Understanding the Convergence of Adversarial Training for Overparameterized Linear Neural Networks

by

Darshan Bhavin Thaker

A thesis submitted in partial satisfaction of the
requirements for the degree of
Master of Science in Computer Science

at

Columbia University

Committee in charge:

Professor John Wright
Professor Daniel Hsu
Professor Alexandr Andoni

Understanding the Convergence of Adversarial Training for Overparameterized Linear Neural Networks

Abstract

The goal of this thesis is to understand the convergence of adversarial training for overparametrized neural networks. Adversarial training can be formalized as a min-max optimization game, where one player attempts to minimize a loss function, while an adversary gets to perturb inputs to maximize the same loss function. With the recent wave of results that demonstrate the linear behavior of training infinite-width networks with gradient descent, we build upon these results that explain some of the empirical successes of adversarial training. A recent result by Gao et al. uses arguments that follow Neural Tangent Kernel (NTK) theory to demonstrate the convergence of adversarial training under general adversaries assuming that the adversary can solve the inner maximization to optimality. This work is concerned with justifying that assumption, proving that in some special cases when we use a projected gradient ascent adversary, we can solve the inner maximization to optimality for linear neural networks. We find that running projected gradient ascent initialized at the center of the constraint ball provably yields a global maxima of the inner maximization problem, and this fixed initialization is preferred to running randomly initialized projected gradient ascent. If we run randomly initialized projected gradient ascent, finding a global maximum of the inner maximization problem becomes tougher over the course of adversarial training. We then extend to the non-linear network case and provide some empirical evidence on the positive effect of overparameterization on the optimization landscape of the inner maximization problem.

Contents

Acknowledgments

I would like to thank Prof. John Wright for advising me over the last year on this project. His guidance and advice has been crucial in helping me reason about different problems, gain clarity on how to break down a research question, and make steady progress. I am also very grateful to Sam Buchanan for being so generous with his time, helping me with some of the proofs in this thesis, and helping me articulate my thoughts better. Finally, I would like to thank Prof. Daniel Hsu and Prof. Alexandr Andoni for being on my committee.

Chapter 1

Introduction

While the field of deep learning has achieved large empirical successes, theoretically underpinning the workings of neural networks has been an open problem for many years. In particular, it has been observed that functions learned by neural networks are not stable against small input perturbations, and predictions can wildly change even when inputs are perturbed slightly [15]. This motivates the research question of understanding the existence of *adversarial examples* for neural networks and how we can develop algorithms to avoid adversarial examples. Figure 1.1 highlights some cases of adversarial examples that can fool neural networks trained for an image recognition task. To the human eye, these perturbations are not visually discernible, however they can fool a neural network to incorrectly classify all perturbed images.

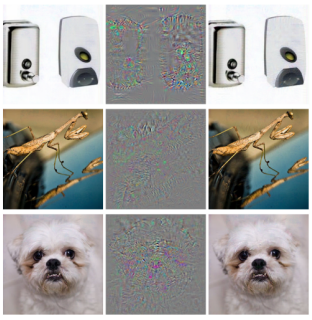


Figure 1.1: Adversarial examples for AlexNet trained on the ImageNet dataset. Left column shows correctly classified images and right column shows incorrectly classified images (which are all classified as ostriches). Middle column shows the image difference between the left and right column. (Figure taken from [15]).

In an effort to train more robust neural networks, many methods have been developed; empirically, one of the simplest, yet strongest algorithms to yield robust neural networks has been *adversarial training*, which is the focus of this thesis [14]. The goal of adversarial training is to for-

mulate a min-max optimization problem considering a robust loss framework instead of a traditional empirical risk minimization (ERM) framework. Specifically, instead of using a first-order method like stochastic gradient descent to minimize a loss function over an input dataset, we can instead consider a game. In this game, the first player, an adversary, first seeks to maximize the given loss function over some perturbation set for a datapoint, which yields an adversarial example. This is known as the inner maximization step. Then, the goal of the second player, the robust learner, is to minimize the loss function for this adversarial example. This is known as the outer minimization step. This game continues until the robust learner has converged to some model, with the hope that this trained model is robust to adversarial examples that come from the given perturbation set.

Theoretically understanding the convergence of adversarial training when the model is a neural network poses many challenges. Firstly, to yield strong adversarial examples during training, we hope to optimally (or near-optimally) solve the inner maximization problem, which is a non-concave optimization problem for non-linear neural networks. Empirically, the community has observed that using a projected gradient ascent (PGA) algorithm to solve the inner maximization gives strong adversarial examples and is widely regarded as a state-of-the-art defense method [3]. Furthermore, the outer minimization step is a non-convex optimization problem. Despite this, simple first-order descent methods have enjoyed large empirical successes [11, 9], and characterizing the theoretical guarantees of these methods is a long-standing open problem.

In the past, there have been several works on attempting to analyze the convergence of gradient descent for empirical risk minimization. Efforts to understand the empirical successes include a study of the geometry of optimization landscapes for neural networks [4] as well as analyzing the trajectory of network parameters [1]. Recently however, there has been a wave of progress and new directions of research which utilizes the fact that these networks are usually overparameterized i.e. the number of parameters of the neural network are far greater than the number of training datapoints [13, 2, 5]. One interesting work by Jacot et al. finds a connection between gradient descent on overparameterized neural networks and kernel methods [10]. They introduce the *Neural Tangent Kernel* (NTK), an object that arises from the analysis of gradient flow on neural networks. This kernel is a Gram matrix of gradient inner products, and its properties determine training dynamics of neural networks. In the infinite-width limit, they show that this kernel is constant at initialization, which allows us to study the properties at random initialization and characterize training dynamics of neural networks.

Leveraging insights from the Neural Tangent Kernel, Gao et al. analyzed the convergence of the outer minimization step in adversarial training assuming some arbitrary adversary to solve the

inner maximization step [8]. They show that for sufficiently overparameterized neural networks, running projected gradient descent (projecting onto a specific convex set) converges to a network whose robust loss with respect to the chosen adversary is within ϵ of the optimal robust loss with respect to the same adversary.

However, a key question remains: can we show that adversarial training yields a network of low robust loss with respect to an optimal adversary i.e. an adversary than can solve the inner maximization to optimality? In this thesis, we give a partial answer to this question, leveraging the work of Gao et al. to shift the problem focus to understanding the convergence of a fixed adversary, specifically the projected gradient ascent adversary, for the inner maximization problem. Our contributions are as follows:

- If we posit a hypothesis class of linear neural networks, ℓ_p perturbation models, and a mean-squared error loss function, running projected gradient ascent initialized at the center of the constraint ball always converges to a global maximum of the inner maximization problem. This claim along with the results of Gao et al. allows us to prove that with a projected gradient ascent adversary with a fixed initialization scheme, we can find linear neural networks that are provably robust to optimal adversaries.
- If we modify the initialization scheme to be randomly initialized over the constraint ball, as is common in practice, at random initialization of the neural network, we can show that projected gradient ascent will converge to a global maximum under some data density assumptions. The randomness arises from the choice of network weights, the choice of data, and the choice of initialization point for running projected gradient ascent. However, as adversarial training proceeds, assuming the outer minimization problem succeeds in minimizing the loss, the probability with which projected gradient ascent converges to a global maximum decreases. This implies that randomly initialized projected gradient ascent becomes more difficult as the network is trained. We empirically demonstrate this phenomenon and provide some theoretical justifications under data assumptions.
- Extending to non-linear neural networks, we provide empirical evidence that overparameterization benefits the optimization landscape of the inner maximization problem.

Chapter 2

Background

2.1 Problem Setup

We will use lowercase letters to denote row or column vectors, and uppercase letters for matrices. We will use A_i to represent the ith row of A, or b_i to represent the ith element of some row or column vector. We begin our study with two-layer fully-connected neural networks of the form

$$f(x, b, A) = \sum_{i=1}^{m} b_i \sigma(\langle A_i, x \rangle) \tag{2.1}$$

Above, $x \in \mathbb{R}^{d \times 1}$ is the input to the neural network f, which depends on two quantities, $b \in \mathbb{R}^{1 \times m}$ and $A \in \mathbb{R}^{m \times d}$. The quantity m represents the number of neurons in the hidden layer. We assume that the output dimension is 1 (although it can be easily generalized to variable output dimensions as well). We assume that $m \gg d$ and $m \gg n$, which means the network is overparameterized. The function $\sigma(\cdot)$ represents an activation function, which is typically the ReLU (Rectified Linear Unit) function $\sigma(x) = \max(0, x)$. In this thesis, we consider two forms of activation function, the identity function and the ReLU activation function. When the activation function is the identity function $\sigma(x) = x$, the network becomes a linear network and can be written as:

$$f(x, b, A) = bAx \tag{2.2}$$

The standard supervised learning task is: given n training examples and corresponding labels $\{x_i, y_i\}_{i=1}^{n}$, we wish to minimize some Lipschitz, smooth, and convex loss function $\ell(f(x_i, b, A), y)$:

$$\min_{b,A} \sum_{i=1}^{n} \ell(f(x_i, b, A), y) \tag{2.3}$$

In this thesis, we will use the quadratic loss function (also known as the mean squared error loss) throughout, so

$$\ell(f(x_i, b, A), y_i) = \frac{1}{2} \|f(x_i, b, A) - y_i\|_2^2 \tag{2.4}$$

As a note, this does not restrict our setup to only regression problems. We can pose any classification problem as a regression problem where the network must regress on the given labels given the mean-squared error loss function.

2.1.1 Adversarial Training Under General Adversaries

The motivation for adversarial training begins with the observation when models are trained using the standard formulation above, also known as Empirical Risk Minimization (ERM), trained models seem to be empirically very brittle to small perturbations in the input. Formally, this means for some model f, given an example x that is classified as class y, we might be able to perturb x slightly to obtain x' such that $f(x') \neq y$ (i.e. the predicted class changes). To take an example in the case of image classification, a valid perturbation might be to slightly rotate an image of a cat, which does not change the underlying label, but can drastically affect a trained model output [14]. Adversarial training is a training strategy to train models that are robust to such perturbations. We first formalize a framework for studying adversarial training, borrowed from the setup in [8].

Definition 2.1.1. A *perturbation set* $\mathcal{B} : \mathbb{R}^d \to \mathcal{P}(\mathbb{R}^d)$ is a function mapping to the powerset of $\mathbb{R}^d$. It captures the list of acceptable perturbations for any input in $\mathbb{R}^d$. For example, for ℓ_p perturbations, $\mathcal{B}_p(x, \epsilon_0) = \{x' : \|x - x'\|_p < \epsilon_0\}$.

In this thesis, we only consider ℓ_p perturbations, so we use $\mathcal{B}(x, \epsilon_0)$ to represent the ball of ℓ_p perturbations.

Definition 2.1.2. A *perturbation function* $\mathcal{A} : \mathcal{W} \times \mathbb{R}^d \to \mathbb{R}^d$ is a function that simply applies some admissible perturbation to any input x given input as x as well as the weight space of the neural network. Here, admissible means with respect to the chosen perturbation set $\mathcal{B}$. Thus, $\mathcal{A}(W, x)$ simply maps x to some $x' \in \mathcal{B}(x)$.

Definition 2.1.3. A *surrogate loss function* $L_{\mathcal{A}}(W)$ is defined to measure the loss under a perturbation function applied to the data. Specifically,

$$L_{\mathcal{A}}(W) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(W, \mathcal{A}(W, x_i)), y_i)$$

where $\mathcal{A}(W, x_i)$ represents a perturbation function applied to x_i.

Definition 2.1.4. Define the *robust loss* as $L_\star(W)$, which is $L_\mathcal{A}(W)$ with $\mathcal{A}(W, x_i)$ as the "optimal" perturbation, i.e. $\mathcal{A}(W, x_i) = \arg\max_{x \in \mathcal{B}(x)} \ell(f(W, x), y)$. Thus, the robust loss is the optimal loss under the min-max problem posed in conventional adversarial training.

We define $B(W_0, R) = \left\{ W : \left\| W^{(h)} - W^{(h)}(0) \right\|_2 \leq \frac{R}{\sqrt{m}}, h \in [H] \right\}$, which is a convex set denoting a ball around the initialization of W (for R to be defined later). We will run projected gradient descent on this set to learn W. Specifically, this algorithm is:

$$V^{k+1} = W^k - \alpha \nabla_W L_\mathcal{A}(W^k)$$
$$W^{k+1} = \mathcal{P}_R(V^{k+1})$$

Let $W^\star$ denote the optimal setting of the weights for the robust loss, given the constraint $W^\star \in B(W_0, R)$. Thus, $W^\star = \arg\min_{W \in B(W_0, R)} L_\star(W)$. The main results of [8] are demonstrated below, and they show the convergence of projected gradient descent to the robust loss induced by $W^\star$.

Theorem 2.1.1 (From [8])**.** *Given $\epsilon > 0$, suppose $R = \Omega(1)$, and $m \geq \max\left(\Theta\left(\frac{R^9 H^{16}}{\epsilon^7}\right), \Theta(d^2)\right)$. If we run PGD on the convex set $B(W_0, R)$ for T iterations with step size $\alpha = O\left(\frac{\epsilon}{mH^2}\right)$ for $T = \Omega\left(\frac{R^2 H^2}{\epsilon^2}\right)$, then with probability at least 0.99, we have that*

$$\min_{k=1,\ldots,T} L_\mathcal{A}(W^k) - L_\star(W^\star) \leq \epsilon$$

Theorem 2.1.2 (From [8])**.** *If a robust classifier exists in the Reproducing Kernel Hilbert Space (RKHS) of the NTK, there exists $R_{\mathcal{D},\mathcal{B},\epsilon}$ such that when $m = \Omega\left(\frac{R^4_{\mathcal{D},\mathcal{B},\epsilon}}{\epsilon^2}\right)$, with high probability, we have that*

$$L_\star(W) \leq \epsilon$$

and

$$W \in B(W_0, R_{\mathcal{D},\mathcal{B},\epsilon})$$

Putting these two theorems together, we have that with the given assumptions, with high probability, the minimum surrogate loss $L_\mathcal{A}(W^k)$ over T iterations is at most ϵ. This shows that if we perform adversarial training, then the learned network is robust to attacks from the specific adversary

$\mathcal{A}$. In practice, however, we would like to reason about the robustness of the trained network to arbitrary adversaries (over a given perturbation set). If the adversary used during training is weak, then ensuring robustness against this adversary is a weak claim. Observe that for the inner maximization problem, if an adversary solves that problem to optimality, we have shown that the adversary used during training is in fact a strong adversary since $\mathcal{A}(W, x_i) \approx \arg\max_{x \in \mathcal{B}(x)} \ell(f(W, x), y)$.

2.1.2 Adversarial Training Under PGA Adversary

Instead of reasoning about general adversaries, which can be arbitrarily weak, we fix a popular adversary: using projected gradient ascent to solve the inner maximization problem. Denote this adversary as $\mathcal{A}_p$. If we can show that projected gradient ascent (PGA) approximately solves the inner maximization problem, then we will have shown that the network learned from adversarial training has a low robust loss, since the surrogate loss is a a good proxy for the robust loss.
In general, for deep neural networks, the inner maximization problem can be highly non-concave, so this would seem to imply that we cannot solve it to optimality using a simple first-order method like projected gradient ascent [6]. To make insights into this problem, we hope to leverage insights from recent work on understanding the behavior of gradient descent for optimizing overparameterized neural networks.

2.2 Neural Tangent Kernel

To provide some intuition for the results in Theorem 2.1.1, we sketch the use of the Neural Tangent Kernel, which motivates the analysis in proving that theorem [10]. Let f_θ be a standard fully connected neural network with parameters θ. Using discrete-time full batch gradient descent, we can update the parameters θ of the neural network iteratively as follows:

$$\theta_t = \theta_{t-1} - \eta \nabla_{\theta_{t-1}} \mathcal{L}(f_{\theta_{t-1}}(X), y) \tag{2.5}$$

To analyze how the parameters evolve, a natural conversion from Equation 2.5 is to consider the time axis during gradient descent in the continuous domain as opposed to in the discrete domain. This gives a continuous-time gradient flow equation for updating the network parameters, shown below:

$$\begin{aligned} \dot{\theta}_t &= -\eta \nabla_{\theta_t} \mathcal{L}(f_{\theta_t}(X), y) \\ &= -\eta \nabla_{\theta_t} f_{\theta_t}(X)^T \nabla_{f_{\theta_t}(X)} \mathcal{L}(f_{\theta_t}(X), y) \end{aligned} \tag{2.6}$$

Above, $\dot{\theta}_t$ represents a time derivative of θ_t, which defines the gradient flow as a differential equation. The second step follows from applying chain rule to expand the gradient of the loss with respect to the parameters θ_t. Next, we can write how the function evolves across the continuous time domain as another differential equation:

$$\begin{aligned} \dot{f_{\theta_t}}(X) &= \nabla_{\theta_t} f_{\theta_t}(X)\dot{\theta}_t \\ &= -\eta \nabla_{\theta_t} f_{\theta_t}(X) \nabla_{\theta_t} f_{\theta_t}(X)^T \nabla_{f_{\theta_t}(X)} \mathcal{L} \end{aligned} \tag{2.7}$$

Assuming that the neural network outputs a vector of dimensionality n_L, the *neural tangent kernel* is defined as $\Theta_t(X, X) = \nabla_{\theta_t} f_{\theta_t}(X) \nabla_{\theta_t} f_{\theta_t}(X)^T \in \mathbb{R}^{(n_L \cdot n) \times (n_L \cdot n)}$. While this object is time-dependent, Jacot et al. demonstrate that in the infinite-width limit, the NTK at initialization fully determines training dynamics [10]. Furthermore, they derive an expression for the NTK in the infinite width.

An important consequence of this is that we can view the network as being linearized around its initialization, or that the network prediction for a point x $f_\theta^t(x)$ is approximately $f_\theta^0(x) + \nabla_\theta f_\theta^0(x)(\theta_t - \theta_0)$, where $f_\theta^0(x)$ represents the output of the network at random initialization for parameters θ_0 [12]. This implies that for sufficiently wide networks, weights do not need to move much from random initialization in order to reach high accuracy on the given dataset. This is part of the motivation for the explicit projection step in the projected gradient descent algorithm outlined for adversarial training, since it explicitly enforces this constraint.

Chapter 3

Theoretical Results: Convergence For Linear Networks

3.1 Convergence of Inner Maximization

Suppose that $f_\theta(x) = bAx$ where $b \in \mathbb{R}^{1\times m}$ and $A \in \mathbb{R}^{m\times d}$. This is an overparameterized linear network. Suppose that $b_i \sim N(0,1)$ and $A_{ij} \sim N\left(0, \frac{1}{m}\right)$.

Lemma 3.1.1. *For any fixed point $x \in \mathbb{R}^d$, with probability at least $1 - \frac{3}{d}$, $\|\nabla_x f_\theta(x)\|_2$ concentrates around $\sqrt{d}$ with deviations on the order of $\sqrt{\frac{d}{\log(d)}}$.*

Proof. We begin by analyzing the squared norm $\|\nabla_x f_\theta(x)\|_2^2$. We write out its value to get:

$$\|\nabla_x f_\theta(x)\|_2^2 = bAA^Tb^T \tag{3.1}$$

Step 1: Changing product of Gaussians to product of chi-squared distributions.
By the rotational invariance of Gaussians, we can multiply A by an orthogonal matrix and this preserves norms. Specifically, we can always find an orthogonal $U \in \mathbb{R}^{m\times m}$ such that $bU = e_1^T \|b\|_2$, where $e_1 \in \mathbb{R}^{m\times 1}$ is the first standard basis vector. Thus, under U, b is mapped to the unit vector (scaled to be the same norm as b). Note we can always find such a U since U is an orthogonal change of basis matrix. This means we can write the norm of the gradient as follows:

$$\begin{aligned}
\|\nabla_x f_\theta(x)\|_2^2 &\overset{d}{=} bUAA^TU^Tb^T \\
&= \|b\|_2^2 \, e_1 AA^T e_1 \\
&= \|b\|_2^2 \cdot \left\|e_1^T A\right\|_2^2
\end{aligned}$$

The first line above follows from the fact that A is equal in distribution to UA. To see this, we compare their density functions. Let p_1 denote the density for A, and let p_2 denote the density for UA. We can consider a change of variables using a function f that maps from A to UA. Then, we have that the density for p_2 is:

$$p_2(f(x)) = p_1(x)/|\det(J_f(x))|$$

Above, $J(x)$ denotes the Jacobian of the transformation f, which is U. The determinant of this Jacobian is always 1 or -1, which proves that p_2 is equal to p_1, or that A and UA are equal in distribution. Rewriting the norm further, we have:

$$\begin{aligned}
\|b\|_2^2 \cdot \left\|e_1^T A\right\|_2^2 &= \sum_{i=1}^{m} b_i^2 \cdot \sum_{j=1}^{d} A_{0,j}^2 \\
&= \sum_{i=1}^{m} b_i^2 \cdot \sum_{j=1}^{d} \left(\frac{1}{\sqrt{m}} \tilde{A}_{0,j}\right)^2 \\
&= \frac{1}{m} \sum_{i=1}^{m} b_i^2 \cdot \sum_{j=1}^{d} \tilde{A}_{0,j}^2
\end{aligned}$$

where $\tilde{A}_{0,j} \sim N(0,1)$. This allows us to write the norm as $\frac{1}{m}XY$, where $X \sim \chi_m^2, Y \sim \chi_d^2$, which are standard chi-squared distributions with m and d degrees of freedom respectively.

Step 2: Ensuring chi-squared random variables X and Y concentrate in a "nice" region close to their expectation.
Let the variable we are interested in analyzing be $Z = \frac{1}{m}X\frac{1}{d}Y$, where $X \sim \chi_m^2, Y \sim \chi_d^2$. Note that dZ is equal in distribution to the norm we wish to analyze. Let E_1 be the event that Z is far from 1 (for a value to be defined later), and let E_2 be the event that both $\frac{1}{m}X$ and $\frac{1}{d}Y$ are close to their expectation, which is equal to 1. We know that

$$\begin{aligned}
\Pr[E_1] &= \Pr[E_1|E_2]\Pr[E_2] + \Pr[E_1|E_2^c]\Pr[E_2^c] &(3.2)\\
&\leq \Pr[E_1|E_2] + \Pr[E_2^c] &(3.3)
\end{aligned}$$

We start with the event E_2^c, which is the probability that either $\frac{1}{m}X$ or $\frac{1}{d}Y$ deviates from its expectation, since this is easy to analyze. We will first argue that $\frac{1}{m}X$ concentrates around its expectation. Because X is a chi-squared random variable with m degrees of freedom, we know it is a sum of m squared standard normal random variables. Denote these random variables as

$X_1, \ldots, X_m \sim N(0,1)$. Since $X_i \sim N(0,1)$, we know that X_i is a sub-gaussian random variable with squared sub-gaussian norm at most C^2 for some absolute constant C [16]. This implies that X_i^2 is a sub-exponential random variable with sub-exponential norm $\left\|b_i^2\right\|_{\psi_1} = \|b_i\|_{\psi_2}^2 \leq C^2$. We can now apply Bernstein's inequality to get a tail bound on X:

$$\begin{aligned}\Pr\left[\left|\frac{1}{m}X - \mathbb{E}[X]\right| \geq t\right] &= \Pr\left[\left|\frac{1}{m}\sum_{i=1}^{m} X_i^2 - 1\right| \geq t\right] \\ &\leq 2\exp\left\{-c\min\left(\frac{t^2}{K^2}, \frac{t}{K}\right)m\right\}\end{aligned}$$

If we pick $t = \sqrt{\frac{\log(m)}{m}}$, we have that this bound becomes:

$$2\exp\left\{-c\frac{\sqrt{m\log m}}{C^2}\right\} \leq \frac{1}{m}$$

This implies that with probability at least $1 - \frac{1}{m}$, $\frac{1}{m}X$ concentrates close to 1. For $m > 100$, the deviation is between $[0.75, 1.25]$ with probability at least 99%. A similar argument for Y gives that with probability at least $1 - \frac{1}{d}$, $\frac{1}{d}Y \in [0.75, 1.25]$.

Step 3: Changing product of chi-squared into a sum of chi-squared distributions.
We now focus on the probability $\Pr[E_1|E_2]$, or the probability that Z is far from its expectation given that both $\frac{1}{m}X$ and $\frac{1}{d}Y$ are close to their expectations (i.e. in the range $[0.75, 1.25]$). For $t \in [0, 2.5]$, this probability can written as

$$\begin{aligned}\Pr\left[\frac{1}{m}X\frac{1}{d}Y - 1 > t\middle|E_2\right] &= \Pr\left[\frac{1}{m}X\frac{1}{d}Y > t+1\middle|E_2\right] \\ &= \Pr\left[\log\left(\frac{1}{m}X\right) + \log\left(\frac{1}{d}Y\right) > \log(t+1)\middle|E_2\right] \\ &\leq \Pr\left[\log\left(\frac{1}{m}X\right) + \log\left(\frac{1}{d}Y\right) > \frac{t+1}{2}\middle|E_2\right]\end{aligned}$$

The last step follows from the assumption that $t \in [0, 2.5]$, which allows us to apply the lower bound $\log(t+1) \geq \frac{t+1}{2}$. Similarly, because we condition on the event E_2, we can apply the same lower bound to replace $\log\left(\frac{1}{m}X\right)$ and $\log\left(\frac{1}{d}Y\right)$ with a linear term, since if the linear lower bound is greater than $\frac{t+1}{2}$, then it is certainly true that the original logarithmic function is also greater than $\frac{t+1}{2}$. Thus, this probability simplifies to:

$$\begin{aligned}
&\leq \Pr\left[\frac{1}{m}X + \frac{1}{d}Y - > t+1 \middle| E_2\right] \\
&\leq 2\exp\left\{-c\min\left(\frac{t^2}{K^2\left\|a\right\|_2^2}, \frac{t}{K\left\|a\right\|_\infty}\right)\right\}
\end{aligned}$$

where $K = \max(\max_i \left\|X_i^2\right\|_{\psi_1}, \max_j \left\|Y_j^2\right\|_{\psi_1})$ for $X = \sum_{i=1}^m X_i^2$ and $Y = \sum_{j=1}^d Y_j^2$. The vector a denotes the vector of size $m+d$ whose first m values are $\frac{1}{m}$ and the remaining d values are $\frac{1}{d}$ (i.e. the coefficients for the weighted sum of X and Y). We know that $\left\|X_i^2\right\|_{\psi_1} = \left\|Y_i^2\right\|_{\psi_1} \leq C^2$ for some constant C since $X_i, Y_i \sim N(0,1)$, implying that $K = C^2$. Similarly, we calculate $\|a\|_2^2 = \frac{1}{d} + \frac{1}{m}$ and $\|a\|_\infty = \frac{1}{d}$. Plugging these values back in and noting that $m \gg d$, we have the following:

$$\leq 2\exp\left\{-c\frac{dt}{C^2}\right\}$$

We pick $t = \sqrt{\frac{\log(d)}{d}}$. Observe that this is a positive decreasing function in d, whose limit goes to 0, and for $d \geq 1$, this satisfies the assumption that $t \in [0, 2.5]$. Plugging in again for t, we have:

$$\begin{aligned}
&\leq 2\exp\left\{-c\frac{\sqrt{d\log d}}{C^2}\right\} \\
&\leq \frac{1}{d}
\end{aligned}$$

The same argument can be applied for the lower tail as well, yielding the same result. From Equation 3.3 and applying a union bound over X and Y both being far away from 1, we now have that

$$\begin{aligned}
\Pr[E_1] &\leq \Pr[E_1|E_2] + \Pr[E_2^c] \\
&\leq \frac{1}{d} + \frac{1}{m} + \frac{1}{d} \\
&\leq \frac{3}{d}
\end{aligned}$$

Thus, we have shown that with probability at least $1 - \frac{3}{d}$, $Z \in 1 \pm \sqrt{\frac{d}{\log(d)}}$, so Z concentrates around 1 with small deviations. Recall that $\|\nabla_x f_\theta(x)\|_2^2$ was equal in distribution to dZ, so we

also have that with high probability, $\|\nabla_x f_\theta(x)\|_2^2$ concentrates around d. We can extend this to a concentration inequality $\|\nabla_x f_\theta(x)\|_2$ as well by noting that for all numbers $z \geq 0$, $|z-1| \geq \delta$ implies that $|z^2-1| \geq \max(\delta, \delta^2)$, which we can apply for the above analysis to get the same concentration for $\|\nabla_x f_\theta(x)\|_2$ around $\sqrt{d}$, completing the proof. □

Corollary 3.1.2. *With high probability, when we perform projected gradient ascent over $B(x, \epsilon)$ to solve the inner maximization, we will always have a direction of ascent inside the interior of $B(x, \epsilon)$. Thus, the maxima of the problem will lie on the boundary of the ball with high probability.*

3.1.1 Maximizing a Quadratic on the Unit Sphere

Corollary 3.1.2 allows us to restrict our attention to the maximization on the boundary of $B(x, \epsilon)$ as opposed to the interior of the ball. Equivalently, to calculate the form of the maxima, we can horizontally translate the ball and rescale to the unit sphere, which means that without loss of generality, we can focus on an optimization problem over the unit sphere. Further, expanding the quadratic loss, we have:

$$\begin{aligned}
\ell(f(x_i, b, A), y) &= \frac{1}{2}\|f(x_i, b, A) - y\|_2^2 \\
&= \frac{1}{2}(f(x_i, b, A)^2 + y^2 - 2f(x)y) \\
&= \frac{1}{2}(x_i^T A^T b^T b A x_i - 2A^T b^T y x + y^2)
\end{aligned}$$

This expansion shows us that maximizing the quadratic loss over the unit sphere is equivalent to the following optimization problem (for the choice of $w = bA$ and $v = 2yw$):

$$\max_{\|x\|_2^2 = \epsilon} \frac{1}{2} x^T w w^T x - v^T x \tag{3.4}$$

This is equivalent to minimizing the negative of $\psi(x) = \frac{1}{2} x^T w w^T x - v^T x$. We will use the results of [7] and build upon them for our use. Using Lagrange multipliers (or looking at the Riemannian gradient on the unit sphere), we can write out the stationary conditions of the problem as:

$$v - w w^T x = \lambda x \tag{3.5}$$

$$x^T x = \epsilon \tag{3.6}$$

To simplify notation, let $Q = ww^T$, and suppose we take an eigenvalue decomposition of Q to be EDE^T, where $D = \text{diag}(\delta_1, \ldots, \delta_n)$ for $\delta_1 \leq \delta_2 \leq \cdots \leq \delta_n$, and the columns of E are the corresponding orthonormal eigenvectors.

Warm-up: 2-variable case

Suppose $w \in \mathbb{R}^2$ (and $w_1, w_2 \neq 0$), so $Q \in \mathbb{R}^{2\times 2}$. Furthermore, suppose $v = cw$ for some constant c. Then, because Q is by definition rank 1, we have that $\delta_1 = 0$. Now, let $u = E^T x$ and $d = E^T v$. Then, Equations 3.5 and 3.6 become

$$Du = d - \lambda u \tag{3.7}$$

$$u^T u = \epsilon \tag{3.8}$$

We wish to first characterize a set of λ values such that Equations 3.7 and 3.8 hold. Suppose first that $\lambda \neq \delta_1$ and $\lambda \neq \delta_2$. Then, $u = (D + \lambda I)^{-1} d$ because $D + \lambda I$ will be invertible (since it has non-zero determinant). Then, to satisfy Equation 3.8, we have that

$$u^T u = d^T (D - \lambda I)^{-2} d = \left(\frac{d_1}{\delta_1 + \lambda}\right)^2 + \left(\frac{d_2}{\delta_2 + \lambda}\right)^2 = \epsilon$$

As in [7], let $f(\lambda)$ be the explicit secular equation as defined above:

$$f(\lambda) = \left(\frac{d_1}{\delta_1 + \lambda}\right)^2 + \left(\frac{d_2}{\delta_2 + \lambda}\right)^2 - \epsilon = 0 \tag{3.9}$$

For our warm-up, note that $\delta_1 = 0$. Furthermore, we can show that $d_1 = 0$ as shown below.

Lemma 3.1.3. *With the assumption that $v = cw$, we have that $d_1 = 0$, and we have that $d_2 = \frac{c^2 \delta_2 h_1}{v_1} = \frac{c^2 \delta_2 h_2}{v_2}$, where h is the eigenvector corresponding to δ_2.*

Proof. Suppose E stores eigenvectors g and h of Q, where g is the eigenvector corresponding to δ_1 and h is the eigenvector corresponding to δ_2. Then, we have that $E^T Q = E^T w w^T$ can be expanded as follows:

$$\begin{aligned}
E^T Q &= \begin{bmatrix} g_1 & g_2 \\ h_1 & h_2 \end{bmatrix} \begin{bmatrix} w_1^2 & w_1 w_2 \\ w_1 w_2 & w_2^2 \end{bmatrix} \\
&= \begin{bmatrix} g_1 w_1^2 + g_2 w_1 w_2 & g_1 w_1 w_2 + g_2 w_2^2 \\ h_1 w_1^2 + h_2 w_1 w_2 & h_1 w_1 w_2 + h_2 w_2^2 \end{bmatrix} \\
&= \begin{bmatrix} \delta_1 g_1 & \delta_1 g_2 \\ \delta_2 h_1 & \delta_2 h_2 \end{bmatrix} && \text{(Definition of eigenvector)} \\
&= \begin{bmatrix} 0 & 0 \\ \delta_2 h_1 & \delta_2 h_2 \end{bmatrix}
\end{aligned}$$

Next, we expand d:

$$\begin{aligned} d = cE^T w &= c \begin{bmatrix} g_1 & g_2 \\ h_1 & h_2 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \\ &= c \begin{bmatrix} g_1 w_1 + g_2 w_2 \\ h_1 w_1 + h_2 w_2 \end{bmatrix} \\ &= c \begin{bmatrix} \delta_1 g_1 / w_1 \\ \delta_2 h_1 / w_1 \end{bmatrix} && \text{(From above)} \\ &= c \begin{bmatrix} 0 \\ c\delta_2 h_1 / v_1 \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ c^2 \delta_2 h_2 / v_2 \end{bmatrix} \end{aligned}$$

□

This expansion of d simplifies the secular equation to the following equation (where h is the eigenvector of Q corresponding to δ_2):

$$f(\lambda) = \left(\frac{c^2 \delta_2 h_1}{v_1(\delta_2 + \lambda)} \right)^2 - \epsilon = 0 \tag{3.10}$$

If we obtain solutions λ_1 and λ_2, we can then solve for the value of x that are stationary points to the original problem through the equation (for $i = 1, 2$)

$$x = E(D + \lambda_i I)^{-1} d \tag{3.11}$$

Lemma 3.1.4. *The solutions for λ from the secular equation are*

$$\lambda = \delta_2 \left(\pm \frac{c^2 h_1}{v_1 \sqrt{\epsilon}} - 1 \right)$$

Proof. The proof is simple, since we just solve for λ in Equation 3.10, a quadratic equation. Taking the square root of both sides, we have that either

$$\frac{c^2 \delta_2 h_1}{v_1(\delta_2 + \lambda)} = \sqrt{\epsilon} \text{ or } \frac{-c^2 \delta_2 h_1}{v_1(\delta_2 + \lambda)} = \sqrt{\epsilon}$$

Solving for λ, we directly obtain that there are two solutions

$$\lambda = \frac{c^2 \delta_2 h_1 - v_1 \delta_2 \sqrt{\epsilon}}{v_1 \sqrt{\epsilon}} \text{ or } \lambda = \frac{-v_1 \delta_2 \sqrt{\epsilon} - c^2 \delta_2 h_1}{v_1 \sqrt{\epsilon}}$$

which is equivalent to $\lambda = \delta_2 \left(\pm \frac{c^2 h_1}{v_1 \sqrt{\epsilon}} - 1 \right)$. □

From Lemma 3.1.3 and Lemma 3.1.4, we can now obtain a closed-form for the x that are stationary points (i.e. maxima since this is a convex quadratic over the unit sphere).

Theorem 3.1.5. *The stationary points for optimization problem in Equation 3.4 for $w \in \mathbb{R}^2$ and $v = cw$ are*

$$x = \pm c^2 \sqrt{\epsilon} \begin{bmatrix} h_1 \\ h_2 \end{bmatrix}$$

for $i = 1, 2$.

Proof. From Equation 3.11, we know that $x = E(D + \lambda_i I)^{-1} d$. From Lemma 3.1.3, we also know the form of d, which allows us to simplify this calculation. First, for notational purposes, let $D_2 = (D + \lambda_i I)^{-1}$. Since this matrix is diagonal, its inverse is trivial, so we have that

$$D_2 = \begin{bmatrix} 1/\lambda & 0 \\ 0 & 1/(\delta_2 + \lambda) \end{bmatrix}$$

Now expanding the multiplication, we have:

$$\begin{aligned} ED_2 d &= \begin{bmatrix} g_1 & h_1 \\ g_2 & h_2 \end{bmatrix} \begin{bmatrix} 1/\lambda_i & 0 \\ 0 & 1/(\delta_2 + \lambda_i) \end{bmatrix} \begin{bmatrix} 0 \\ c^2 \delta_2 h_1 / v_1 \end{bmatrix} \\ &= \begin{bmatrix} g_1/\lambda_i & h_1/(\delta_2 + \lambda_i) \\ g_2/\lambda_i & h_2/(\delta_2 + \lambda_i) \end{bmatrix} \begin{bmatrix} 0 \\ c^2 \delta_2 h_1 / v_1 \end{bmatrix} \\ &= c^2 \begin{bmatrix} \delta_2 h_1^2 / v_1 (\delta_2 + \lambda_i) \\ \delta_2 h_1 h_2 / v_1 (\delta_2 + \lambda_i) \end{bmatrix} \\ &= \frac{c^2}{(\delta_2 + \lambda_i)} \begin{bmatrix} \delta_2 h_1^2 / v_1 \\ \delta_2 h_2^2 / v_2 \end{bmatrix} \\ &= \pm c^2 \sqrt{\epsilon} \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} && \text{(Lemma 3.1.4)} \end{aligned}$$

□

Theorem 3.1.6. *The stationary points for optimization problem in Equation 3.4 for $w \in \mathbb{R}^2$ and $v = w$ obtain objective function values*

$$\psi(x) = \frac{1}{2}(\alpha_1 + \alpha_2)^2 \pm (c\alpha_1 + c\alpha_2)$$

for

$$\alpha_i = c^2\sqrt{\epsilon}h_i w_i$$

Proof. We have that $\psi(x) = \frac{1}{2}x^T Q x - v^T x$. From Theorem 3.1.5, we know that the optimal x are the positive and negative eigenvectors h of Q scaled by $c^2\sqrt{\epsilon}$. Plugging this into $\psi(x)$, we have that

$$\begin{aligned}
\psi(x) &= \frac{1}{2}c^4\epsilon \begin{bmatrix} h_1 & h_2 \end{bmatrix} \begin{bmatrix} w_1^2 & w_1 w_2 \\ w_1 w_2 & w_2^2 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} - c^2\sqrt{\epsilon} \begin{bmatrix} v_1 & v_2 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} \\
&= \frac{1}{2}c^4\epsilon \begin{bmatrix} h_1 w_1^2 + h_2 w_1 w_2 & h_1 w_1 w_2 + h_2 w_2^2 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} - c^2\sqrt{\epsilon}h_1 v_1 - c^2\sqrt{\epsilon}h_2 v_2 \\
&= \frac{1}{2}c^4\epsilon(h_1^2 w_1^2 + h_1 h_2 w_1 w_2 + h_1 h_2 w_1 w_2 + h_2^2 w_2^2) - c^2\sqrt{\epsilon}h_1 v_1 - c^2\sqrt{\epsilon}h_2 v_2 \\
&= \frac{1}{2}(\alpha_1 + \alpha_2)^2 - \alpha_1 - \alpha_2
\end{aligned}$$

where $\alpha_i = c^2\sqrt{\epsilon}h_i w_i$. □

Generalizing to more than 2 variables

By the same arguments as above, we can show that for $Q = ww^T$ for $w \in \mathbb{R}^d$, we still have that only eigenvector direction matters (corresponding to the non-zero eigenvalue of Q), and this controls the optimal x as well the maximal loss values on the sphere. To see this, note that the form of d does not change and is zero in all entries except one again. Thus, the remaining analysis holds. We summarize these theorems below and note that their proofs are almost identical to the proofs of their corresponding 2-variable statements in the previous section.

Theorem 3.1.7. *The stationary points for optimization problem in Equation 3.4 for $w \in \mathbb{R}^d$ and $v = cw$ obtain objective function values*

$$\psi(x) = \frac{1}{2}\left(\sum_{i=1}^{d} \alpha_i\right)^2 \pm c\sum_{i=1}^{d} \alpha_i$$

for

$$\alpha_i = c^2\sqrt{\epsilon}h_i w_i$$

With this generalization, we can prove a general statement about projected gradient ascent over the ball.

Theorem 3.1.8. *Suppose we run projected gradient ascent over $\mathcal{B}(x, \epsilon)$ to maximize the quadratic loss. Starting at the center of the ball, $x_0 = x$, we have that projected gradient ascent will converge to a global maximizer of the problem.*

Proof. Without loss of generality, it is enough to prove the statement for the optimization problem that is over the unit sphere and starts from the all-zeros vector. Note that the smaller value of λ minimizes the negative objective function (i.e. maximizes the normal objective function). Since $\delta_2 > 0$, the smaller value of λ corresponds to sign that is opposite to the sign of v_1. When we run gradient ascent, this is exactly the direction we move in, and assuming we find a stationary point using gradient ascent, we would always reach a global maximum. □

Introducing Randomness

Our first effort at introducing randomness into the problem will be to suppose that $w_i \sim N(0, 1)$, so Q is a random matrix. In the context of our 2-layer linear network, this is equivalent to setting $b = (1, \ldots, 1)$ and setting $A_{ij} \sim N(0, \frac{1}{m})$ - such an assumption to keep the last layer fixed is often made to study the effect of overparametrization [4]. Then, $w = bA$ is a d-dimensional vector such that $w_i \sim N(0, 1)$.

Lemma 3.1.9 (Upper bound on local to global maxima difference)**.** *Under the random choice of w, with high probability, the difference between objective function values at the local and global maxima can be at most $O(\sqrt{d})$.*

Proof. From Theorem 3.1.7, we have the form of the objective function for the two stationary points, the local and global maximum. Let f_1 denote the objective function value at one maxima and f_2 the objective function value at the other maxima. Considering their difference, we have:

$$
\begin{aligned}
|f_1 - f_2| &= \left| \left(\frac{1}{2} \left(\sum_{i=1}^{d} \alpha_i \right)^2 + c \sum_{i=1}^{d} \alpha_i \right) - \left(\frac{1}{2} \left(\sum_{i=1}^{d} \alpha_i \right)^2 - c \sum_{i=1}^{d} \alpha_i \right) \right| \\
&= \left| 2c \sum_{i=1}^{d} \alpha_i \right| \\
&\leq 2c \sum_{i=1}^{d} |\alpha_i| && \text{(Triangle Inequality)} \\
&= 2c \left\| \alpha \right\|_1 \\
&= 2c^3 \sqrt{\epsilon} \left\| \langle h, w \rangle \right\|_1 \\
&\leq 2c^3 \sqrt{\epsilon} \left\| h \right\|_2 \left\| w \right\|_2 && \text{(Hölder's inequality)}
\end{aligned}
$$

Recall that h is a orthonormal eigenvector, so its norm is 1. Since w is a Gaussian random vector, its expected ℓ_2 norm concentrates around $\sqrt{d}$ with high probability [16]. We can make this probability arbitrarily high for any choice of deviation that is a multiple of $\sqrt{d}$. Thus, with high probability, we have that the difference between objective function values at the two stationary points is at most $O(\sqrt{d})$. □

The above upper bound can also be tight, which shows that the ideal property that all local maxima are equal in value (or close in value) to the value of the global maxima is not true. However, we have already shown that we can run projected gradient ascent from the center of the ball and always reach the global maximizer. In practice, randomly initialized projected gradient ascent is much more common than fixing an arbitrary starting point, so we would like to prove a statement about the optimization landscape given this algorithm. We first define some terms that we will use afterwards. Recall that in the given optimization problem, there are two directions that lead to stationary points. As Lemma 3.1.9 shows, in general, the objective function values between the local and global maxima can be high. The observation is that this discrepancy is dictated by the position of the global minimizer of the problem with respect to the center of the ball $\mathcal{B}(x, \epsilon)$. To see this, note the stationary condition for the minimizer of the quadratic function is:

$$v - ww^T x = 0 \tag{3.12}$$

For $v = cw$, we have that $ww^T x = cw$, which implies that all global minima of the problem lie on the hyperplane $w^T x = c$.

Definition 3.1.1. We denote the hyperplane $w^T x = c$ as the *separating hyperplane* of the optimization problem in Equation 3.4.

Specifically, note that all points on one side of the hyperplane converge to the same stationary point. From Theorem 3.1.8, we know that if we initialize projected gradient ascent to any point on the side of the hyperplane that contains the center of the ball, we will then converge to a global maximizer of the problem. To prove a stronger statement about the optimization landscape of Equation 3.4, we make an assumption on the specific data distribution. This is because the data distribution $\mathcal{D}$ determines the center of the ball $\mathcal{B}(x, \epsilon)$ for $x \sim \mathcal{D}$.

Lemma 3.1.10. *Suppose our data is drawn from the zero-centered sphere of radius $\epsilon + c_{\max}$ under a uniform density, where $c_{\max}$ is the largest label value for any point in the given dataset. Then, with probability at least 99% over the random choice of data and choice of w, for any point x drawn under the density, the separating hyperplane (defined by $w^T x = c$) is at least distance $O\left(\frac{\epsilon}{\sqrt{d}}\right)$ from the center of the ball $\mathcal{B}(x, \epsilon)$.*

Proof. For a given point v, the distance s from v to the separating hyperplane is:

$$s = \frac{|w^T v - c|}{\|w\|_2}$$

Note that if our data comes from the sphere of radius ϵ, the center of the ball $\mathcal{B}(x, \epsilon)$ is a point on this sphere, since we run the inner maximization over a ball from any point in the original dataset. Consider the random variable $g \sim N(0, I_d)$ and let $v = (\epsilon + c_{\max})\frac{g}{\|g\|_2}$. Then, v is uniformly distributed on the sphere of radius $\epsilon + c$, so v can be thought of as a datapoint drawn from a uniform measure over the sphere [16]. We can then calculate $\mathbb{E}[s]$ as:

$$\begin{aligned}
\mathbb{E}[s] &= \mathbb{E}\left[\frac{|w^T v - c|}{\|w\|_2}\right] \\
&= \mathbb{E}\left[\left|\frac{w^T v - c}{\|w\|_2}\right|\right] \\
&\geq \mathbb{E}\left[\left|\left|(\epsilon + c_{\max})\langle\frac{w}{\|w\|_2}, \frac{g}{\|g\|_2}\rangle\right| - \frac{|c|}{\|w\|_2}\right|\right]
\end{aligned}$$

Because g and w are distributed as isotropic Gaussians of dimension d, they are near orthogonal in high dimensions, so we have that $\left\|(\epsilon + c_{\max})\langle\frac{w}{\|w\|_2}, \frac{g}{\|g\|_2}\rangle - O\left(\frac{\epsilon + c_{\max}}{\sqrt{d}}\right)\right\|_{\psi_2} \leq C$ for some absolute constant C, where $\|\cdot\|_{\psi_2}$ denotes the sub-gaussian norm [16]. By the definition of the sub-gaussian norm, this gives us that with probability at least 99% over the random choice of both w and g, this term will not deviate by a constant from $\frac{\epsilon + c_{\max}}{\sqrt{d}}$. Similarly, we can analyze the second term to also concentrate around $\frac{c}{\sqrt{d}}$ with high probability, where this probability is over the choice of w. Because $c_{\max} \geq c$, we conclude with probability at least 99%, if we randomly choose both the data and the choice of weights w, the distance from the center of the ball (the chosen datapoint) to the hyperplane (determined by w) is at least $O\left(\frac{\epsilon}{\sqrt{d}}\right)$. □

Theorem 3.1.11. *Suppose our data is drawn from the zero-centered sphere of radius $\epsilon + c_{\max}$ under a uniform density. Under this measure, then draw a datapoint x and run projected gradient ascent initialized from a uniformly drawn point in $\mathcal{B}(x, \epsilon)$. With probability at least 95% over the random choice of data, the random initialization of the network, and the chosen initialization point for projected gradient ascent, this algorithm will converge to a global maximizer.*

Proof. We will prove this noting the geometric property that the point that projected gradient ascent converges to depends on which side of the separating hyperplane the initialization point lies on. Thus, the proof strategy will be to show that if the separating hyperplane is sufficiently far from the center of the ball, then most of the volume of the ball is contained on one side of the ball, specifically the side containing the center of the ball. If we show this, then we will have shown that with high probability over the choice of random initialization point, running projected gradient ascent from any point uniformly at random in $\mathcal{B}(x, \epsilon)$ will converge to a global maximizer.

First, we show that the separating hyperplane is sufficiently far from the center of the ball. Since both the center of the ball and the separating hyperplane itself are random variables, from Lemma 3.1.10, under the random choices of these two variables, with probability at least 99%, this distance is at least $O\left(\frac{\epsilon}{\sqrt{d}}\right)$. Thus, our problem reduces to analyzing the probability of convergence based on the random initialization point, which we can analyze using the aforementioned volume argument.

To prove the concentration of volume on one side of the hyperplane, we follow the proof strategy of the "blow-up" lemma [16]. Without loss of generality, we prove the claim for the unit sphere S^{d-1}, and suppose the distance from the center of the ball to the hyperplane is $O\left(\frac{1}{\sqrt{d}}\right)$ to account for this. Let $H = \{x \in S^{d-1} : x_1 \leq 0\}$ denote one hemisphere of the unit sphere. Denoting $\sigma(\cdot)$ as the normalized area on the sphere, we know that $\sigma(H) = \frac{1}{2}$ since it covers half the sphere. Let $H_t = \{x \in S^{d-1} : \exists y \in A \text{ such that } \|x - y\|_2 \leq t\}$. Viewing these sets as one side of some separating hyperplane, we see that H corresponds to a hyperplane that cuts through the origin, and H_t corresponds to a hyperplane that is translated by distance t. We can view $\sigma(\cdot)$ as equivalent to a uniform measure over the sphere, which means that it is equal to the random variable $X \sim \text{Unif}(S^{d-1})$ with sub-gaussian norm $\|X\|_{\psi_2} \leq \frac{C}{\sqrt{d}}$ for some absolute constant C. Further, we claim that

$$\{x \in S^{d-1} : x_1 \leq t/\sqrt{2}\} \subset H_t$$

To see this, let $x \in S^{d-1}$ be an element such that $x_1 \leq t/\sqrt{2}$. Then, let y' be x, except $y_1 = 0$. Note that $y \in H$. We have that $\|x - y'\|_2 = \frac{t}{\sqrt{2}} < t$, which implies that $x \in H_t$. Using this fact and utilizing the definition of the sub-gaussian tail, we have

$$\begin{aligned}\sigma(H_t) &= \Pr[X \in H_t] \\ &\geq \Pr[X_1 \leq t/\sqrt{2}] \\ &\geq 1 - 2\exp\left\{-ct^2/\left\|X\right\|_{\psi_2}^2\right\}\end{aligned}$$

For our problem, $t = O\left(\frac{1}{\sqrt{d}}\right)$. Using the absolute constants for the bound, we have that with probability at least 99%, most of the volume of the ball will be on one side of the hyperplane. Taking a union bound over the failure probability of Lemma 3.1.10 which depends on the randomness of w and data x, we have our desired result. □

Two-Layer Randomness

Now, if we suppose that b is not fixed to the all-ones vector, we can appeal to Lemma 3.1.1 to prove similar statements as above. Specifically, we notice that the proofs above simply depend on the norm of w, which concentrates around $\sqrt{d}$ for a Gaussian random vector. Lemma 3.1.1 shows that bA behaves similarly (but does not concentrate as well), which implies that the above analyses can be restated for two-layer linear networks as well.

3.2 Extending to Outer Minimization

Theorem 3.1.8 implies that for any setting of weights in the outer minimization, we can always optimally solve the inner maximization problem by running projected gradient descent initialized at the center of the ball. This result along with the result from Gao et al. yields our overall statement that adversarial training finds a linear network of low robust loss with respect to an optimal adversary.

In the previous section, we also argued that randomly initialized projected gradient ascent will converge to a global maximizer with high probability at random initialization of the neural network, where our choices of randomness arose from the assumption on data density to be a uniform measure over a sufficiently large sphere, the random initialization of the network, and the random initialization of projected gradient ascent. However, we cannot expect this to hold at all points throughout training. To see this, note that the separating hyperplane represents the hyperplane for which we correctly classify that datapoint. Thus, as we minimize the overall loss in the outer minimization, we expect the datapoint to continually get closer to the separating hyperplane and increasing the probability of reaching a local maximum from randomly initialized projected gradient ascent. In the following section, we empirically demonstrate this phenomenon.

Chapter 4

Experiments

4.1 Linear Networks

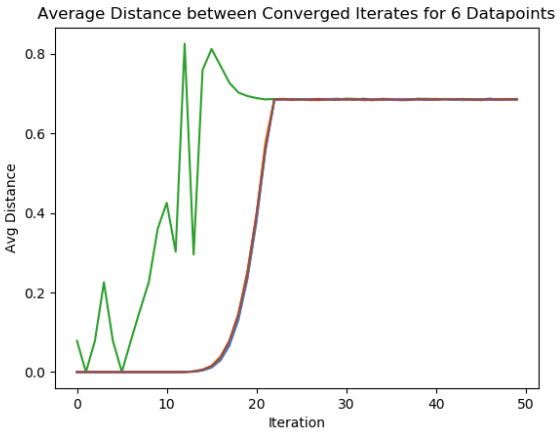


Figure 4.1: At each iteration of adversarial training, we run to convergence 50 iterations of PGA and report the average distance between the 50 converged iterates. At iteration 20, the outer minimization loss converges. For this setup, $\epsilon = 1$, $d = 100$, and $m = 150$.

Setup

We sample data from the sphere of dimension 100 and radius 5. This is to ensure that Assumption **??** is met with high enough probability at random initialization, as argued in Lemma 3.1.10. We run adversarial training for 50 iterations with learning rate set to 0.01, and the model used is a 2-layer neural network with 150 hidden neurons. The setup is a binary classification task, where labels are generated uniformly at random from $\{0, 1\}$ for each datapoint. Note that the assignment of labels does not influence the result of Lemma 3.1.10, since that already performs the worst-case

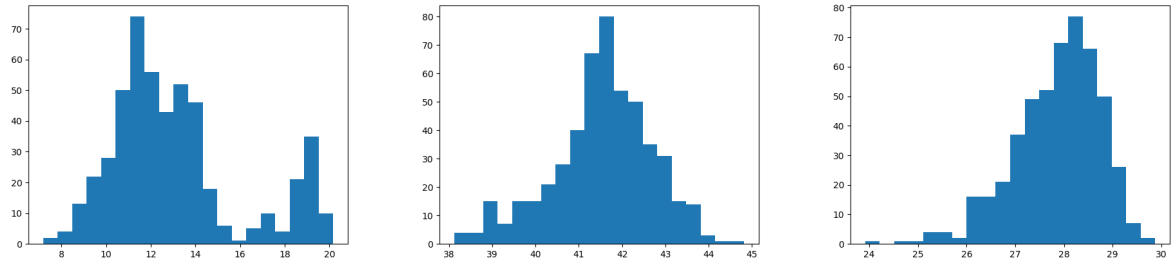
analysis of when all measure lies on data points with a large label, ensuring that the separating hyperplane is far from the origin, and could be to close to any point on the ϵ-sphere.

Discussion

Empirically, we first verify Assumption **??** over the course of adversarial training for linear networks. While we showed that at random initialization, with a constant high probability, randomly initialized projected gradient ascent will converge to a global maximizer, we also demonstrated an argument for why this should not hold across adversarial training. In Figure 4.1, we see that for 6 randomly chosen datapoints, at random initialization, we converge to the same iterate (the global maximum) with high probability, since the average difference between iterates is low. This is because the separating hyperplane is far enough from the balls that center around each datapoint, satisfying Assumption **??**. However, as training continues, the datapoints move closer to the separating hyperplane. This is also intuitive because the separating hyperplane represents minimizing the loss function (i.e. labelling all points correctly), so over the course of training, the datapoints will always naturally shift to the separating hyperplane assuming convergence of the outer minimization.

An implication of this is that the convergence of the inner maximization for adversarial training does not follow arguments such as the Neural Tangent Kernel, for which we can make claims that analyzing dynamics at random initialization is enough to characterize training dynamics for all of training. Even if we suppose that the outer minimization problem follows arguments from NTK theory, small perturbations in weight space can amplify to lead to large output changes.

4.2 Non-Linear Networks



(a) $m = 100$. Average distance between converged iterates: 0.99

(b) $m = 500$. Average distance between converged iterates: 0.40

(c) $m = 1000$. Average distance between converged iterates: 0.39

Figure 4.2: Analysis of Inner Maximization at Random Initialization: We plot a histogram of loss values, where the x axis represents the loss values obtained over 500 random initializations of projected gradient ascent, and the y axis represents the histogram counts.

We follow the same problem setup as for the linear networks in the previous section. For non-linear networks, we observe that even at random initialization, the optimization landscape is difficult to analyze, and concentration of maxima is not as clear as the linear case. From Figure 4.2, we demonstrate a similar result to Madry et al., who observe that the distribution of maxima for the inner maximization concentrates well under random initializations. However, the effect of overparameterization from the loss landscape is not as clear as analyzing the distribution of iterates obtained. As we can see in the captions of Figure 4.2, the average distance between obtained iterates falls as we increase the degree of overparameterization. This suggests that unlike the linear case, where the landscape depends only on the location of the separating hyperplane, the optimization landscape depends on the size of the network in the nonlinear case. This is expected since the optimization problem in the inner maximization is no longer a convex quadratic, since a non-linearity is applied to the input datapoints. We leave further analyzing the landscape of this problem to future work, but note that a successful approach to analyzing this problem would likely move away from landscape analysis, since it seems that there are many local optima for this problem. We discuss potential approaches to understanding this problem in the following section.

Chapter 5

Conclusion and Future Work

In this thesis, we explored the optimization landscape of the inner maximization problem for adversarial training. In the case of linear neural networks, our main findings are that projected gradient ascent initialized at the center of the ball is preferred to randomly initialized projected gradient ascent. If we initialize at the center of the ball, we reach a global maximizer of the inner maximization problem with probability 1, which implies that adversarial training finds a network of low robust loss with respect to any optimal adversary under that perturbation framework. On the other hand, the performance of randomly initialized projected gradient ascent degrades as we continue adversarial training.

There are several avenues for future work that would provide more insight into this problem. First, the analysis for the linear case can likely be extended to handle any convex and Lipschitz loss function. The observation that the separating hyperplane dictates the difference between the local and global maxima would hold for any such α-strongly convex loss function for which we can bound the growth of the function.

A more open direction would be to begin to make insights into the non-linear case. While the linear case is much easier than the non-linear case since we can find a closed-form solution to the optimization problem, we hope that the insights from the linear case can help motivate theoretical analyses for understanding the convergence of projected gradient ascent for the inner maximization. Specifically, we conjecture that randomly initialized projected gradient ascent also begins to degrade as adversarial training continues, which is a phenomenon observed in [17].

It is likely the case that performing a landscape analysis on the inner maximization problem for non-linear networks is a difficult problem, as landscape analysis for ERM for even three layer non-

linear neural networks has yielded strong negative results. One technique towards getting around this analysis could be to do algorithm-based analysis of the convergence of projected gradient ascent. However, unlike the intuition for the Neural Tangent Kernel arguments, instead of showing that weights do not need to deviate much from initialization to minimize the loss function, we have shown that with high probability, we always maximize the loss function on the boundary of the constraint ball. This implies that understanding the inner maximization problem probably requires an analysis of optimization beyond the NTK regime, which is an open problem for neural network learning as a whole.

References

[1] Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in Neural Information Processing Systems* (2019), pp. 6155–6166.

[2] Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. *arXiv preprint arXiv:1811.03962* (2018).

[3] Carlini, N., and Wagner, D. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)* (2017), IEEE, pp. 39–57.

[4] Du, S. S., and Lee, J. D. On the power of over-parametrization in neural networks with quadratic activation. *arXiv preprint arXiv:1803.01206* (2018).

[5] Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054* (2018).

[6] Engstrom, L., Tran, B., Tsipras, D., Schmidt, L., and Madry, A. A rotation and a translation suffice: Fooling cnns with simple transformations. *arXiv preprint arXiv:1712.02779* (2017).

[7] Gander, W., Golub, G. H., and von Matt, U. A constrained eigenvalue problem. *Linear Algebra and its applications 114* (1989), 815–839.

[8] Gao, R., Cai, T., Li, H., Wang, L., Hsieh, C.-J., and Lee, J. D. Convergence of adversarial training in overparametrized networks. *arXiv preprint arXiv:1906.07916* (2019).

[9] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 770–778.

[10] Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems* (2018), pp. 8571–8580.

[11] Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (2012), pp. 1097–1105.

[12] Lee, J., Xiao, L., Schoenholz, S. S., Bahri, Y., Sohl-Dickstein, J., and Pennington, J. Wide neural networks of any depth evolve as linear models under gradient descent. *arXiv preprint arXiv:1902.06720* (2019).

[13] Li, Y., Ma, T., and Zhang, H. Algorithmic regularization in over-parameterized matrix sensing and neural networks with quadratic activations. *arXiv preprint arXiv:1712.09203* (2017).

[14] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* (2017).

[15] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

[16] Vershynin, R. *High-dimensional probability: An introduction with applications in data science*, vol. 47. Cambridge University Press, 2018.

[17] Wang, Y., Ma, X., Bailey, J., Yi, J., Zhou, B., and Gu, Q. On the convergence and robustness of adversarial training.