-robust at all the $2m$ points $\{ ((u^{(\ell)}, z_u^{(\ell)}), y_u^{(\ell)}), ((v^{(\ell)}, z_v^{(\ell)}), y_v^{(\ell)}) : \ell \in [m] \}$.*

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 $s = sgn(\nabla p(x^{(\ell)})) \in \{ -1, 1 \}^n$. Hence $z^* = x^{*T} A x^*$, $(u, z_u) = (x^* - \delta s, z^* + \delta)$ and $(v, z_v) = (x^* + \delta s, z^* - \delta)$. We want to prove that the points (u, z_u) and (v, z_v) are δ robust i.e., these points are δ away in ℓ_∞ distance from the decision boundary of $sgn(z - p(x))$. 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., $z - x^T Ax > 0$.*

Note that (u, z_u) itself lies inside the ball, and hence the claim will show that $sgn(z - x^T Ax)$ is δ-robust at (u, z_u). An analogous proof also holds that δ-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

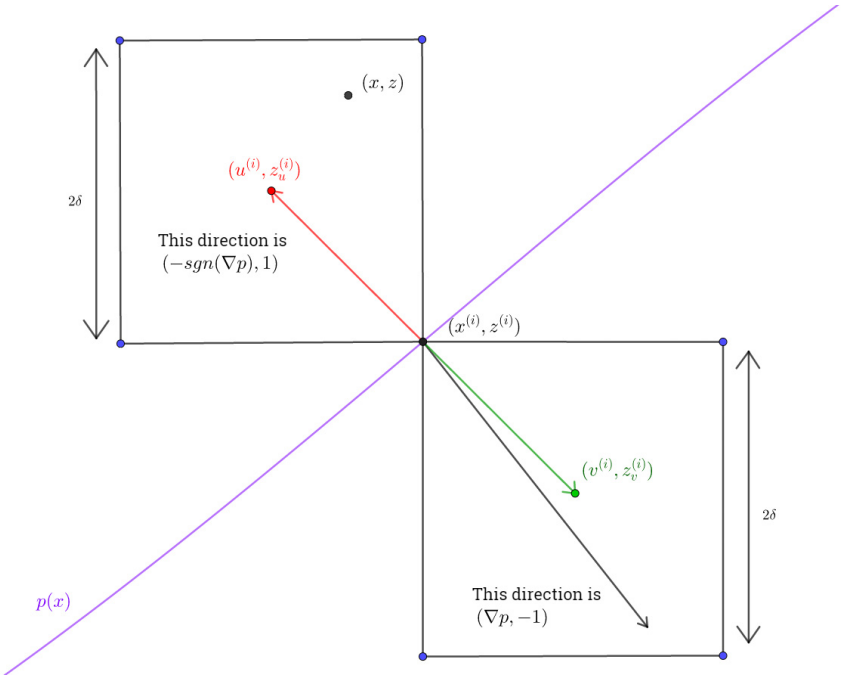

Figure 5.5. *The figure shows the radius of robustness around the point* $(x^{(i)}, z^{(i)})$. Any degree-2 PTF that is δ-robust at all the data points, must take a value of $+1$ in the upper ball around each $(x^{(i)}, z^{(i)})$ of ℓ_∞ radius of 2δ and must take a value of -1 in the lower ball around each $(x^{(i)}, z^{(i)})$ of ℓ_∞ radius of 2δ. We use this fact to establish that such a PTF must pass through the points $(x^{(i)}, z^{(i)})$.

Figure 5.5)

$$\forall j \in [d], -2\delta \leq s(j)\tilde{x}(j) \leq 0, \quad \text{and } \tilde{z} \geq 0.$$

Using $z^* = p(x^*)$ and $\tilde{z} \geq 0$, for all $(x, z) \in B^{n+1}((u, z_u), \delta)$ we have

$$\begin{aligned} z - p(x) &= z^* + \tilde{z} - p(\tilde{x} + x^*) = \tilde{z} + p(x^*) - p(\tilde{x} + x^*) = \tilde{z} - \langle \nabla p, \tilde{x} \rangle - \frac{1}{2}\tilde{x}^T \nabla^2 p \tilde{x} \\ &\geq -\sum_{i=1}^{n} \tilde{x}(i) \Big(\sum_{j=1}^{n} a_{ij} x^*(j) \Big) - \frac{1}{2} \sum_{i=1}^{n} \tilde{x}(i) \Big(\sum_{j=1}^{n} a_{ij} \tilde{x}(j) \Big) \\ &= \sum_{i=1}^{n} (-\tilde{x}(i) s(i)) \Big| \sum_{j=1}^{n} a_{ij} x^*(j) \Big| - \frac{1}{2} \sum_{i=1}^{n} \tilde{x}(i) \sum_{j=1}^{n} a_{ij} \tilde{x}(j) \\ &\geq \sum_{i=1}^{n} |\tilde{x}(i)| \Big(\Big| \sum_{j=1}^{n} a_{ij} x^*(j) \Big| - \delta \sum_{j=1}^{n} |a_{ij}| \Big), \end{aligned}$$

using the fact that $\tilde{x}(i)s(i) \in [-2\delta, 0]$ for each $i \in [n]$. Applying Lemma 5.2.4 we see that with probability at least $(1 - \eta)$, (5.60) 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 $sgn(z - p(x))$ is the only degree-2 PTF that is robust at the added $2m$ examples.

Lemma 5.2.7. *Consider any degree-2 PTF $sgn(q(x, z))$ that is δ-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 5.2.7 follows immediately from the following two lemmas (Lemma 5.2.8 and Lemma 5.2.9).

Lemma 5.2.8. *Consider any degree-2 PTF on $n + 1$ variables $sgn(q(x, z))$ that satisfies the conditions of Lemma 5.2.7. Then $q(x^{(\ell)}, z^{(\ell)}) = 0$ for each $\ell \in [m]$.*

Proof. Since $sgn(q(u^{(\ell)}, z_u^{(\ell)})) \neq sgn(q(v^{(\ell)}, z_v^{(\ell)}))$, by the Intermediate Value Theorem,

$$\exists \gamma \in [0,1] \text{ s.t. } (\widehat{x}, \widehat{z}) = \gamma(u^{(\ell)}, z_u^{(\ell)}) + (1-\gamma)(v^{(\ell)}, z_v^{(\ell)}) \text{ and } q(\widehat{x}, \widehat{z}) = 0.$$

Also, since q is δ-robust at $(u^{(\ell)}, z_u^{(\ell)})$ and $(v^{(\ell)}, z_v^{(\ell)})$, we must have that $(\widehat{x}, \widehat{y})$ is atleast δ far away in ℓ_∞ distance from both $(u^{(\ell)}, z_u^{(\ell)})$ and $(v^{(\ell)}, z_v^{(\ell)})$. Further by design two points are separated by exactly 2δ in each co-ordinate (see Figure 5.5 for an illustration)! Hence it is easy to see that $\gamma = 1/2$ i.e., $(\widehat{x}, \widehat{z}) = (x^{(\ell)}, z^{(\ell)})$ 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 5.2.8 this establishes the proof of Lemma 5.2.7.

Lemma 5.2.9. *Let $m > (n+1)^2$ and let $q : \mathbb{R}^{n+1} \to \mathbb{R}$ be any degree-2 polynomial with $q(x^{(\ell)}, z^{(\ell)}) = 0$ for all $\ell \in [m]$, where $z^{(\ell)} = (x^{(\ell)})^T A^* x^{(\ell)}$ and $x^{(\ell)} \sim N(0, \rho^2)^n$ 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

$$q(x, z) = x^T A x + b_1^T x + c_1 + z b_2^T x + c_2 z^2 + c_3 z, \text{ where } x \in \mathbb{R}^n, z \in \mathbb{R}.$$

This polynomial is parameterized by a vector $w = (A, b_1, c_1, b_2, c_2, c_3) \in \mathbb{R}^r$ where $r = \binom{n+1}{2} + 2n + 3$ (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

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

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

where $u \oplus v$ refers to the concatenation of vectors u and v, and M_ℓ represents the row ℓ 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 jth component of x and $z = x^T A^* x$. Observe that the "intended" polynomial $q^*(x,z) = z - x^T A^* x$ 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 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 $\{ x_i x_j : i \leq j \}$ since $z := x^T A^* x$ 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.,

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

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

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 ℓ 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 $S(R_\ell)^\perp$.

$$\langle v, M'_{\ell+1} \rangle = \widehat{q}(x^{(\ell+1)}, z^{(\ell+1)}), \text{ where } \widehat{q}(x,z) := \langle v, g(x,z) \rangle$$

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 5.2.10), we get that $\widehat{q}(x^{(\ell+1)}, z^{(\ell+1)}) \neq 0$ 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 5.2.10. *A non-zero multivariate polynomial $p : \mathbb{R}^n \to \mathbb{R}$ 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 5.2.9 can also be made robust to inverse polynomial error by using polynomial anti-concentration bounds (e.g., Carbery-Wright inequality) instead of Fact 5.2.10; however this is not required for proving NP-hardness. We now complete the proof of Theorem 5.2.2.

PROOF OF THEOREM 5.2.2. We start with the NP-hardness of $\mathcal{QP}$, and for the reduction in Figure 5.3, we will show that in the YES case, we will show that there is a δ-robust degree-2 PTF (completeness), and in the NO case we will show that there is no δ robust degree-2 PTF (soundness). As a reminder, the NP-hard problem $\mathcal{QP}$ 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 $q(y^*) = (y^*)^T A y^* > \beta$,

YES Case : $\max_{\|y\|_\infty \leq 1} y^T A y < \beta$.

Completeness (**YES Case**): Consider the degree-2 PTF given by $sgn(z - p(x)) = sgn(z - x^T Ax)$. From Lemma 5.2.6, we have that it is δ 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 η being any sufficiently small constant). Further, from multilinearity of p we have that,

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

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

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

Soundness (**NO Case**): From Lemma 5.2.7, we see that the degree-2 PTF given by $sgn(z - p(x)) = sgn(z - x^T Ax)$ 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,

$$\max_{\|y\|_\infty \le \delta} y^T A y = \delta^2 \max_{\|y\|_\infty \le 1} y^T A y > \delta^2 \beta = \alpha - \delta.$$

$$\text{Hence } (\alpha - \delta) - \max_{\|y\|_\infty \le \delta} y^T A y < 0,$$

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

□

Stronger Hardness. We now prove the robust lower bound stated below.

Theorem 5.2.11. *[Stronger Hardness] For every $\delta > 0$ and $\varepsilon \in (0, \frac{2}{7})$, 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 δ-robust empirical error of 0, can output a degree-2 PTF that has δ-robust empirical error of at most ε on these N points.*

Proof. The proof of this theorem closely follows the proof of Theorem 5.2.2 (the $\varepsilon = 0$ case), so we only point out the differences here. The reduction uses the same gadget (Figure 5.3) used in Theorem 5.2.2. The main challenge is the soundness analysis (NO case), where we need to rule out the existence of degree-2 PTFs which are δ-robust and consistent at all but an ε fraction of the points. To handle this, we introduce "redundancy" by including more points (of both kinds) to ensure that even when an arbitrary ε fraction of these points

are ignored (the PTF makes errors on them), we can still use the arguments in the soundness analysis of Theorem 5.2.2.

Recall that our reduction (see Figure 5.3) 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 $\{ (u^{(\ell)}, z_u^{(\ell)}) : \ell \in [N_2/2] \}$ which are generated as described in Figure 5.3 after drawing $x^{(\ell)} \sim N(0, \rho^2)^n$ for $\ell \in [N_2/2]$ (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 5.2.2, as $sgn(z - p(x))$ will be δ-robust at all of the N points (from Lemma 5.2.6 and our choice of α).

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

$$N_1 > \varepsilon(N_1 + N_2) \tag{5.61}$$

$$(1 - \varepsilon)(N_1 + N_2) > N_1 + \frac{N_2}{2} + (n+1)^2 \tag{5.62}$$

From (5.62) 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 (5.62) represents a lower bound on the number of points the candidate degree-2 PTF is

robust on. The RHS of (5.62) represents the number of points needed to ensure that atleast $(n+1)^2$ pairs of points from Type B are picked. Hence using Lemma 5.2.7 along with a union bound over all the $\binom{N_2}{(n+1)^2}$ choices of the pairs (note that the failure probability in Lemma 5.2.9 is 0), the "intended" PTF $sgn(z - p(x))$ is the only surviving degree-2 PTF.

Again from (5.61) and the pigeonhole principle, any $(1-\varepsilon)$ fraction of the points will contain *atleast one* point of the Type A i.e., $(0, \alpha)$. Hence in the NO case, the "intended" PTF $sgn(z - p(x))$ is not δ-robust. This completes the soundness analysis and establishes the theorem.

□

CHAPTER 6

Experiments

6.1. Stable Clustering

We evaluate Algorithm 3.3.1 on multiple real world datasets and compare its performance to the performance of k-means++, and also check how well these datasets satisfy our geometric conditions.

Table 6.1. Comparison of k-means cost for Alg 3.3.1 and k-means++

Dataset	Alg 3.3.1	k-means++	Alg 3.3.1 with Lloyd's	k-means++ with Lloyd's
Wine	2.376e+06	2.426e+06	2.371e+06	2.371e+06
Wine (normalized)	48.99	65.50	48.99	48.95
Iris	81.04	86.45	78.95	78.94
Iris (normalized)	7.035	7.676	6.998	6.998
Banknote Auth.	44808.9	49959.9	44049.4	44049.4
Banknote (norm.)	138.4	155.7	138.1	138.1
Letter Recognition	744707	921643	629407	611268
Letter Rec. (norm.)	3367.8	4092.1	2767.5	2742.3

Table 6.2. Values of ε satisfying Lemma 3.1.5

Dataset	Minimum ε	Average ε	Maximum ε
Wine	0.0115	0.0731	0.191
Wine (normalized)	0.000119	0.0394	0.107
Iris	0.00638	0.103	0.256
Iris (normalized)	0.00563	0.126	0.237
Banknote Auth.	0.00127	0.00127	0.00127
Banknote (norm.)	0.00175	0.00175	0.00175
Letter Recognition	3.22e-05	0.0593	0.239
Letter Rec. (norm.)	8.49e-06	0.0564	0.247

Table 6.3. Values of $(\rho, \varepsilon, \Delta)$ satisfied by $(1-\eta)$-fraction of points

Dataset	η	ε	minimum ρ/Δ	average ρ/Δ	maximum ρ/Δ
Wine	0.05	0.1	0.355	0.992	2.19
		0.01	0.374	1	2.2
	0.1	0.1	0.566	1.5	3.05
		0.01	0.609	1.53	3.07
Wine (normalized)	0.05	0.1			
		0.01	0.399	1.06	2.29
	0.1	0.1	0.451	1.3	2.66
		0.01	0.735	1.96	3.62
Iris	0.05	0.1	0.156	2.47	5.37
		0.01	0.263	2.88	6.43
	0.1	0.1	0.398	4.35	7.7
		0.01	0.496	5.04	9.06
Iris (normalized)	0.05	0.1	0.0918	1.89	3.08
		0.01	0.213	2.21	3.4
	0.1	0.1	0.223	3.74	7.12
		0.01	0.391	4.42	8.3
Banknote Auth.	0.05	0.1	0.0731	0.0731	0.0731
		0.01	0.198	0.198	0.198
	0.1	0.1	0.264	0.264	0.264
		0.01	0.398	0.398	0.398
Banknote (norm.)	0.05	0.1			
		0.01	0.197	0.197	0.197
	0.1	0.1	0.246	0.246	0.246
		0.01	0.474	0.474	0.474
Letter Recognition	0.05	0.1			
		0.01	0.168	2.06	6.96
	0.1	0.1	0.018	2.19	7.11
		0.01	0.378	3.07	11.4
Letter Rec. (norm.)	0.05	0.1			
		0.01	0.157	1.97	7.14
	0.1	0.1			
		0.01	0.378	2.92	11.2

Datasets : Experiments were run on unnormalized and normalized versions of four labeled datasets from the UCI Machine Learning Repository: Wine ($n = 178$, $k = 3$, $d = 13$),

Iris ($n = 150$, $k = 3$, $d = 4$), Banknote Authentication ($n = 1372$, $k = 2$, $d = 5$), and Letter Recognition ($n = 20,000$, $k = 26$, $d = 16$). Normalization was used to scale each feature to unit range.

Performance : The cost of the solution returned by Algorithm 3.3.1 for each of the normalized and unnormalized versions of the datasets is recorded in Table 6.1 column 2. Our guarantees show that under $(\rho, \Delta, \varepsilon)$-separation for appropriate values of ρ (see section 3.3), the algorithm will find the optimal clustering after a single iteration of Lloyd's algorithm. Even when ρ does not satisfy our requirement, we can use our algorithm as an initialization heuristic for Lloyd's algorithm. We compare our initialization with the k-means++ initialization heuristic (D^2 weighting). In Table 6.1, this is compared to the smallest initialization cost of 1000 trials of k-means++ on each of the datasets, the solution found by Lloyd's algorithm using our initialization and the smallest k-means cost of 100 trials of Lloyd's algorithm using a k-mean++ initialization.

Separation in real data sets : As the ground truth clusterings in our datasets are not in general linearly separable, we consider the clusters given by Lloyd's algorithm initialized with the ground truth solutions.

Values of ε for Lemma 3.1.5. We calculate the maximum value of ε such that every pair of clusters satisfies the angular and margin separations implied by ε-APS (Lemma 3.1.5). The results are recorded in Table 6.2. We see that the average value of ε lies approximately in the range $(0.01, 0.1)$.

Values of $(\rho, \Delta, \varepsilon)$-separation. We attempt to measure the values of ρ, Δ, and ε in the datasets. For $\eta = 0.05, 0.1$, $\varepsilon = 0.1, 0.01$, and a pair of clusters C_i, C_j, we calculate ρ as the maximum margin separation a pair of axis-aligned cones with half-angle $\arctan(1/\varepsilon)$ can

have while capturing a $(1-\eta)$-fraction of all points. For some datasets and values for η and ε, there may not be any such value of ρ, in this case we leave the corresponding entry blank. These results are collected in Table 6.3.

Ground truth recovery : The clustering returned by our algorithm recovers well ($\approx 97\%$) the solution returned by Lloyd's algorithm initialized with the ground truth for Wine, Iris, and Banknote Authentication across normalized and unnormalized datasets.

6.2. Adversarial Learning

In this section, we evaluate the performance of the SDP based rounding algorithm outlined in Figure 4.3 to generate adversarial examples for depth-2 neural networks with ReLU gates, and compare it with the projected gradient descent(PGD) based attack of Madry et al. Madry et al. [2017]. We will show that our approach indeed finds more adversarial examples. This however, comes at a computational cost since we need to solve one SDP per example and per pair of classes. We use the MNIST data set and our two layer neural network has $d = 784$ input units, $k = 1024$ hidden units and 10 output units. This leads to an SDP with $d + k + 1$ vector variables. On an standard desktop with Intel i5 4590 processor, and 4 cores 3.30GHz, solving one SDP instance takes 200 seconds on average. As a consequence we perform our experiments on randomly chosen subsets of the MNIST data set. Another optimization we perform for computational reasons is that given an example x with predicted class i, rather than checking for every class j, if one can find an attack example z that misclassifies $x + z$ to be in class j, we simply pick j to be the class label of the second highest prediction at x. Hence, the numbers we report below are an underestimate of the effectiveness of the full SDP based algorithm

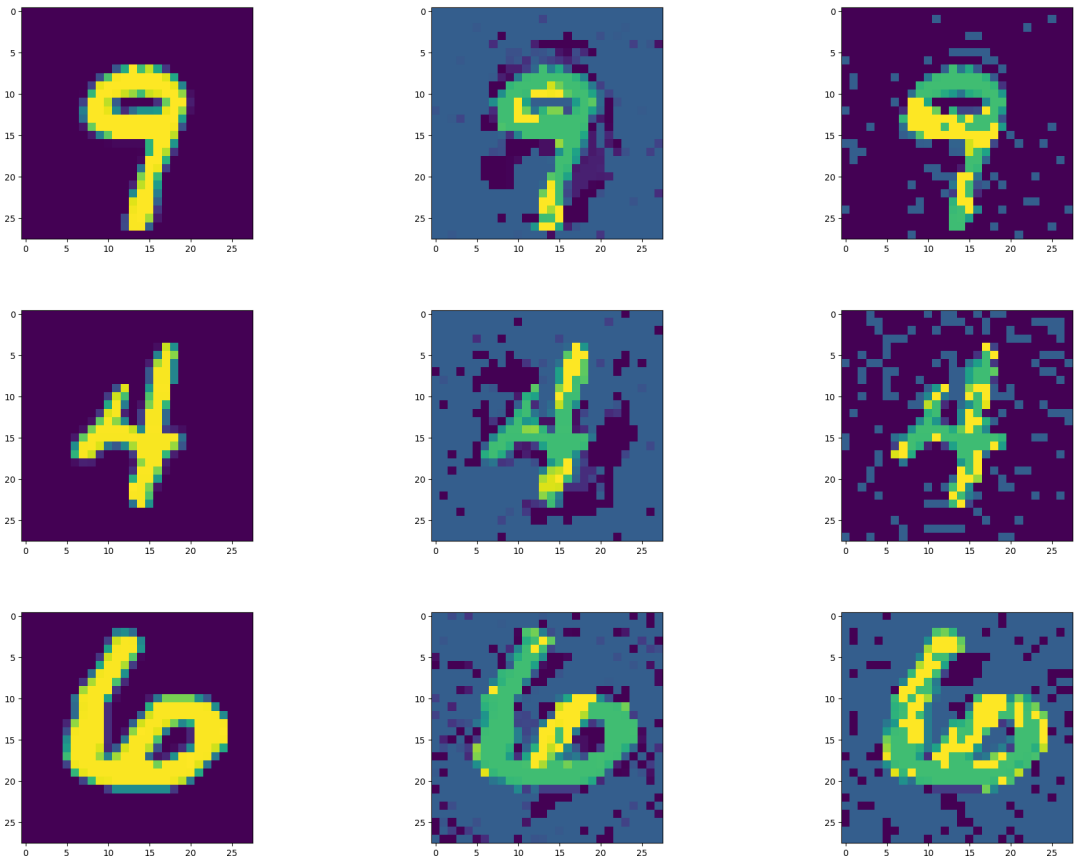

Figure 6.1. The figure shows three MNIST random samples from PDGfail (i.e., examples where PGDattack failed to find an adversarial perturbation), where SDPattack successfully finds adversarial perturbations for $\delta = 0.3$. The images in the first column represent the original images corresponding to three, the second column represents the perturbed images produced by the failed PGDattack, and perturbed images produced by the successful SDPattack. Visual inspection of these examples suggest that our method often produces sparse targeted perturbations.

We compare the effectiveness of our attack in finding adversarial examples when compared to the the PGD based attack of Madry et al. Madry et al. [2017]. We consider two settings of the parameter δ, the maximum amount by which each pixel can be perturbed to produce a valid attack example. As in Madry et al. [2017] we first choose $\delta = 0.3$ and train a robust

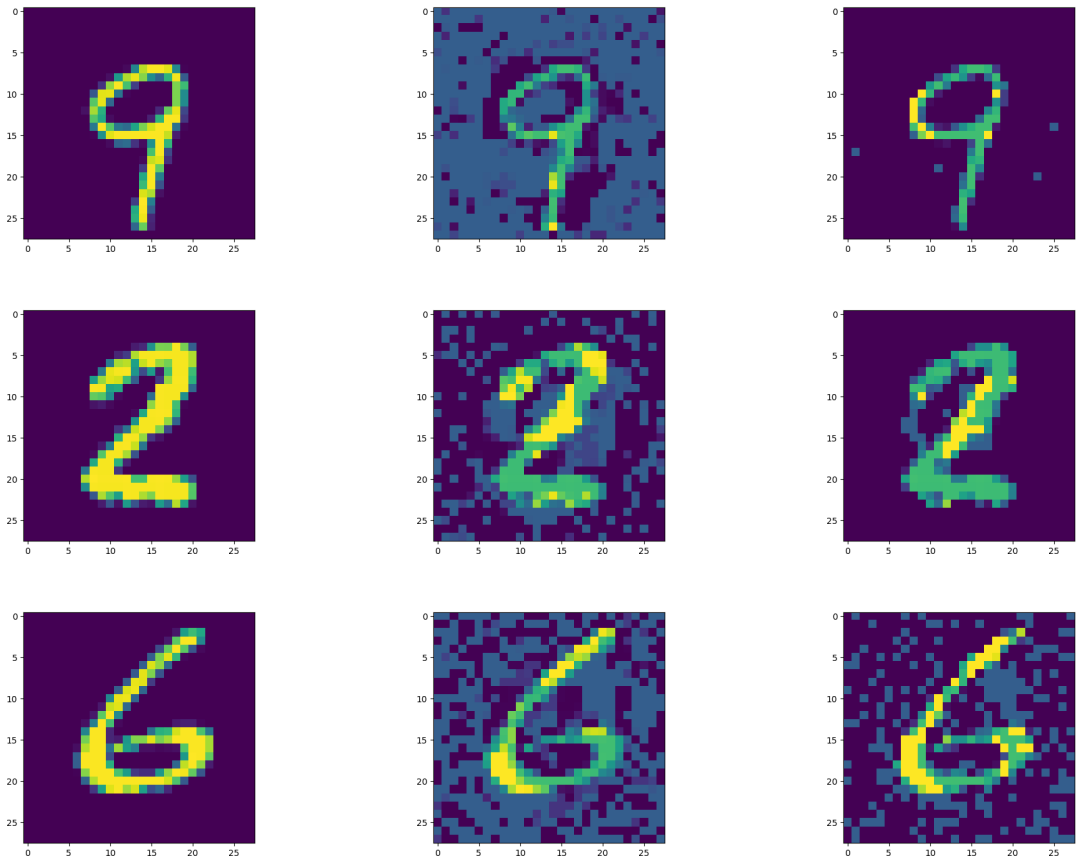

Figure 6.2. The figure shows three MNIST random samples from PDGpass (i.e., examples where PGDattack succeeded to find an adversarial perturbation), where SDPattack successfully finds adversarial perturbations for $\delta = 0.3$. The images in the first column represent the original images corresponding to three, the second column represents the perturbed images produced by the successful PGDattack, and perturbed images produced by the successful SDPattack. Visual inspection of these examples suggest that our method often produces sparse targeted perturbations.

2-layer network using the algorithm of Madry et al. Madry et al. [2017]. We then run the PGD attack and divide the test set into examples where the PGD attack succeeds (PGDPass) and examples where the PGD attack fails (PGDfail). We then run our attack on batches of random subsets chosen from each set. In the algorithm we set $\delta' = \alpha\delta$ for a hyperparameter

$\alpha \leq 1$. This is because we want to ensure that the rounded solutions have ℓ_∞ norm of at most δ. In our experiment we set $\alpha = 0.07$. The first row of Table 6.4 shows the precision and recall of our method. We report the average and the standard deviation across the chosen batches. As one can see, our method has very high recall, i.e., whenever the PGD attack succeeds, our SDP based algorithm also finds adversarial examples. Furthermore, on examples where the PGD attack fails, our method is still able to discover new adversarial examples 30% of the time. Please see Figure 6.1 for the images corresponding to some of the examples where the SDP based attack succeeds, but the PGDattack fails and Figure 6.2 for the images of some examples where both the PGDattack and SDP based attack succeed. A visual inspection of both the figures reveals that our attack often produces sparse targeted attacks as opposed to PGDattack.

We repeat the same methodology with $\delta = 0.01, \alpha = 0.2$. Here we notice that PGD attack succeeds on only 138 test examples and hence we can afford to run our attack on all of them. As can be seen from the second row of Table 6.4 our attack succeeds on all of these examples. Furthermore, we rank the examples in PGDfail according to the difference of the highest and the second highest of the ten network outputs. The smaller the difference, the easier it should be to find an adversarial example. Indeed as can be seen from the table, our method finds 45 new adversarial examples out of the first 100 such ranked examples.

The experiments above suggest that our algorithms can lead to improved adversarial attacks. We would like to note that the recent work of Raghunathan et al. [2018] also studied semi-definite programming based methods for providing adversarial certificates for 2-layer neural networks. However, our SDP as outlined is Figure 4.3 is strictly stronger. In particular, the SDP of Raghunathan et al. [2018] is independent of the given example x and as a result

$\delta = 0.3$	PGDpass (6×50 random samples)	PGDfail (8×100 random samples)
SDP succeds	297 out of 300 total Mean : 49.5 of 50, Std : 0.76	244 out of 800 total Mean 30.6 of 100, Std : 2.87
$\delta = 0.01$	PGDpass (138 samples)	PGDfail (100 ranked)
SDP succeeds	138	45

Table 6.4. For $\delta = 0.3$, we report mean and standard deviation of number of adversarial examples found by running our SDPattack algorithm on 6 batches of 50 random examples from PGDpass and 8 batches of 100 random samples from PGDfail. For $\delta = 0.01$, we run SDPattack on all 138 examples in PGDpass and first 100 sorted examples from PGDfail.

we expect our method to produce better certificates. We leave as future work the task of making our theoretical analysis practical for large scale applications.

CHAPTER 7

Open Problems

In this thesis we try to better understand the connection between resilience to adversarial perturbations and how this property makes NP-hard problems tractable in both the supervised and unsupervised setting. We study this in the context of clustering in the unsupervised setting and adversarial learning of PTFs in the supervised setting. The resilience to adversarial perturbations comes up in two different forms in the problems as do the margin conditions they imply. There are two kinds of further work we propose in this thesis. The first is further work on these specific two problems, while the second is exploring this perhaps surprising connection further.

7.1. Lower Bounds for ε Additive Stable k-means Instances

In our clustering work, we gave algorithms for general Euclidean k-means as well as robust k-means. Proving lower bounds on additive stable instances of euclidean k-means however remains an open question.

Question 7.1.1. For some "non-trivial" ε is it NP hard to find the optimal clustering in ε additive stable instances of euclidean k-means?

It might be easy to show hardness when ε is exponentially small. However, we believe the answer to the question is yes for a non-trivial value of ε as well. We also think think that reductions like [Vattani] where the author shows hardness of k-means clustering in the plane,

is key in proving hardness of ε stable instances. The main idea would be to use a similar reduction to reduce to a final instance that is ε stable.

7.2. Further Directions in Adversarial Learning

Several questions still remain as theoretical study in this field is still in its infancy. A straightforward follow up to our work is to investigate whether our framework can be used to design robust algorithms for degree d PTFs.

Question 7.2.1. Can the framework discussed in Section 4.2 be used to design algorithms for degree d PTFs that are robust to adversarial perturbations?

While there are algorithms that approximately maximize degree d polynomials, they only work for the homogeneous case which do not suffice for our purpose. It is also an open problem whether our adversarial attack for depth-2 neural networks can be converted into a robust learning algorithm via the same framework. The most straightforward use of the framework does not lead to a convex constraint set.

Currently, the bottleneck in our work is the speed of the SDP solution. If the SDP solving step can be sped up significantly, then we can make our SDP attack work on a much larger scale leading to improved adversarial attacks.

Question 7.2.2. Can the SDP attack be sped up to make it work on a larger scale?

If the answer to the above question is yes, then it will be possible to run our SDP based adversarial attack on real world scenarios. This will produce better adversarial examples which in turn will help train algorithms that are very robust to adversaries.

7.3. Adversarial Resilience and tractability of NP-hard problems

Finally, this thesis is an initial exploration into the connection between an instance being tractable and adversarially robust. Further work that explores this connection should be very interesting.

Question 7.3.1. What instances of NP-hard problems become easy if they have a solution that is adversarially robust?

Recent work have already started to address this question. For instance [Montasser et al., 2020] efficiently learns noisy halfspaces that are adversarially robust, while [Dan et al., 2020] gives statistical guarantees for adversarially robust Gaussian classification. Another line of very interesting work analyzes adversarially robust principal component analysis [Awasthi et al., 2019a, 2020].

References

Margareta Ackerman and Shai Ben-David. Clusterability: A theoretical study. In David van Dyk and Max Welling, editors, *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pages 1–8, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 16–18 Apr 2009. PMLR. URL `http://proceedings.mlr.press/v5/ackerman09a.html`.

Noga Alon, Konstantin Makarychev, Yury Makarychev, and Assaf Naor. Quadratic forms on graphs. *Inventiones mathematicae*, 163(3):499–522, 2006.

Haris Angelidakis, Konstantin Makarychev, and Yury Makarychev. Algorithms for stable and perturbation-resilient problems. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, page 438–451, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450345286. doi: 10.1145/3055399.3055487. URL `https://doi.org/10.1145/3055399.3055487`.

Sanjeev Arora, Eli Berger, Hazan Elad, Guy Kindler, and Muli Safra. On non-approximability for quadratic programs. In *Foundations of Computer Science, 2005. FOCS 2005. 46th Annual IEEE Symposium on*, pages 206–215. IEEE, 2005.

Idan Attias, Aryeh Kontorovich, and Yishay Mansour. Improved generalization bounds for robust learning. *arXiv preprint arXiv:1810.02180*, 2018.

Pranjal Awasthi and Or Sheffet. Improved spectral-norm bounds for clustering. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 37–49. Springer, 2012.

Pranjal Awasthi, Avrim Blum, and Or Sheffet. Improved guarantees for agnostic learning of disjunctions. In Adam Tauman Kalai and Mehryar Mohri, editors, *COLT*, pages 359–367. Omnipress, 2010. ISBN 978-0-9822529-2-5. URL `http://dblp.uni-trier.de/db/conf/colt/colt2010.html#AwasthiBS10`.

Pranjal Awasthi, Avrim Blum, and Or Sheffet. Center-based clustering under perturbation stability. *Information Processing Letters*, 112(1–2):49 – 54, 2012. ISSN 0020-0190. doi: http://dx.doi.org/10.1016/j.ipl.2011.10.006. URL `http://www.sciencedirect.com/science/article/pii/S0020019011002778`.

Pranjal Awasthi, Afonso S. Bandeira, Moses Charikar, Ravishankar Krishnaswamy, Soledad Villar, and Rachel Ward. Relax, no need to round: integrality of clustering formulations, 2014.

Pranjal Awasthi, Vaggos Chatziafratis, Xue Chen, and Aravindan Vijayaraghavan. Adversarially robust low dimensional representations, 2019a.

Pranjal Awasthi, Abhratanu Dutta, and Aravindan Vijayaraghavan. On robustness to adversarial examples and polynomial optimization, 2019b.

Pranjal Awasthi, Xue Chen, and Aravindan Vijayaraghavan. Estimating principal components under adversarial perturbations, 2020.

Maria Florina Balcan and Yingyu Liang. Clustering under perturbation resilience, 2011.

Shai Ben-David. Computational feasibility of clustering under clusterability assumptions, 2015.

Aharon Ben-Tal and Arkadi Nemirovski. Robust solutions of uncertain linear programs. *Operations research letters*, 25(1):1–13, 1999.

Dimitris Bertsimas and Melvyn Sim. The price of robustness. *Operations research*, 52(1): 35–53, 2004.

Chiranjib Bhattacharyya. Robust classification of noisy data using second order cone programming approach. In *Intelligent Sensing and Information Processing, 2004. Proceedings of International Conference on*, pages 433–438. IEEE, 2004.

Alberto Bietti, Grégoire Mialon, and Julien Mairal. On regularization and robustness of deep neural networks. *arXiv preprint arXiv:1810.00363*, 2018.

Yonatan Bilu and Nathan Linial. Are stable instances easy? In *ICS'10*, pages 332–341, 2010.

Hans-Dieter Block. The perceptron: A model for brain functioning. *Reviews of Modern Physics*, 34(1):123–135, 1962.

Avrim Blum and John Dunagan. Smoothed analysis of the perceptron algorithm for linear programming. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '02, page 905–914, USA, 2002. Society for Industrial and Applied Mathematics. ISBN 089871513X.

Sébastien Bubeck, Yin Tat Lee, Eric Price, and Ilya Razenshteyn. Adversarial examples from cryptographic pseudo-random generators. *arXiv preprint arXiv:1811.06418*, 2018a.

Sébastien Bubeck, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. *arXiv preprint arXiv:1805.10204*, 2018b.

M Charikar and A Wirth. Maximizing quadratic programs: extending grothendieck's inequality. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 54–60. IEEE, 2004.

Daniel Cullina, Arjun Nitin Bhagoji, and Prateek Mittal. Pac-learning in the presence of evasion adversaries. *arXiv preprint arXiv:1806.01471*, 2018.

Chen Dan, Yuting Wei, and Pradeep Ravikumar. Sharp statistical guarantees for adversarially robust gaussian classification, 2020.

Sanjoy Dasgupta. *The hardness of k-means clustering.* Department of Computer Science and Engineering, University of California, San Diego, 2008.

Dimitrios Diochnos, Saeed Mahloujifar, and Mohammad Mahmoody. Adversarial risk and robustness: General definitions and implications for the uniform distribution. In *Advances in Neural Information Processing Systems*, pages 10380–10389, 2018.

Abhratanu Dutta, Aravindan Vijayaraghavan, and Alex Wang. Clustering stable instances of euclidean k-means, 2017.

Laurent El Ghaoui and Hervé Lebret. Robust solutions to least-squares problems with uncertain data. *SIAM Journal on matrix analysis and applications*, 18(4):1035–1064, 1997.

Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. Robustness of classifiers: from adversarial to random noise. In *Advances in Neural Information Processing Systems*, pages 1632–1640, 2016.

Uriel Feige, Yishay Mansour, and Robert Schapire. Learning and inference in the presence of corrupted inputs. In *Conference on Learning Theory*, pages 637–657, 2015.

Zachary Friggstad, Kamyar Khodamoradi, and Mohammad R. Salavatipour. Exact algorithms and lower bounds for stable instances of euclidean k-means. 2018.

Michael R Garey and David S Johnson. *Computers and intractability*, volume 29. wh freeman New York, 2002.

Justin Gilmer, Ryan P Adams, Ian Goodfellow, David Andersen, and George E Dahl. Motivating the rules of the game for adversarial example research. *arXiv preprint arXiv:1807.06732*, 2018a.

Justin Gilmer, Luke Metz, Fartash Faghri, Samuel S Schoenholz, Maithra Raghu, Martin Wattenberg, and Ian Goodfellow. Adversarial spheres. *arXiv preprint arXiv:1801.02774*, 2018b.

Amir Globerson and Sam Roweis. Nightmare at test time: robust learning by feature deletion. In *Proceedings of the 23rd international conference on Machine learning*, pages 353–360. ACM, 2006.

Pascale Gourdeau, Varun Kanade, Marta Kwiatkowska, and James Worrell. On the hardness of robust classification. *arXiv preprint arXiv:1909.05822*, 2019.

Michael J Kearns, Umesh Virkumar Vazirani, and Umesh Vazirani. *An introduction to computational learning theory*. MIT press, 1994.

Justin Khim and Po-Ling Loh. Adversarial risk bounds for binary classification via function transformation. *arXiv preprint arXiv:1810.09519*, 2018.

Subhash Khot and Ryan O'Donnell. Sdp gaps and ugc-hardness for maxcutgain. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 217–226. IEEE, 2006.

Amit Kumar and Ravindran Kannan. Clustering with spectral norm and the k-means algorithm. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 299–308. IEEE, 2010.

Hunter Lang, David Sontag, and Aravindan Vijayaraghavan. Block stability for map inference, 2018.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Saeed Mahloujifar and Mohammad Mahmoody. Can adversarially robust learning leverage computational hardness? *arXiv preprint arXiv:1810.01407*, 2018.

Saeed Mahloujifar, Dimitrios I Diochnos, and Mohammad Mahmoody. The curse of concentration in robust learning: Evasion and poisoning attacks from concentration of measure. *arXiv preprint arXiv:1809.03063*, 2018.

Konstantin Makarychev and Yury Makarychev. Metric perturbation resilience, 2016.

Konstantin Makarychev, Yury Makarychev, and Aravindan Vijayaraghavan. Bilu-linial stable instances of max cut and minimum multiway cut, 2013.

D. G. Mixon, S. Villar, and R. Ward. Clustering subgaussian mixtures with k-means. In *2016 IEEE Information Theory Workshop (ITW)*, pages 211–215, Sept 2016. doi: 10.1109/ITW.2016.7606826.

Omar Montasser, Surbhi Goel, Ilias Diakonikolas, and Nathan Srebro. Efficiently learning adversarially robust halfspaces with noise, 2020.

Yu Nesterov. Semidefinite relaxation and nonconvex quadratic optimization. *Optimization methods and software*, 9(1-3):141–160, 1998.

Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, New York, NY, USA, 2014. ISBN 1107038324, 9781107038325.

Orr Paradise. Smooth and Strong PCPs. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*, volume 151 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:41, Dagstuhl, Germany, 2020. Schloss

Dagstuhl–Leibniz-Zentrum fuer Informatik. ISBN 978-3-95977-134-4. doi: 10.4230/LIPIcs.ITCS.2020.2. URL https://drops.dagstuhl.de/opus/volltexte/2020/11687.

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

Tim Roughgarden. Beyond worst-case analysis, 2018.

Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. *arXiv preprint arXiv:1804.11285*, 2018.

Pannagadatta K Shivaswamy, Chiranjib Bhattacharyya, and Alexander J Smola. Second order cone programming approaches for handling missing and uncertain data. *Journal of Machine Learning Research*, 7(Jul):1283–1314, 2006.

Aman Sinha, Hongseok Namkoong, Riccardo Volpi, and John Duchi. Certifying some distributional robustness with principled adversarial training, 2017.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy, 2018.

Andrea Vattani. The hardness of k-means clustering in the plane.

Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5283–5292, 2018.

Huan Xu and Shie Mannor. Robustness and generalization. *Machine learning*, 86(3):391–423, 2012.

Huan Xu, Constantine Caramanis, and Shie Mannor. Robustness and regularization of support vector machines. *Journal of Machine Learning Research*, 10(Jul):1485–1510, 2009.

Dong Yin, Kannan Ramchandran, and Peter Bartlett. Rademacher complexity for adversarially robust generalization. *arXiv preprint arXiv:1810.11914*, 2018.