香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

Understanding Adversarially Robust Generalization: A Learning Theory Perspective

Jiancong Xiao

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Doctor of Philosophy
in
Computer and Information Engineering

The Chinese University of Hong Kong, Shenzhen
February 11th 2023

Thesis Assessment Committee

Committee Chairman:	Prof. Hongyuan Zha
Supervisor:	Prof. Zhi-Quan Luo
Other Member:	Prof. Ruoyu Sun
Examiner from CUHK:	Prof. Anthony Man-Cho So
External Examiner:	Prof. Weijie Su (University of Pennsylvania)

To my parents, my sister, and my wife.

Abstract

Abstract of thesis entitled:

Understanding Adversarially Robust Generalization: A Learning Theory Perspective

Submitted by Jiancong Xiao

for the degree of Doctor of Philosophy

at The Chinese University of Hong Kong, Shenzhen in February 11$^{\text{th}}$ 2023

Deep neural networks (DNNs) have become successful in many machine learning tasks. However, they are shown to be vulnerable to adversarial examples. Adding perturbed samples into the training dataset effectively improves robustness, but it still cannot lead to satisfactory performance. It is observed that the major challenges come from generalization in practice. In this thesis, we study the poor adversarially robust generalization from the perspective of learning theory.

Firstly, we study the adversarial Rademacher complexity (ARC) of deep neural networks. Unlike the well-developed Rademacher complexity bounds in standard training, how to bound ARC in multi-layers cases is largely unclear due to the difficulty of analyzing adversarial loss in the definition of ARC. We provide a solution to this unsolved problem. Specifically, we provide the first bound of adversarial Rademacher complexity of deep neural networks. We provide a novel adversarial perturbation bound such that we are able to calculate the ARC. The bounds of ARC provide an interpretation of poor adversarial generalization.

Secondly, we study the robust overfitting issue of adversarial training by using tools from uniform stability. One major challenge is that the outer function (as a maximization of the inner function) is nonsmooth, so the standard technique (*e.g.,* (Hardt et al., 2016)) cannot be applied. Our approach is to define and consider η-approximate smoothness. We derive stability-based generalization bounds for stochastic gradient descent (SGD) on the general class of η-approximate smooth functions, which covers the adversarial loss. Our results suggest adversarial training can have nonvanishing generalization errors. Robust test accuracy decreases in ϵ when T is large, with a speed between $\Omega(\epsilon\sqrt{T})$ and $\mathcal{O}(\epsilon T)$. A natural question arises: can we eliminate the generalization error floor in adversarial training? Chapter 4 gives an affirmative answer. We employ a smoothing technique to smooth the adversarial loss function. Based on the smoothed loss function, we design a smoothed ME-SGD achieving a generalization bound $\mathcal{O}(T/n)$, which eliminates the generalization error floor.

Lastly, we study the properties of on-manifold adversarial examples. One of the hypotheses of the existence of adversarial examples is the off-manifold assumption: adversarial examples lie off the data manifold. However, we prove that on-manifold adversarial examples are powerful, yet adversarial training focuses on off-manifold directions and ignores the on-manifold adversarial examples. Our analysis suggests that on-manifold adversarial examples are important, and we should pay more attention to on-manifold adversarial examples for training robust models.

摘要

深度神经网络在许多机器学习任务中取得了成功。然而，它们容易受到对抗样本的攻击。在训练数据中加入对抗样本能有效地提高了模型鲁棒性，但性能仍无法让人满意。通过实验观察，研究人员发现其主要难点来源于鲁棒泛化。在本论文中，我们从经典学习理论的角度研究对抗鲁棒泛化。

首先，我们研究了深度神经网络的对抗 Rademacher 复杂度。在标准设定下，Rademacher 复杂度上界的研究已经比较完善。然而，在对抗设定下，如何计算 Rademacher 复杂度还不清楚。主要困难来源于对抗损失函数的复杂性。对这个公开问题，我们给出了一个解。我们首次证明了深度神经网络的对抗 Rademacher 复杂度的上界。我们的方法是给出了一个新的对抗扰动上界以计算对抗 Rademacher 复杂度。该复杂度上界能够为对抗鲁棒泛化困难给出一个解释。

其次，我们从一致稳定性的角度来研究对抗训练的鲁棒过度拟合问题。一个主要挑战是对抗损失函数是非光滑的，因此无法将现有结果应用到对抗设定。我们定义了 η 近似平滑度并给出了其性质。基于此，我们给出了此类函数的一致稳定性泛化上界，它可以应用到对抗训练。我们的结果表明对抗训练可能具有不消失泛化误差。当 T 很大时，模型的鲁棒测试准确率会随着 T 下降，速度在 $\Omega(\epsilon\sqrt{T})$ 和 $\mathcal{O}(\epsilon T)$ 之间。一个自然的问题出现了：可以消除对抗训练泛化上界中的非零误差项吗？我们在第四章给出了肯定的答案。我们采用一个平滑技术来平滑对抗损失函数。基于该损失函数，我们设计了一种平滑的 ME-SGD 算法，得到了泛化误差上界 $\mathcal{O}(T/n)$，从而消除了不消失泛化误差项。

最后，我们研究了流形上对抗样本的特性。对抗样本的存在性的一个重要假设是流形外假设：对抗样本位于数据流形之外。然而，我们证明了流形上的对抗样本是强大

的，但对抗训练侧重于流形外的方向而忽略了流形上的对抗样本。我们应该更加关注流形上的对抗样本来训练稳健的模型。

Acknowledgement

First and foremost, I would like to thank my supervisor, Professor Zhi-Quan Luo, for giving me the freedom to work on research questions that I am passionate about and for providing invaluable guidance during this research. I deeply appreciate that he trusted me and allowed me to switch my research area to learning theory in the third year of my Ph.D. study. Despite his busy schedule, he made time to hold individual meetings and group meetings regularly to discuss the research problems with me. He has taught me how to investigate and get deeper and deeper into a research problem. He also taught me how to be a good and real researcher in an incredibly fast-paced AI research area. He spent lots of time teaching me how to improve my writing, communication, and presentation skills and spent lots of time polishing and proofreading the paper. What he has taught me is not only the understanding of research problems but also the understanding of life.

Secondly, I would like to thank my committee members, Professor Hongyuan Zha, Professor Ruoyu Sun, Professor Anthony Man-Cho So, and Professor Weijie Su, for reviewing my thesis and providing invaluable comments and feedback to improve the quality and presentation of my research. Special thanks to Prof. Ruoyu Sun for the help when I am working on robust generalization theory. He has an incredible and inconceivable understanding of deep learning theory and will never hesitate to discuss the mathematical details and proving techniques with me when I ask. I am grateful for the mentorship of Dr. Yanbo Fan during my internship at Tencent AI Lab. He has tremendous experience in empirical adversarial machine learning research. I enjoy

discussing how to apply the theory to the real world with him. I would also like to thank Professor Tsung-hui Chang and Professor Xiang Wan for their help when I was working on a clinical data information extraction project. They taught me a lot about how to do research when I was a junior Ph.D. student. Many thanks to other professors or researchers with whom I shortly worked or communicated. I also learned a lot from them.

Thirdly, I would like to thank all the collaborators and co-authors. Special thanks to Zeyu Qin and Jiawei Zhang. I will never forget the days we spent together, from having an initial idea to finishing a research paper. I would also like to thank Dr. Liusha Yang, Professor Baoyuan Wu, Dr. Jue Wang, and Professor Asuman Ozdaglar for providing comments and suggestions in this research.

Fourthly, I am grateful to my officemates at Daoyuan 225 and Research Building 310, Wenqiang Pu, Jingwei Mao, Jiawei Zhang, Hao Liang, Liping Tang, Tianjian Zhang, Shuyi Ren, Ye Liu, Congliang Chen, Yingru Li, Boyuan Wang, Xiang Liu, Wenxuan Wang, Qingyan Meng, Kai Li, Ying Li, Liangqi Liu, Yushun Zhang, Ziniu Li, Lei Liu, Rongjun Tang, Wentao Lei, Sha Lai, Dmitry Rybin, Sergei Kudria, Dawei Li, Peijun Xiao and many others who have been in or visited our office. Special thanks to those who help to proofread this thesis.

I dedicate this thesis to my parents, Ziliang Xiao and Shaojian Ke. They have no idea what graduate study or academic research is, but they unconditionally support me chasing the life I want. I also dedicate this thesis to my sister Qianying Xiao. She encourages me to keep working on mathematical theory. Without the support from my family, none of this would have been possible. Finally, I also dedicate this thesis to my wife, Baoying Huang. I run out of my luck to have met you in my life. I am grateful for your company and look forward to future adventures with you.

Contents

List of Figures

List of Tables

Acronyms

AI	artificial intelligence
DNNs	deep neural networks
CV	computer vision
NLP	natural language processing
AT	adversarial training
RC	Rademacher complexity
VC-dim	Vapnik-Chervonenkis dimension
ARC	adversarial Rademacher complexity
UAS	uniform argument stability
SGD	stochastic gradient descent
ME-SGD	Moreau envelope-SGD
PGD	projected gradient descent
FGSM	fast gradient sign method
SWA	stochastic weight averaging
GAN	generative adversarial network
VAE	variational auto-encoder
Gen-AE	generative adversarial examples
Eigen-AE	eigenspace adversarial examples
GAT	generative adversarial training
EAT	eigenspace adversarial training

Chapter 1

Introduction

1.1 Background

Deep neural networks (DNNs) (Krizhevsky et al., 2012; Hochreiter & Schmidhuber, 1997) have become successful in many machine learning tasks such as computer vision (CV) and natural language processing (NLP). However, they are shown to be vulnerable to adversarial examples (Szegedy et al., 2013; Goodfellow et al., 2014b). More specifically, a well-trained model performs badly on slightly perturbed data samples. The notion of adversarial examples for deep learning is relatively young, with their initial discovery posted on arXiv in 2013 (Szegedy et al., 2013). However, due to the mysterious power of adversarial examples to threaten DNNs, it has attracted extensive attention rapidly since then. Nowadays, adversarial attacks and defense have become one of the central problems in deep learning. We start by introducing the concept of adversarial examples.

1.1.1 Adversarial Examples

Adversarial examples were originally introduced by (Szegedy et al., 2013). They are inputs to machine learning models that an attacker has carefully designed to cause the model to make a mistake. Let f_w be a neural network parameterized by w. By adding a small perturbation to the original image $\boldsymbol{x}$, the crafted adversarial example $\boldsymbol{x}'$ can

make the neural network f_w predict a wrong label, *i.e.,* $f_w(\boldsymbol{x}') \neq y$. The perturbation is usually small, such that it is imperceptible to human eyes, *i.e.,* $\|\boldsymbol{x} - \boldsymbol{x}'\| \leq \epsilon$ for some norms.

Mathematically, crafting adversarial examples is to solve the following optimization problem,

$$\max_{\|\boldsymbol{x}-\boldsymbol{x}'\|_p \leq \epsilon} \ell(f_w(\boldsymbol{x}'), y), \tag{1.1.1}$$

where $\ell(\cdot, \cdot)$ is the loss function, $(\boldsymbol{x}, y)$ is the input-label pair, and $\|\cdot\|_p$ is the ℓ_p-norm with $p \geq 1$. The earliest algorithm for generating adversarial examples is the fast gradient sign method (FGSM),

$$\boldsymbol{x}' = \boldsymbol{x} + \epsilon \cdot \operatorname{sign}(\nabla_{\boldsymbol{x}} \ell(f_w(\boldsymbol{x}), y)), \tag{1.1.2}$$

where even a single, small gradient step was sufficient to hurt the performance of deep learning models (Goodfellow et al., 2014b). After that, projected gradient descent (PGD) (Madry et al., 2017) becomes a more popular algorithm for finding adversarial examples. PGD-attack is to use multiple gradient steps, *i.e.,*

$$\begin{aligned} &\boldsymbol{x}^0 \sim \{\boldsymbol{x}' \mid \|\boldsymbol{x} - \boldsymbol{x}'\|_p \leq \epsilon\}, \\ &\boldsymbol{x}^{t+1} = \operatorname{Proj}_{\{\boldsymbol{x}' \mid \|\boldsymbol{x}-\boldsymbol{x}'\|_p \leq \epsilon\}}[\boldsymbol{x}^t + \alpha_t \cdot \nabla_{\boldsymbol{x}} \ell(f_w(\boldsymbol{x}^t), y)], \end{aligned} \tag{1.1.3}$$

to find a more powerful adversarial example. x^0 is sampled, *e.g.,* uniformly, from the feasible set. Proj is the projection operator, and α_t is the step size.

Nowadays, methods for generating these adversarial examples, such as adaptive attacks (Tramer et al., 2020) and AutoAttacks (Croce & Hein, 2020), are significantly more sophisticated and powerful. For more examples, see Sec. 1.3. Standard-trained models will completely fail (*i.e.,* achieve zero adversarially robust accuracy) when evaluated on adversarial examples.

1.1.2 Adversarial Robustness

Since standard-trained models are vulnerable to adversarial examples, a great amount of work has looked towards improving DNNs performance against adversarial attacks, resulting in what is referred to as adversarial defenses. To improve the adversarial robustness of DNNs, we aim to minimize the worst-case loss against an adversarial example. Mathematically, this can be formulated as the following min-max optimization problem:

$$\min_{w} \max_{\{\boldsymbol{x}' | \|\boldsymbol{x}-\boldsymbol{x}'\|_p \leq \epsilon\}} \ell(f_w(\boldsymbol{x}'), y), \tag{1.1.4}$$

where the attacker (maximizer) aims to find an adversarial example $\boldsymbol{x}'$ to fool the target model f_w, and the defender aims to find a more robust model f_w to correctly predict the label of $\boldsymbol{x}'$. Adversarial training (Madry et al., 2017) (and its variants, see Sec. 1.3) is one of the most effective methods to solve the problem in Eq. (1.1.4). For example, PGD-AT is to alternatively run PGD on the inner max problem and run SGD on the outer minimization problem. Defense methods other than adversarial training, *e.g.,* certified defense, are discussed in Sec. 1.3.

Adding perturbed samples into the training dataset effectively improves robustness, but it still cannot lead to satisfactory performance. Up to February 2023, the state-of-the-art robust performance on CIFAR-10 dataset is 66.58%, given by Deepmind (Rebuffi et al., 2021). How to achieve satisfactory adversarial robustness is still an open problem, both theoretically and empirically.

One of the major challenges comes from generalization. It is found that training a model to fit perturbed training samples is relatively easy, but such a model does not perform well on the adversarial examples of the test set. For example, when applying ResNet to CIFAR-10, adversarial training can achieve nearly 100% robust accuracy on the training set, but it only gets 47% robust accuracy on the test set (Madry et al., 2017). Recent works (Gowal et al., 2020; Rebuffi et al., 2021) mitigated the overfitting issue, but it still has a 20% robust generalization gap between robust test accuracy and robust training accuracy. On the other hand, a standard-trained model on CIFAR-10

can generalize well to the test set within a 5% generalization gap. Therefore, it is essential to provide a theoretical understanding of adversarial generalization. In this thesis, we aim to study the following question:

Why adversarially robust generalization gap is large?

We study this question from the perspective of classical machine learning theory. We introduce the statistical learning theory framework in the next section.

1.2 Statistical Learning Theory Framework

1.2.1 Statistical Learning Theory Framework in Standard Settings

Consider the following setting of statistical learning. There is an unknown distribution $\mathcal{D}$ over examples from some space $\mathcal{Z}$. We receive a sample dataset $S = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_1, y_n)\} = \{z_1, \ldots, z_n\}$ of n examples drawn i.i.d. from $\mathcal{D}$. Let f be a model parameterized by w. Let ℓ be the loss function. The *population risk* is defined as:

$$R_{\mathcal{D}}(w) \stackrel{\text{def}}{=} \mathbb{E}_{z\sim\mathcal{D}}\, \ell(f_w(\boldsymbol{x}), y).$$

Since $\mathcal{D}$ is unknown, we minimize the *empirical risk*,

$$R_S(w) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} \ell(f_w(\boldsymbol{x}_i), y_i),$$

in practice. To simplify the notation, we use $g(w, z) = \ell(f_w(\boldsymbol{x}), y)$ in Chapter 3 and Chapter 4.

Risk Decomposition. Let w^* and $\bar{w}$ be the optimal solution of $R_{\mathcal{D}}(w)$ and $R_S(w)$ respectively. Then for the algorithm output $\hat{w} = A(S)$, the excess risk can be decom-

posed as

$$\begin{aligned} & R_{\mathcal{D}}(\hat{w}) - R_{\mathcal{D}}(w^*) \\ = & \underbrace{R_{\mathcal{D}}(\hat{w}) - R_S(\hat{w})}_{\mathcal{E}_{gen}} + \underbrace{R_S(\hat{w}) - R_S(\bar{w})}_{\mathcal{E}_{opt}} + \underbrace{R_S(\bar{w}) - R_S(w^*)}_{\leq 0} + \underbrace{R_S(w^*) - R_{\mathcal{D}}(w^*)}_{\mathbb{E}[\cdot]=0}. \end{aligned} \tag{1.2.1}$$

Therefore, the performance of a machine learning model, *i.e.,* test error $R_{\mathcal{D}}(\hat{w})$, can be decomposed into representation error, optimization error, and generalization error.

Representation. Given a function class $\mathcal{F} = \{f_w(\cdot) \mid w \in W\}$, representation power is the ability of $\mathcal{F}$ to fit the data distribution $\mathcal{D}$. $R_{\mathcal{D}}(w^*)$ is the representation error of machine learning models. If $\mathcal{F}$ has enough ability to fit the target function over the target distribution, $R_{\mathcal{D}}(w^*)$ is small.

Optimization. Given the empirical risk, optimization error is the gap between the loss of the algorithm's output and the loss of the global optimum. In optimization analysis, we hope the algorithm can find a near-optimal solution such that the optimization error $\mathcal{E}_{opt}$ is small.

Generalization. Generalization error $\mathcal{E}_{gen}$ is the gap between the performance of the output model $\hat{w}$ on the true distribution and on the sample dataset. We hope the machine learning model trained on the sample dataset S can generalize well to the unseen example in $\mathcal{D}$.

1.2.2 Statistical Learning Theory Framework in Adversarial Settings

Adversarial Loss. In adversarial settings, we consider the following surrogate loss

$$\tilde{\ell}(f_w(\boldsymbol{x}'), y) = \max_{\|\boldsymbol{x}-\boldsymbol{x}'\|_p \leq \epsilon} \ell(f_w(\boldsymbol{x}'), y), \tag{1.2.2}$$

where $\|\cdot\|_p$ is the ℓ_p-norm, $p \geq 1$. By replacing standard loss ℓ by adversarial loss $\tilde{\ell}$, the *robust population risk* and *robust empirical risk* are defined as:

$$\mathbb{E}_{z\sim\mathcal{D}} \max_{\|\boldsymbol{x}-\boldsymbol{x}'\|_p\leq\epsilon} \ell(f_w(\boldsymbol{x}), y) \quad \text{and} \quad \frac{1}{n}\sum_{i=1}^{n} \max_{\|\boldsymbol{x}-\boldsymbol{x}'\|_p\leq\epsilon} \ell(f_w(\boldsymbol{x}), y).$$

Then, the risk decomposition in Eq. (1.2.1) holds in adversarial settings. But the three subjects, representation, optimization, and generalization, have different meanings.

Adversarially Robust Representation. Given a function class $\mathcal{F} = \{f_w(\cdot) \mid w \in W\}$, robust representation power is the ability of $\mathcal{F}$ to fit the adversarial examples of data in $\mathcal{D}$.

Adversarially Robust Optimization. Robust optimization error is the gap between the loss of the algorithms (*e.g.,* adversarial training) output and the loss of the optimal saddle point or Nash equilibrium.

Adversarially Robust Generalization. Robust generalization error $\mathcal{E}_{gen}$ is the gap between the performance of the output model $\hat{w}$ on the adversarial examples of data from the true distribution and that from the sample dataset. We hope the machine learning model trained on the adversarial examples of dataset S can generalize well to the adversarial examples of unseen data.

From the observation in practice, adversarially robust generalization is the key issue of the poor performance of adversarial robustness.

In Chapter 2, we start with the uniform convergence analysis. Uniform convergence considers the generalization of the worst function in the function class

$$\sup_{f\in\mathcal{F}}[R_{\mathcal{D}}(f) - R_S(f)].$$

Uniform convergence analysis is algorithm-independent and focuses on the hypothesis function class. In Chapter 3 and 4, we take the algorithms into consideration. We

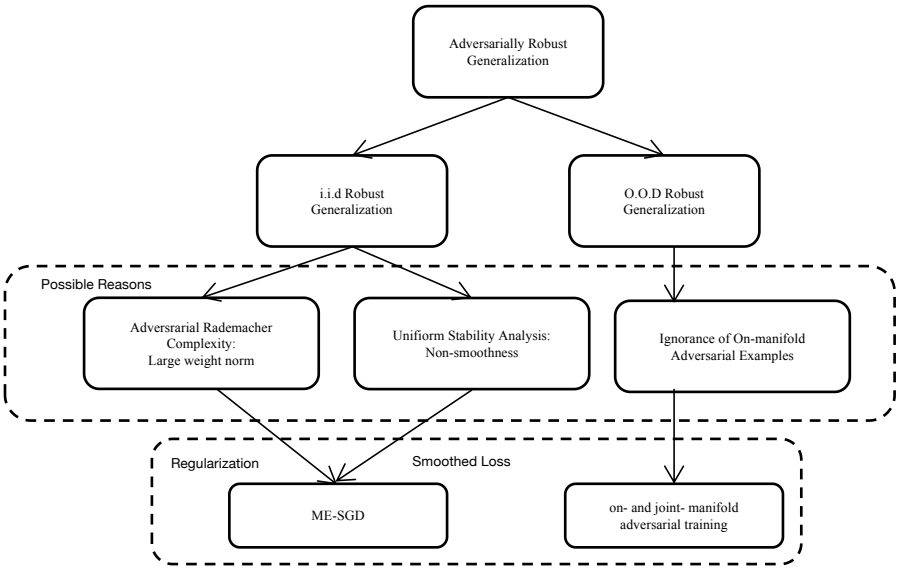


Figure 1.1: Overview of the thesis.

study algorithm-based generalization gaps,

$$R_{\mathcal{D}}(\hat{w}) - R_S(\hat{w}),$$

through the lens of uniform algorithmic stability.

In Chapter 5, we turn to the out-of-distribution (O.O.D) generalization in adversarial machine learning. We study a special type of adversarial examples we refer to as on-manifold adversarial examples. We study the following question: how well can an on-manifold adversarially-trained models generalize to off-manifold attacks, and vice versa? We provide both empirical and theoretical evidence to answer this question. The overview of the thesis is provided in Fig. 1.1.

1.3 Related Work

1.3.1 Adversarial Attack and Defense

Since 2013, it has now been well known that deep neural networks trained by standard gradient descent are highly susceptible to small corruptions to the input data. A lines of work aimed at finding more powerful attacks. Another works aimed at increasing the robustness of neural networks.

Adversarial Attack. Adversarial examples for deep neural networks were first introduced in (Szegedy et al., 2013). However, adversarial machine learning or robust machine learning has been studied for a long time (Biggio & Roli, 2018). In the setting of white box attack (Kurakin et al., 2016; Papernot et al., 2016; Moosavi-Dezfooli et al., 2016; Carlini & Wagner, 2017), the attackers have fully access to the model (weights, gradients, etc.). In black box attack (Chen et al., 2017; Su et al., 2019; Ilyas et al., 2018), the attackers have limited access to the model. First order optimization methods, which use the gradient information to craft adversarial examples, such as PGD (Madry et al., 2017), are widely used for white box attack. Zeroth-order optimization methods (Chen et al., 2017) are used in black box setting. (Li et al., 2019) improved the query efficiency in black-box attack. *Generative adversarial examples:* Generative models have been used to craft adversarial examples (Xiao et al., 2018; Song et al., 2018; Kos et al., 2018). The adversarial examples are more natural (Zhao et al., 2017).

Adversarial Defense. Training algorithms against adversarial attacks can be subdivided into the following categories. *Adversarial training:* The training data is augmented with adversarial examples to make the models more robust (Madry et al., 2017; Szegedy et al., 2013; Tramèr et al., 2017). *Preprocessing:* Inputs or hidden layers are quantized, projected onto different sets or other preprocessing methods (Buckman et al., 2018; Guo et al., 2017; Kabilan et al., 2018). *Stochasticity:* Inputs or hidden activations are randomized (Prakash et al., 2018; Dhillon et al., 2018; Xie et al., 2017). However, some of them are shown to be useless defenses given by obfuscated gradients

(Athalye et al., 2018). Adaptive attack (Tramer et al., 2020) is used for evaluating defenses to adversarial examples. A series of works study the certified robustness within the norm constraint around the original data. (Cohen et al., 2019) provides an analysis on certified robustness via random smoothing. (Lecuyer et al., 2019) studies certified robustness through the lens of differential privacy. Semi-supervised learning has been used to improve adversarial robustness (Carmon et al., 2019). Fast adversarial training (Wong et al., 2020) was introduced to save training time.

1.3.2 Theoretical Analysis

VC-Dimension Bounds. A classical approach in statistical learning is to use VC dimension to bound the generalization gap. It is thus natural to apply the VC-dim framework to adversarial setting, as (Cullina et al., 2018; Montasser et al., 2019; Attias et al., 2021) did. let $\mathcal{H}$ be the hypothesis class (e.g. the set of neural networks with a given architecture).

In the work of (Cullina et al., 2018), the authors defined adversarial VC-dim (AVC) and gave an bound on adversarial generalization gap with respect to $AVC(\mathcal{H})$. They provided an example: calculating $AVC(\mathcal{H})$ of half space classifiers. It is unclear how to calculate AVC of neural networks.

In the work of (Montasser et al., 2019), the authors defined the adversarial function class as $\mathcal{L}^{\mathcal{U}}_{\mathcal{H}}$, where $\mathcal{L}$ is the loss and $\mathcal{U}$ is the uncertainty set. They bound the adversarial generalization gap by $\mathcal{L}^{\mathcal{U}}_{\mathcal{H}}$, which is different from $AVC(\mathcal{H})$ of (Cullina et al., 2018). However, the authors did not provide a computable bound of as well.

In the work of (Attias et al., 2021), the authors assume that the perturbation set $U(\boldsymbol{x})$ is finite, i.e., for each sample $\boldsymbol{x}$, there are only k adversarial examples that can be chosen. They showed that the adversarial generalization gap can be bounded by

$$\mathcal{O}\left(\frac{1}{\varepsilon^2}(\sqrt{kVC(\mathcal{H})}\log(\frac{3}{2}+a)kVC(\mathcal{H}))+\log\frac{1}{\delta}\right).$$

Note that there is a computable bound of $VC(\mathcal{H})$, which is the number of parameters.

This bound is stronger than the previous two. However, this comes at a price: their bound depends on k, the number of allowed examined perturbed samples. This is a deviation from the original notion of adversarial generalization, where $U(\boldsymbol{x})$ is assumed to be an infinite set ($k \neq +\infty$).

Adversarial Rademacher Complexity. In classical learning theory, the generalization gap can be bounded in terms of Rademacher complexity with high probability. Similarly, robust generalization gap can be bounded in terms of adversarial Rademacher complexity. There have been two types of attempts to study adversarial Rademacher complexity of neural networks. *Linear and one-hidden layer cases.* The work of (Khim & Loh, 2018; Yin et al., 2019) both provided a bound for ARC in linear cases when they introduced ARC. The work of (Awasthi et al., 2020) analyzed the one-hidden layer cases and provided a bound for ARC in this case. These approaches seem hard to be extended to multi-layer cases. *Surrogate loss in multi-layer cases.* Standard method cannot be applied in multi-layer cases. Three work used surrogate loss $\hat{h}(\boldsymbol{x}, y) \approx \max_{\boldsymbol{x}'} h(\boldsymbol{x}', y)$ to bypass the difficulty and applied the layer peeling technique (Neyshabur et al., 2015) or covering number technique (Bartlett et al., 2017). Surrogate loss $\hat{h}(\boldsymbol{x}, y)$ includes tree-transformation loss (Khim & Loh, 2018), SDP relaxation loss (Yin et al., 2019), and FGSM loss (Gao & Wang, 2021). However, these approaches change the definition of ARC. It is not guaranteed to be a meaningful bound for robust generalization gap. To the best of our knowledge, how to bound the adversarial Rademacher complexity of deep neural networks has been an unsolved problem since it was raised in 2018.

Uniform Stability. Stability can be traced back to the work of (Rogers & Wagner, 1978). In statistical learning problems, it was well developed in analyzing the algorithm-based generalization bounds (Bousquet & Elisseeff, 2002). These bounds have been significantly improved in a recent sequence of works (Feldman & Vondrak, 2018, 2019). The work of (Chen et al., 2018) discussed the optimal trade-off between stability and convergence. (Bassily et al., 2020) studied the stability of SGD on non-smooth loss. They proved that the generalization bound contains a sample size-independent term.

The work of (Xing et al., 2021b; Xiao et al., 2022) showed that adversarial loss is non-smooth and SGDmax-based adversarial training algorithms will incur the generalization error floor.

Other generalization analysis. (Sinha et al., 2017) study the generalization of an adversarial training algorithm in terms of distributional robustness. The work of (Xing et al., 2021a,c; Javanmard et al., 2020) studied the generalization properties in the setting of linear regression. Gaussian mixture models are used to analyze adversarial generalization (Taheri et al., 2020; Javanmard et al., 2020; Dan et al., 2020). The work of (Allen-Zhu & Li, 2020) explains adversarial generalization through the lens of feature purification. These analyses proved that adversarial training has poor generalization ability in special settings or assumptions.

Other Theoretical Studies on Adversarial Examples. A series of works (Gilmer et al., 2018; Khoury & Hadfield-Menell, 2018) study the geometry of adversarial examples. The off-manifold assumption tells us that the adversarial examples leave the underlying data manifold (Szegedy et al., 2013). Pixeldefends (Song et al., 2017) uses a generative model to show that adversarial examples lie in a low probability region of the data distribution. The work of (Ma et al., 2018) uses Local Intrinsic Dimensionality (LID) to argue that the adversarial subspaces are of low probability and lie off the data submanifold. In Chapter 5, we revisit the off-manifold assumption and show the importance of on-manifold adversarial examples.

1.4 Main Contents

In Chapter 2, we study the adversarial Rademacher complexity (ARC) of deep neural networks to better understand adversarial generalization. Unlike the well-developed Rademacher complexity bounds in standard training, how to bound ARC in multi-layers cases is largely unclear due to the difficulty of analyzing adversarial loss in the definition of ARC. There have been two types of analysis of ARC. One is to provide

the upper bound of ARC in linear and one-hidden layer cases. However, this approach seems hard to extend to multi-layer cases. Another is to modify the adversarial loss and provide upper bounds of Rademacher complexity on such surrogate loss in multi-layer cases. However, such variants change the definition, are not guaranteed to be bounds for any meaningful robust generalization gaps. In this chapter, we provide a solution to this unsolved problem. Specifically, we provide the first bound of adversarial Rademacher complexity of deep neural networks. Our approach is based on covering numbers. We provide a method to handle the adversarial function classes of DNNs such that we can calculate the covering numbers. Finally, we provide experiments to study the empirical implication of our bounds and provide an analysis of poor adversarial generalization.

In Chapter 3, we study the robust overfitting issue of adversarial training by using tools from uniform stability. One major challenge is that the outer function (as a maximization of the inner function) is nonsmooth, so the standard technique (*e.g.,* (Hardt et al., 2016)) cannot be applied. Our approach is to consider η-approximate smoothness: we show that the outer function satisfies this modified smoothness assumption with η being a constant related to the adversarial perturbation ϵ. Based on this, we derive stability-based generalization bounds for stochastic gradient descent (SGD) on the general class of η-approximate smooth functions, which covers the adversarial loss. Our results suggest that robust test accuracy decreases in ϵ when T is large, with a speed between $\Omega(\epsilon\sqrt{T})$ and $\mathcal{O}(\epsilon T)$. This phenomenon is also observed in practice. Additionally, we show that a few popular techniques for adversarial training (*e.g.,* early stopping, cyclic learning rate, and stochastic weight averaging) are stability-promoting in theory.

We show that adversarial training can have nonvanishing generalization error even if the sample size n goes to infinity in Chapter 3. A natural question arises: can we eliminate the non-vanishing term? This paper provides an affirmative answer. We prove that the non-smooth loss minimization problem can achieve a generalization bound in $\mathcal{O}(T\alpha/n)$, which matches the bound in smooth cases. Additionally, our analysis can

be extended to weakly-convex cases, while existing uniform stability analysis on non-smooth loss requires a convexity assumption. The main idea is to use tools from Moreau envelopes, and the key technical ingredient to obtain the bound is an error bound on the worst-case change in the optimal solutions of Moreau envelopes. We refer to this variant of SGD as Moreau envelope-SGD. Combining with the optimization gap, we show that ME-SGD achieves the minimax lower bound. We conduct experiments on three non-smooth losses and show that ME-SGD reduces the generalization gap in practice.

In chapter 5, we study a different topic: on-manifold adversarial examples. One of the hypotheses of the existence of the adversarial examples is the off-manifold assumption: adversarial examples lie off the data manifold. However, recent research showed that on-manifold adversarial examples also exist. In this chapter, we revisit the off-manifold assumption and want to study a question: at what level is the poor performance of neural networks against adversarial attacks due to on-manifold adversarial examples? Since the true data manifold is unknown in practice, we consider two approximated on-manifold adversarial examples on both real and synthesis datasets. On real datasets, we show that on-manifold adversarial examples have greater attack rates than off-manifold adversarial examples on both standard-trained and adversarially-trained models. On synthetic datasets, theoretically, We prove that on-manifold adversarial examples are powerful, yet adversarial training focuses on off-manifold directions and ignores the on-manifold adversarial examples. Furthermore, we provide analysis to show that the properties derived theoretically can also be observed in practice. Our analysis suggests that on-manifold adversarial examples are important, and we should pay more attention to on-manifold adversarial examples for training robust models.

1.5 Itemized Summary of Contributions

Chapter 2 provides an analysis of adversarial Rademacher complexity of deep neural networks. The contribution are listed as follows:

1. How to bound ARC is an unsolved problem raised in 2018 with several attempts in recent years. We give the first upper bound. The bound provides a further understanding of the robust generalization of DNNs.
2. Technical contribution: Our approach is based on covering numbers. We provide an adversarial perturbation bound such that we can calculate the covering number of adversarial function classes. This technique might be helpful in another problem.
3. We conduct experiments on CIFAR-10 and CIFAR-100 to study the relationship between the factors in our bounds and the robust generalization gap.

The code of Chapter 2 is available at

`https://github.com/JiancongXiao/Adversarial-Rademacher-Complexity`.

Chapter 3 provides a uniform stability analysis of adversarial training and gives a possible explanation of robust overfitting from a new perspective. The contributions are listed as follows:

1. Main results: we derive stability-based generalization bounds for adversarial training using the notion of η-approximate smoothness. Based on this, we provide an analysis to understand robust overfitting.
2. We provide the stability analysis of a few popular techniques for adversarial training and show that they are indeed stability-promoting.
3. We provide experiments on SVHN, CIFAR-10, CIFAR-100, and ImageNet. The results verify the generalization bounds.
4. Technical contribution: we develop a set of properties of η-approximately smooth function, which might be useful in other tasks.

The code of Chapter 3 is available at

`https://github.com/JiancongXiao/Stability-of-Adversarial-Training`.

Chapter 4 proposes an algorithm, ME-SGD, to eliminate the generalization error floor in non-smooth minimization problems. The contributions are listed as follows:

1. Main result: we prove $\mathcal{O}(T\alpha/n)$ uniform stability in non-smooth loss minimization problems, which improves the bound induced by SGD in $\mathcal{O}(\sqrt{T}\alpha + T\alpha/n)$, matches the bound in smooth cases. The minimax lower bound of the excess risk is achieved by combining the generalization and optimization error.
2. Technical contribution: The main idea is to use the Moreau envelopes to smooth the loss function. The key ingredient is an error bound on the worst-case change in the optimal solutions of Moreau envelopes when a single data point in the dataset is replaced.
3. We extend our analysis to weakly-convex cases. Notice that the existing uniform stability analysis of non-smooth loss requires a convexity assumption.
4. Experiments on three non-smooth losses verify the theoretical results and show the effectiveness of ME-SGD in practice.

The code of Chapter 4 is available at

`https://github.com/JiancongXiao/Moreau-Envelope-SGD`.

Chapter 5 considers the adversarial robustness of DNNs against on-manifold adversarial examples. The contributions are listed as follows:

1. We develop two approximate on-manifold adversarial attacks to take a closer look at on-manifold adversarial examples.
2. We provide comprehensive analyses of on-manifold adversarial examples empirically and theoretically. We summarize our main results in Table 5.1. Our results suggest the importance of on-manifold adversarial examples, and we should pay more attention to on-manifold adversarial examples to train robust models.
3. Technical contributions: our main technical contribution is providing the closed-form solutions to the min-max problems of on-manifold adversarial training (Thm. 5.3 and 5.4) in GMMs. We also provide the upper and lower bounds of excess risk (Thm. 5.1 and 5.2) of on-manifold attacks.

The code of Chapter 5 is available at

`https://github.com/JiancongXiao/On-manifold-adversarial-examples`.

□ **End of chapter.**

Chapter 2

Adversarial Rademacher Complexity

Summary

To better understand adversarial generalization, we study the adversarial Rademacher complexity (ARC) of deep neural networks. Unlike the well-developed Rademacher complexity bounds in standard training, how to bound ARC in multi-layers cases is largely unclear due to the difficulty of analyzing adversarial loss in the definition of ARC. There have been two types of analysis of ARC. One is to provide the upper bound of ARC in linear and one-hidden layer cases. However, these approaches seem hard to extend to multi-layer cases. Another is to modify the adversarial loss and provide upper bounds of Rademacher complexity on such surrogate loss in multi-layer cases. However, such variants of Rademacher complexity are not guaranteed to be bounds for meaningful robust generalization gaps. In this chapter, we provide a solution to this unsolved problem. Specifically, we provide the first bound of adversarial Rademacher complexity of deep neural networks. Our approach is based on covering numbers. We provide a method to handle the adversarial function classes of DNNs such that we can calculate the covering numbers. Finally, we provide experiments to study the empirical implication of our bounds and provide an analysis of poor adversarial generalization.

2.1 Introduction

In classical learning theory, the generalization gap can be bounded in terms of Rademacher complexity with high probability. (Empirical) Rademacher complexity is defined as

$$\mathcal{R}_{\mathcal{S}}(\mathcal{H}) = \mathbb{E}_{\sigma}\frac{1}{n}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i h(\boldsymbol{x}_i, y_i)\right], \tag{2.1.1}$$

where $\mathcal{S} = \{\boldsymbol{x}_i, y_i\}_{i=1,\cdots,n}$ is the sample dataset, $\mathcal{H}$ is the hypothesis function class, and σ_i are i.i.d. Rademacher random variable, *i.e.,* σ_i equals to 1 or -1 with equality probability. Moreover, Rademacher complexity of deep neural nets has been studied recently and is well-developed. Neyshabur et al. (2015) used a layer-peeling technique to provide a bound: for depth-l neural nets, assume that the weight matrices $W_1, W_2, \cdots, W_l$ in each of the l layers have Frobenius norms bounded by $M_1, \cdots, M_l$, and all n input instances have ℓ_2-norm bounded by B, the generalization gap between population risk and empirical risk is bounded by $\mathcal{O}(B2^l\prod_{j=1}^{l}M_j/\sqrt{n})$ with high probability. Bartlett et al. (2017) used a covering number technique to provide a spectral norm bound for Rademacher complexity.

The work of (Khim & Loh, 2018; Yin et al., 2019) concurrently extended Rademacher complexity to adversarial settings to study the causes of poor adversarial generalization. They showed that robust generalization gap (in misclassification error) is bounded by *adversarial Rademacher complexity* (ARC), which is defined as

$$\mathcal{R}_{\mathcal{S}}(\tilde{\mathcal{H}}) = \mathbb{E}_{\sigma}\frac{1}{n}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i \max_{\boldsymbol{x}_i'\in\mathbb{B}(\boldsymbol{x}_i,\epsilon)} h(\boldsymbol{x}_i', y_i)\right], \tag{2.1.2}$$

with high probability, where $\mathbb{B}(\boldsymbol{x}, \epsilon)$ is a norm ball around sample $\boldsymbol{x}$ with radius ϵ.

However, The work of (Khim & Loh, 2018; Yin et al., 2019) both pointed out that it is difficult to provide a bound for ARC in deep neural networks cases due to the max operation in Eq. (2.1.2). Existing methods, e.g., layer peeling (Neyshabur et al.,

Table 2.1: Comparison of our work with the two types of attempts on adversarial Rademacher complexity. Type 1: Thm. 2 in (Yin et al., 2019) , Lemma 1 & Corollary 1 in (Khim & Loh, 2018), and the work of (Awasthi et al., 2020). Type 2: Thm. 8 in (Yin et al., 2019) , Lemma 2 & Corollary 2 in (Khim & Loh, 2018) , and the work of (Gao & Wang, 2021). We provide the first bound for adversarial Rademacher complexity of DNNs, resulting the first bound for robust generalization gap of DNNs.

	Setting	Techniques	Limitation	Bound RGG of DNNs?
Type 1	$\leq$ 2 Layers	Optimal Attack	only$\leq$2 layer	✗
Type 2	Multi-layer	Surrogate Loss	Not bounds for ARC	✗
Ours	Multi-layer	in Sec. 2.3.1	N/A	✓

2015) and covering number (Bartlett et al., 2017), cannot be directly applied. There have been two types of attempts to study adversarial Rademacher complexity of neural networks.

Linear and one-hidden layer cases. The work of (Khim & Loh, 2018; Yin et al., 2019) both provided a bound for ARC in linear cases when they introduced ARC. The work of (Awasthi et al., 2020) analyzed the max problem in Eq. (2.1.2) in one-hidden layer cases and provided a bound for ARC in this case. These approaches seem hard to be extended to multi-layer cases.

Surrogate loss in multi-layer cases. Standard method cannot be applied in multi-layer cases. Three work used surrogate loss $\hat{h}(\boldsymbol{x}, y) \approx \max_{\boldsymbol{x}'} h(\boldsymbol{x}', y)$ to bypass the difficulty and applied the layer peeling technique (Neyshabur et al., 2015) or covering number technique (Bartlett et al., 2017). Surrogate loss $\hat{h}(\boldsymbol{x}, y)$ includes tree-transformation loss (Khim & Loh, 2018), SDP relaxation loss (Yin et al., 2019), and FGSM loss (Gao & Wang, 2021). However, these approaches change the definition of ARC. It is not guaranteed to be a meaningful bound for robust generalization gap.

To the best of our knowledge, how to bound the adversarial Rademacher complexity of deep neural networks has been an unsolved problem since it was raised in 2018. In this chapter, we provide the first bound for adversarial Rademacher complexity of deep neural networks. Our approach is still based on covering number. Then, the main

difficulty becomes how to calculate the covering number of adversarial function classes. We provide a novel adversarial perturbation bound such that we can calculate the covering number of adversarial function classes.

Specifically, for depth-l, width-h fully-connected neural nets, with high probability,

$$\text{Adversarial Rademacher Complexity} \leq \mathcal{O}\left(\frac{(B+\epsilon)h\sqrt{l\log l}\prod_{j=1}^{l}M_j}{\sqrt{n}}\right).$$

We provide a comparison with the existing bounds in similar settings. We show that our bound is comparable to 1) the upper bound for standard Rademacher complexity and 2) the upper bound for ARC in one-hidden layer cases. We also provide a lower bound for ARC correspondingly.

Finally, we trained 88 models to study some empirical implications of our bounds. We find that the weight norm positively relates to the adversarial generalization gap in our experiments.

2.2 Preliminaries

2.2.1 Generalization Gap and Rademacher Complexity

Rademacher Complexity. A classical measure of the generalization gap is Rademacher complexity (Bartlett & Mendelson, 2002). Given the hypothesis class $\mathcal{H}$, the (empirical) Rademacher complexity is defined as

$$\mathcal{R}_{\mathcal{S}}(\mathcal{H}) = \mathbb{E}_{\sigma}\frac{1}{n}\left[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i h(\boldsymbol{x}_i, y_i)\right],$$

where σ_i are i.i.d Rademacher random variables, i.e. σ_i equals to 1 or -1 with equal probability. Define the function class $\ell_{\mathcal{F}} = \{\ell(f(\boldsymbol{x}), y) | f \in \mathcal{F}\}$, then we have the following generalization bound.

Proposition 2.1 ((Bartlett & Mendelson, 2002))**.** *Suppose that the range of the loss function $\ell(f(\boldsymbol{x}), y)$ is $[0, C]$. Then, for any $\delta \in (0, 1)$, with probability at least $1-\delta$,*

the following holds for all $f \in \mathcal{F}$,

$$L_{\mathcal{D}}(f) \leq L_S(f) + 2\mathcal{R}_{\mathcal{S}}(\ell_{\mathcal{F}}) + 3C\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

2.2.2 Robust Generalization Gap and Adversarial Rademacher Complexity

We consider general ℓ_p attacks for $p \geq 1$. The robust generalization gap is defined as follow:

$$\text{Robust Generalization Gap} := R_{\mathcal{D}}(f) - R_S(f).$$

Let the adversarial hypothesis class be $\tilde{\ell}_{\mathcal{F}} = \{\tilde{\ell}(f(\boldsymbol{x}), y) | f \in \mathcal{F}\}$, according to Proposition 2.1, the robust generalization gap can be bounded by the Rademacher complexity of $\tilde{\ell}_{\mathcal{F}}$. We have the following adversarial generalization bound.

Proposition 2.2 ((Yin et al., 2019))**.** *Suppose that the range of the loss function* $\tilde{\ell}(f(\boldsymbol{x}), y)$ *is* $[0, C]$*. Then, for any* $\delta \in (0, 1)$*, with probability at least* $1 - \delta$*, the following holds for all* $f \in \mathcal{F}$,

$$R_{\mathcal{D}}(f) \leq R_S(f) + 2\mathcal{R}_{\mathcal{S}}(\tilde{\ell}_{\mathcal{F}}) + 3C\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

Binary Classification. We first discuss the binary classification case, then we discuss the extension to the multi-class classification case in Section 2.5. Following (Yin et al., 2019; Awasthi et al., 2020), we assume that the loss function can be written as $\ell(f(\boldsymbol{x}), y) = \phi(yf(\boldsymbol{x}))$ where ϕ is a non-increasing function. Then

$$\max_{\boldsymbol{x}'} \ell(f(\boldsymbol{x}'), y) = \phi(\min_{\boldsymbol{x}'} yf(\boldsymbol{x}')).$$

Assume that the function ϕ is L_ϕ-Lipschitz, by Talagrand' s Lemma (Ledoux & Talagrand, 2013), we have $\mathcal{R}_{\mathcal{S}}(\tilde{\ell}_{\mathcal{F}}) \leq L_\phi \mathcal{R}_{\mathcal{S}}(\tilde{\mathcal{F}})$, where we define the adversarial function

class as

$$\tilde{\mathcal{F}} = \{\tilde{f} : (\boldsymbol{x}, y) \to \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\|_p \leq \epsilon} yf(\boldsymbol{x}') | f \in \mathcal{F}\}. \tag{2.2.1}$$

Despite of the loss function $\ell()$, ARC can also be defined in the following form.

Definition 2.1 (Adversarial Rademacher Complexity)**.** *We define the Rademacher complexity of the adversarial function class $\tilde{\mathcal{F}}$ in Eq. (2.2.1), i.e.*

$$\mathcal{R}_S(\tilde{\mathcal{F}}) = \mathbb{E}_\sigma \frac{1}{n}\left[\sup_{f\in\mathcal{F}} \sum_{i=1}^{n} \sigma_i \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\|_p \leq \epsilon} yf(\boldsymbol{x}')\right],$$

as adversarial Rademacher complexity.

Our goal is to give upper bounds for adversarial Rademacher complexity. Then, it induces the guarantee of the robust generalization gap.

Hypothesis Class. We consider depth-l, width-h fully-connected neural networks,

$$\mathcal{F} = \{\boldsymbol{x} \to W_l\rho(W_{l-1}\rho(\cdots\rho(W_1\boldsymbol{x})\cdots)), \|W_j\| \leq M_j, j = 1, \cdots, l\}, \tag{2.2.2}$$

where $\rho(\cdot)$ is an element-wise L_ρ-Lipschitz activation function, W_j are $h_j \times h_{j-1}$ matrices, for $j = 1, \cdots, l$. We have $h_l = 1$ and $h_0 = d$ is the input dimension. Let $h = \max\{h_0, \cdots, h_l\}$ be the width of the neural networks. Denote the (a, b)-group norm $\|W\|_{a,b}$ as the a-norm of the b-norm of the rows of W. We consider two cases, Frobenius norm and $\|\cdot\|_{1,\infty}$-norm in equation (2.2.2). Additionally, we assume that $\|\boldsymbol{X}\|_{p,\infty} = B$, where $\boldsymbol{X}$ is a matrix of the all the samples, the i^{th} rows of $\boldsymbol{X}$ is $\boldsymbol{x}_i^T$.

2.3 Challenge of Bounding Adversarial Rademacher Complexity

2.3.1 Layer Peeling

To use the layer peeling techniques, Rademacher complexity should be in the structure of composition functions. However, the max operation in the definition of ARC destroys

this structure. For more detail, see Sec. 2.9.

Recent Attempts. Khim & Loh (2018) introduced a surrogate loss, tree-transformation loss, to get rid of the max operation. Then, the layer peeling bound can be applied to adversarial settings.

Covering Number

In this part, we discuss the difficulty of calculating the covering number of a robustified function class, and how we resolve it. For simplicity, We use the 1-dimensional function as an example.

We first state the problem for scalar functions. Consider $f_w(x) : \mathbb{R} \to \mathbb{R}$ for scalars $|w| \leq M$, and its robustified function $g_w(x) \triangleq \min_{x' \in [x-\epsilon, x+\epsilon]} f_w(x')$. Suppose we have only one sample $S = \{x\}$ with $|x| = B$. Suppose $f_w(x)$ is L-Lipschitz w.r.t. w in x, e.g., $wx, w^2x^8, \sin(w)x^5$. Note that $g_w(x)$ may not be Lipschitz continuous w.r.t. w. The problem is the following.

Math Problem 1: Bound the size of an ζ-cover $\mathcal{N}(\tilde{\mathcal{F}}, \|\cdot\|_S, \zeta)$ of the robustified function class $\tilde{\mathcal{F}} = \{g(x) \triangleq \min_{x' \in [x-\epsilon, x+\epsilon]} f_w(\cdot) : |w| \leq M\}$ given sample $S = \{x\}$. Here, the ζ-cover is a set of functions whose distance to any function in F is no more than ζ.

Review: Covering Number of Function Space. To help readers better understand the problem, we consider a simpler problem of standard function class $\mathcal{F} = \{f_w(\cdot) : \mathbb{R} \to \mathbb{R} \mid |w| \leq M\}$.

Math Problem 2: Bound the size of an ζ-cover $\mathcal{N}(\mathcal{F}, \|\cdot\|_S, \zeta)$ of the function space $F = \{f_w(\cdot) : \mathbb{R} \to \mathbb{R} \mid |w| \leq M\}$ given sample $S = \{x\}$.

The idea is to build a relation between a cover of function class and a cover of the parameter region (a subset of Euclidean space). For this purpose, we only need to relate the distance in the parameter space to the distance in the function space. More specifically, suppose $|w_1 - w_2| \leq \epsilon_w$, then $|f_{w_1}(x) - f_{w_2}(x)| \leq L|w_1 - w_2| \leq \epsilon_w L$. As

a result, $|w_1 - w_2| \leq \epsilon_w = \zeta/L \Rightarrow |f_{w_1}(x) - f_{w_1}(x)| \leq \zeta$. This implies an ϵ-cover of $[-M, M]$ leads to an ζ-cover of $\mathcal{F}$. In the rest of the paper, we refer to $|f_{w_1}(x) - f_{w_1}(x)|$ as weight perturbation.

The proof sketch (of linear function $f_w(\cdot) = wx$) of problem 2 is the following.

Step 1: Calculate the diameter of $\mathcal{F}$. $|f_w(x)| = 2|wx| = 2MB \triangleq D$. (This step is not necessary in linear case. However, it can simplify the notation in high dimension and multi-layer cases.)

Step 2: Relate the distance in the parameter space to the distance in the function space. More specifically, $|w_1 - w_2| \leq \epsilon_w$, then $|f_{w_1}(x) - f_{w_1}(x)| = |w_1 x - w_2 x| = |w_1 - w_2||x| \leq \epsilon_w B$. As a result, $|w_1 - w_2| \leq \epsilon_w \triangleq \zeta/B \Rightarrow |f_{w_1}(x) - f_{w_2}(x)| \leq \zeta$. This implies an ζ/B-cover of $[-M, M]$ (region in w-space) leads to an ζ-cover of F.

Step 3: Bound the size of the ϵ_w-cover of $[-M, M]$. In fact, $\{M, M-2\epsilon_w, M-4\epsilon_w, \dots\}$ is one such cover, with size no more than $2M/(2\epsilon_w) = M/\epsilon_w$.

Conclusion: The ζ-cover of $\mathcal{F}$ is no more than $M/\epsilon_w = MB/\zeta$. For generalization nonlinear L-Lipschitz function, The ζ-cover of $\mathcal{F}$ is no more than $M/\epsilon_w = ML/\zeta$.

Discussing Problem 1 on Robustified Function Space. Now we come back to Problem 1. We can adopt the same proof procedure for Problem 2: Step 1 and Step 3 would be the same (or small modification), but Step 2 needs to significantly modified. In the prove of Problem 2, we can see that the estimation of L is is the key ingredient ($L = B$ in linear case). However, the robustified function $g_w(x)$ might not be Lipschitz continous. A small change in the input x might cause a large change in the function value. The following naive method can illustrate the challenge.

Naive Method for Linear f_w Fails. Assume $f_w(x) = wx$. Let

$$x_i^* = \inf_{\|x - x'\| \leq \epsilon} f_{w_i}(x'), i = 1, 2.$$

A naive method is to use triangle inequality to get $|f_{w_1}(x_1^*) - f_{w_2}(x_2^*)| = |w_1 x_1^* - w_2 x_2^*| \leq |(w_1 - w_2)x_1^*| + |w_2(x_1^* - x_2^*)| \leq \epsilon_w B + 2M\epsilon \overset{\text{want}}{=} \zeta$. Thus an ζ-cover of the function

space can be built from an $((\zeta - 2M\epsilon)/B)$-cover in w-space. This is only possible when $\zeta > M\epsilon$, thus we cannot build an ζ-cover for arbitrarily small $\zeta > 0$. As a result, as $n \to \infty$, the corresponding adversarial Rademacher complexity bound would have a non-vanishing term, which is undesirable.

Challenge. The key issue seems to be controlling the extra term $|w_2(x_1^* - x_2^*)|$. For a linear function f_w, this can be resolved by noting $x_1^* = x_2^*$ when $|w_1| > \epsilon_w$. However, for general f_w (*e.g., DNNs*), the relation of the worst-perturbed points x_1^* and x_2^* is unclear.

Recent Attempts. Yin et al. (2019) introduced a SDP-relaxation loss. Then, ARC $\approx$ RC defined in SDP-relaxation loss. They directly applied the standard weight perturbation bound provided in the work of (Bartlett et al., 2017). Gao & Wang (2021) considered FGSM loss. Then, the weight perturbation can be bounded in terms of $|f_{w_1}(x) - f_{w_2}(x)|$ and $|\nabla f_{w_1}(x) - \nabla f_{w_2}(x)|$. Then, they provide a bound for RC defined in FGSM loss. However, these two approaches change the definition of ARC.

2.3.2 Our approach

From the previous discussion, we can see that Problem 1 can be solved if we have the following robustified weight perturbation bound.

Assumption 1: If $|w_1 - w_2| \leq \epsilon_w$, then $|g_{w_1}(x) - g_{w_2}(x)| \leq \epsilon_w \psi(B)$.

Under Assumption 1, the size of ζ-cover of $\tilde{\mathcal{F}}$ is no more than $M\psi(B)/\zeta$. The remaining problem is to prove Assumption 1 holds, namely, solve the following math problem.

Math Problem 3: Find some function $\psi(B) : \mathbb{R} \to \mathbb{R}$, such that if $|w_1 - w_2| \leq \epsilon_w$, then $|g_{w_1}(x) - g_{w_2}(x)| \leq \epsilon_w \psi(B)$, where $g_w(x) \triangleq \min_{x' \in [x-\epsilon, x+\epsilon]} f_w(x)$.

Our solution to Problem 3 is based on an intermediate adversarial example. Let

$$x_1^* = \arg \inf_{\|x - x'\| \leq \epsilon} w_1 x', \quad \text{and} \quad x_2^* = \arg \inf_{\|x - x'\| \leq \epsilon} w_2 x',$$

and

$$\bar{x} = \begin{cases} x_2^* & if \quad w_1 x_1^* \geq w_2 x_2^* \\ x_1^* & if \quad w_1 x_1^* < w_2 x_2^*. \end{cases}$$

Then, we have

$$|w_1 x_1^* - w_2 x_2^*| \leq |w_1 \bar{x} - w_2 \bar{x}| \leq |w_1 - w_2||\bar{x}| \leq \epsilon_w (B + \epsilon).$$

Therefore, $\psi(B) = B + \epsilon$. □

Extension to General Cases. Firstly, the definition of the intermediate adversarial examples can be extended to multi-layer cases. Secondly, for general dataset S with n samples, we can define the intermediate adversarial examples of these n samples independently. Therefore, we obtain the proof in the next sectiion.

Assumption 2: If $|f_{w_1}(x) - f_{w_2}(x)| \leq \epsilon_w \phi(B)$, then $\psi(B) = \phi(B + \epsilon)$.

Assumption 2 holds in linear case, where $\phi(B) = B$ and $\psi(B) = B + \epsilon$. In the next section, we will see that Assumption 2 holds in DNNs cases. It shows that the covering number bound of robustified function class is comparable to the bound of standard function class, differing in a factor of ϵ, resulting a comparable bound for ARC.

2.4 Bounds of Adversarial Rademacher Complexity

2.4.1 Main Result on Adversarial Rademacher Complexity and Proof Sketch

Our main result states an upper bound of adversarial Rademacher complexity in Frobenius norm.

Theorem 2.1 (Frobenius Norm Bound)**.** *Given the function class $\mathcal{F}$ in equation (2.2.2) under Frobenius Norm, and the corresponding adversarial function class $\tilde{\mathcal{F}}$ in equation (2.2.1). The adversarial Rademacher complexity of deep neural networks $\mathcal{R}_S(\tilde{\mathcal{F}})$ satis-*

fies

$$\mathcal{R}_S(\tilde{\mathcal{F}}) \leq \frac{24}{\sqrt{n}} \max\{1, q^{\frac{1}{2}-\frac{1}{p}}\}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon) L_\rho^{l-1} \sqrt{\sum_{j=1}^{l} h_j h_{j-1} \log(3l)} \prod_{j=1}^{l} M_j. \tag{2.4.1}$$

By assuming that $L_\rho = 1$, $p \leq 2$, $\|\boldsymbol{X}\|_{p,\infty} = B$, *and* $h = \max\{h_0, \cdots, h_l\}$, *we have*

$$\mathcal{R}_S(\tilde{\mathcal{F}}) \leq \mathcal{O}\left(\frac{(B+\epsilon) h \sqrt{l \log(l)} \prod_{j=1}^{l} M_j}{\sqrt{n}} \right). \tag{2.4.2}$$

Our proof is based on calculating the covering number of $\tilde{\mathcal{F}}$. Below we sketch the proof. The completed proof is provided in Sec. 2.8.

Step 1: Diameter of $\tilde{\mathcal{F}}$. We first calculate the diameter of $\tilde{\mathcal{F}}$. We have

$$2 \max_{\tilde{f} \in \tilde{\mathcal{F}}} \|\tilde{f}\|_S \leq 2 L_\rho^{l-1} \max\{1, q^{\frac{1}{2}-\frac{1}{p}}\}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon) \prod_{j=1}^{l} M_j \triangleq D.$$

Step 2: Distance to $\tilde{\mathcal{F}}^c$. Let $\mathcal{C}_j$ be δ_j-covers (Definition 2.2) of $\{\|W_j\|_F \leq M_j\}$, $j = 1, 2, \cdots, l$. Let

$$\mathcal{F}^c = \{f^c : \boldsymbol{x} \to W_j^c \rho(W_{j-1}^c \rho(\cdots \rho(W_1^c \boldsymbol{x}) \cdots)), W_j^c \in \mathcal{C}_j, j = 1, 2 \cdots, l\}$$

$$\text{and } \tilde{\mathcal{F}}^c = \{\tilde{f} : (\boldsymbol{x}, y) \to \inf_{\|\boldsymbol{x} - \boldsymbol{x}'\|_p \leq \epsilon} y f(\boldsymbol{x}') | f \in \mathcal{F}^c\}.$$

For all $\tilde{f} \in \tilde{\mathcal{F}}$, we need to find the smallest distance to $\tilde{\mathcal{F}}^c$, i.e. we need to calculate the

$$\max_{\tilde{f} \in \tilde{\mathcal{F}}} \min_{\tilde{f}^c \in \tilde{\mathcal{F}}^c} \|\tilde{f} - \tilde{f}^c\|_S.$$

Let

$$x_i^* = \arg \inf_{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_p} y_i f(\boldsymbol{x}_i'), \quad \text{and} \quad x_i^c = \arg \inf_{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_p} y_i f^c(\boldsymbol{x}_i'),$$

and

$$\bar{\boldsymbol{x}}_i = \begin{cases} \boldsymbol{x}_i^c & if \quad f(\boldsymbol{x}_i^*) \geq f^c(\boldsymbol{x}_i^c) \\ \boldsymbol{x}_i^* & if \quad f(\boldsymbol{x}_i^*) < f^c(\boldsymbol{x}_i^c). \end{cases}$$

Define $g_b^a(\cdot)$ as

$$g_b^a(\bar{\boldsymbol{x}}) = W_b\rho(W_{b-1}\rho(\cdots W_{a+1}\rho(W_a^c\cdots\rho(W_1^c\bar{\boldsymbol{x}})\cdots))).$$

In words, for the layers $b \geq j > a$ in $g_b^a(\cdot)$, the weight is W_j, for the layers $a \geq j \geq 1$ in $g_b^a(\cdot)$, the weight is W_j^c. Then we have $f(\bar{\boldsymbol{x}}_i) = g_l^0(\bar{\boldsymbol{x}}_i)$, $f(\bar{\boldsymbol{x}}_i) = g_l^l(\bar{\boldsymbol{x}}_i)$. We can decompose

$$|f(\bar{\boldsymbol{x}}_i) - f^c(\bar{\boldsymbol{x}}_i)| \leq |g_l^0(\bar{\boldsymbol{x}}_i) - g_l^1(\bar{\boldsymbol{x}}_i)| + \cdots + |g_l^{l-1}(\bar{\boldsymbol{x}}_i) - g_l^l(\bar{\boldsymbol{x}}_i)| \leq \sum_{j=1}^{l} \frac{D\delta_j}{2M_j}. \tag{2.4.3}$$

Step 3: Covering Number of $\tilde{\mathcal{F}}$. Then, we can calculate the ε-covering number $\mathcal{N}(\tilde{\mathcal{F}}, \|\cdot\|_S, \varepsilon)$. Because $\tilde{\mathcal{F}}^c$ is a ε-cover of $\tilde{\mathcal{F}}$. The cardinality of $\tilde{\mathcal{F}}^c$ is

$$\mathcal{N}(\tilde{\mathcal{F}}, \|\cdot\|_S, \varepsilon) = |\tilde{\mathcal{F}}^c| \leq (\frac{3lD}{2\varepsilon})^{\sum_{j=1}^{l} h_j h_{j-1}}. \tag{2.4.4}$$

Step 4: Integration. By Dudley's integral, we obtain the bound in Theorem 2.1.

2.4.2 Further Discussions of the Bounds

Theorem 2.2 ($\|\cdot\|_{1,\infty}$-Norm Bound)**.** *Given the function class $\mathcal{F}$ in equation (2.2.2) under $\|\cdot\|_{1,\infty}$-norm, and the corresponding adversarial function class $\tilde{\mathcal{F}}$ in equation (2.2.1). The adversarial Rademacher complexity of deep neural networks $\mathcal{R}_S(\tilde{\mathcal{F}})$ satisfies*

$$\mathcal{R}_S(\tilde{\mathcal{F}}) \leq \frac{24}{\sqrt{n}}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon)L_\rho^{l-1}\sqrt{\sum_{j=1}^{l} h_j h_{j-1}\log(3l)}\prod_{j=1}^{l} M_j. \tag{2.4.5}$$

The proof is similar to the proof of Theorem 2.1 and is defered to Sec. 2.8. In the case of $\|\cdot\|_{1,\infty}$-norm, the bound is similar to the bound in the Frobenius norm case

except the term $\max\{1, q^{1/2-1/p}\}$. Therefore, for all $p \geq 1$, the $\|\cdot\|_{1,\infty}$-norm bound have the same order in Eq. (2.4.2). Similarly, we can obtain a spectral norm bound in Eq. (2.4.2) if M_j is the bound of the spectral norm of W_j and h_j is the rank in each of the layers j.

We compare our bound to the bounds in similar settings. Specifically, we compare our bound with the covering number bounds for (standard) Rademacher complexity (Bartlett et al., 2017) and the bound of ARC in two-layer cases.

Covering Number Bound for (Standard) RC. The work of (Bartlett et al., 2017) used a covering numbers argument to show that the generalization gap is bounded by

$$\tilde{\mathcal{O}}\left(\frac{B\prod_{j=1}^{l}\|W_j\|}{\sqrt{n}}\left(\sum_{j=1}^{l}\frac{\|W_j\|_{2,1}^{2/3}}{\|W_j\|^{2/3}}\right)^{3/2}\right).$$

As discussed in the work of (Neyshabur et al., 2017b) and (Golowich et al., 2018), this bound is no smaller than $\tilde{\mathcal{O}}(B\sqrt{l^3 h}\prod_{j=1}^{l} M_j/\sqrt{n})$. Our bound has an additional dependence on ϵ, which is unavoidable in adversarial settings, and has a different dependence on the network size $\mathcal{O}(\sqrt{l\log(l)h})$. We have a lower dependence on depth-l and a higher dependence on width-h.

Bound for ARC in Two-layers Cases. The work of (Awasthi et al., 2020) showed that the ARC is bounded by

$$\mathcal{O}\left(\frac{(B+\epsilon)\sqrt{h_1 d}\sqrt{\log n}M_1 M_2}{\sqrt{n}}\right) \tag{2.4.6}$$

in two-layers cases. If we apply our bound to two-layers cases, our bound becomes $\mathcal{O}((B+\epsilon)\sqrt{h_1 d}M_1 M_2/\sqrt{n})$, which is strictly better than the bound in Eq. (2.4.6). If other factors remain the same, our bound has a lower dependence on the sample size n.

Theorem 2.3 (Lower Bound)**.** *Given the function class of DNNs $\mathcal{F}$ in equation (2.2.2) with Relu activation functions, and the corresponding adversarial function class $\tilde{\mathcal{F}}$ in*

equation (2.2.1). There exists a dataset S, s.t. the adversarial Rademacher complexity of deep neural networks $\mathcal{R}_S(\tilde{\mathcal{F}})$ satisfies

$$\mathcal{R}_S(\tilde{\mathcal{F}}) \geq \Omega\left(\frac{(B+\epsilon)\prod_{j=1}^{l} M_j}{\sqrt{n}}\right). \tag{2.4.7}$$

The proof is provided in Sec. 2.8. The gap between the upper bound and the lower bound is $\mathcal{O}(h\sqrt{l \log l})$.

Interpretation of the Main Results. Our main results suggest that the robust generalization gap will approach zero if we have infinite samples. When the number of samples is finite, the robust generalization gap might positively relate to the magnitude of sample B, the perturbation intensity, the depth-l, the width-h, and the product of weight norm. In practice, it is easy to see that the robust generalization gap is larger when B and ϵ are larger. From the experiments presented later, we will see that the robust generalization gap is positively related to the product of the weight norms.

Discussion on Network Size. It is observed in practice that larger networks usually have better robust generalization ability. Therefore, depth-l and width-h should not exist in the upper bound. We conjecture that the lower bound, which is independent of l and h, might be closer to the true bound.

In the next section, we extend the adversarial Rademacher complexity to the Multi-class classification cases.

2.5 Margin Bounds for Multi-Class Classification

2.5.1 Setting for Multi-Class Classification

The setting for multi-class classification follows (Bartlett & Mendelson, 2002). In a K-class classification problem, let $\mathcal{Y} = \{1, 2, \cdots, K\}$. The functions in the hypothesis class $\mathcal{F}$ map $\mathcal{X}$ to $\mathbb{R}^K$, the k-th output of f is the score of $f(\boldsymbol{x})$ assigned to the k-th class.

Define the margin operator $M(f(\boldsymbol{x}), y) = [f(\boldsymbol{x})]_y - \max_{y' \neq y}[f(\boldsymbol{x})]_{y'}$. The function makes a correct prediction if and only if $M(f(\boldsymbol{x}), y) > 0$. We consider a particular loss function $\ell(f(\boldsymbol{x}), y) = \phi_\gamma(M(f(\boldsymbol{x}), y))$, where $\gamma > 0$ and $\phi_\gamma : \mathbb{R} \to [0, 1]$ is the ramp loss:

$$\phi_\gamma(t) = \begin{cases} 1 & t \leq 0 \\ 1 - \frac{t}{\gamma} & 0 < t < \gamma \\ 0 & t \geq \gamma. \end{cases}$$

$\phi_\gamma(t) \in [0, 1]$ and $\phi_\gamma(\cdot)$ is $1/\gamma$-Lipschitz. The loss function $\ell(f(\boldsymbol{x}), y)$ satisfies:

$$\mathbb{1}(y \neq \arg\max_{y' \in [K]} [f(\boldsymbol{x})]_{y'}) \leq \ell(f(\boldsymbol{x}), y) \leq \mathbb{1}([f(\boldsymbol{x})]_y \leq \gamma + \max_{y' \neq y} [f(\boldsymbol{x})]_{y'}). \tag{2.5.1}$$

Define the function class $\ell_{\mathcal{F}} := \{(\boldsymbol{x}, y) \mapsto \phi_\gamma(M(f(\boldsymbol{x}), y)) : f \in \mathcal{F}\}$.

In adversarial training, let $\mathbb{B}^p_{\boldsymbol{x}}(\epsilon) = \{x' : \|\boldsymbol{x}' - \boldsymbol{x}\|_p \leq \epsilon\}$ and we define the adversarial function class $\tilde{\ell}_{\mathcal{F}} := \{(\boldsymbol{x}, y) \mapsto \max_{x' \in \mathbb{B}^p_{\boldsymbol{x}}(\epsilon)} \ell(f(\boldsymbol{x}'), y) : f \in \mathcal{F}\}$. We have

Corollary 2.1 ((Yin et al., 2019)). *Consider the above adversarial multi-class classification setting. For any fixed $\gamma > 0$, we have with probability at least $1 - \delta$, for all $f \in \mathcal{F}$,*

$$\begin{aligned} &\mathbb{P}_{(\boldsymbol{x}, y) \sim \mathcal{D}} \left\{ \exists\ \boldsymbol{x}' \in \mathbb{B}^p_{\boldsymbol{x}}(\epsilon)\ s.t.\ y \neq \arg\max_{y' \in [K]} [f(\boldsymbol{x}')]_{y'} \right\} \\ \leq &\frac{1}{n} \sum_{i=1}^{n} \mathbb{1}(\exists\ \boldsymbol{x}_i' \in \mathbb{B}^p_{\boldsymbol{x}_i}(\epsilon)\ s.t.\ [f(\boldsymbol{x}_i')]_{y_i} \leq \gamma + \max_{y' \neq y} [f(\boldsymbol{x}_i')]_{y'}) + 2\mathcal{R}_S(\tilde{\ell}_{\mathcal{F}}) + 3\sqrt{\frac{\log \frac{2}{\delta}}{2n}}. \end{aligned}$$

2.5.2 Adversarial Rademacher Complexity in Multi-Class Cases

Under the multi-class setting, we have the following bound for adversarial Rademacher complexity.

Theorem 2.4. *Given the function class $\mathcal{F}$ in equation (2.2.2) under Frobenius Norm, and the corresponding adversarial function class $\tilde{\mathcal{F}}$ in equation (2.2.1). The adversarial*

Rademacher complexity of deep neural networks $\mathcal{R}_S(\tilde{\ell}_{\mathcal{F}})$ *satisfies*

$$\mathcal{R}_S(\tilde{\ell}_{\mathcal{F}}) \leq \frac{48K}{\gamma\sqrt{n}} \max\{1, q^{\frac{1}{2}-\frac{1}{p}}\}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon)L_\rho^{l-1}\sqrt{\sum_{j=1}^{l} h_j h_{j-1}\log(3l)}\prod_{j=1}^{l} M_j. \quad (2.5.2)$$

The $\|\cdot\|_{1,\infty}$-norm bound is similar, except the term $\max\{1, q^{\frac{1}{2}-\frac{1}{p}}\}$. Below we sketch the proof. The completed proof is provided in Sec. 2.8.

Step 1: Let $\tilde{\mathcal{F}}^k = \{(\boldsymbol{x}, y) \to \inf_{\|\boldsymbol{x}'-\boldsymbol{x}\|\leq\epsilon}([f(\boldsymbol{x}')]_y - [f(\boldsymbol{x}')]_k), f \in \mathcal{F}\}$, then $\mathcal{R}_S(\tilde{\ell}_{\mathcal{F}}) \leq K\mathcal{R}_S(\tilde{\ell}_{\mathcal{F}^k})$. Step 2: By the Lipschitz property of $\phi_\gamma(\cdot)$, $\mathcal{R}_S(\tilde{\ell}_{\mathcal{F}^k}) \leq \frac{1}{\gamma}\mathcal{R}_S(\mathcal{F}^k)$. Step 3: The calculation of $\mathcal{R}_S(\tilde{\mathcal{F}}^k)$ follows the binary case.

2.5.3 Comparison of Standard and Adversarial Rademacher Complexity

Now, we compare the difference between the bounds for (standard) Rademacher complexity and adversarial Rademacher complexity. We have shown that

$$\mathcal{R}_S(\ell_{\mathcal{F}}) \leq \mathcal{O}\left(\frac{B\sqrt{l}\prod_{j=1}^{l} M_j}{\gamma\sqrt{n}}\right) \text{ and } \mathcal{R}_S(\tilde{\ell}_{\mathcal{F}}) \leq \mathcal{O}\left(\frac{(B+\epsilon)h\sqrt{l\log l}\prod_{j=1}^{l} M_j}{\gamma\sqrt{n}}\right), \quad (2.5.3)$$

where we use the upper bound of $\mathcal{R}_S(\mathcal{F})$ in (Golowich et al., 2018).

Algorithm-Independent Factors. In the two bounds, the algorithm-independent factors include Sample size B, perturbation intensity ϵ, depth-l, and width-h. To simplify the notations, we let $C_{std} = B\sqrt{l}$ and $C_{adv} = (B+\epsilon)h\sqrt{l\log l}$ be the constants in standard and adversarial Rademacher complexity, respectively. We simply have $C_{adv} > C_{std}$.

Algorithm-Dependent Factors. In the two bounds, the margins γ and the product of the matrix norms $\prod_{j=1}^{l}\|W_j\|$ depend on the training algorithms. To simplify the notations, we define $W_{std} := \prod_{j=1}^{l}\|W_j\|/\gamma$ if the training algorithm is standard training. Correspondingly, let $W_{adv} := \prod_{j=1}^{l}\|W_j\|/\gamma$ if the training algorithm is adversarial training. As shown in the next section and Sec. 2.10, $W_{adv} > W_{std}$ holds in our

experiments.

Notation of generalization gaps. In the next section, we use $\mathcal{E}(\cdot)$ and $\tilde{\mathcal{E}}(\cdot)$ to denote the standard and robust generalization gap. We use f_{std} and f_{adv} to denote the standard- and adversarially-trained model. Our goal is to analyze the correlation between these factors and generalization gap to study why the robust generalization gap of an adversarial training model is large, which is quite different from the standard generalization gap of a standard-trained model, i.e., we want to analyze why $\tilde{\mathcal{E}}(f_{adv}) > \mathcal{E}(f_{std})$. Based on the standard and adversarial Rademacher complexity bounds, it is suggested that

$$\tilde{\mathcal{E}}(f_{adv}) \propto C_{adv} W_{adv} \quad \textit{and} \quad \mathcal{E}(f_{std}) \propto C_{std} W_{std}.$$

To analyze the individual effect of factors C_{adv} and W_{adv}, we further introduce two kinds of generalization gaps, the robust generalization gap of a standard-trained model ($\tilde{\mathcal{E}}(f_{std})$) and the standard generalization gap of an adversarially-trained model ($\mathcal{E}(f_{adv})$). The standard and adversarial Rademacher complexity suggest that

$$\tilde{\mathcal{E}}(f_{std}) \propto C_{adv} W_{std} \quad \textit{and} \quad \mathcal{E}(f_{adv}) \propto C_{std} W_{adv}.$$

2.6 Experiment

As we discuss in the previous section, the product of weight norm $\prod_{j=1}^{l} \|W_j\|/\gamma$ are algorithm-dependent factors in the ARC bounds. We provide experiments comparing the difference between these terms in standard and adversarial settings. Since the bounds also hold for convolution neural networks, we consider the experiments of training VGG networks (Simonyan & Zisserman, 2014) on CIFAR-10 (Krizhevsky et al., 2009) and CIFAR-100. We trained 88 models to study the weight norm and generalization gap. Additional experiments are provided in Sec. 2.10. [1]

[1] https://github.com/JiancongXiao/Adversarial-Rademacher-Complexity.

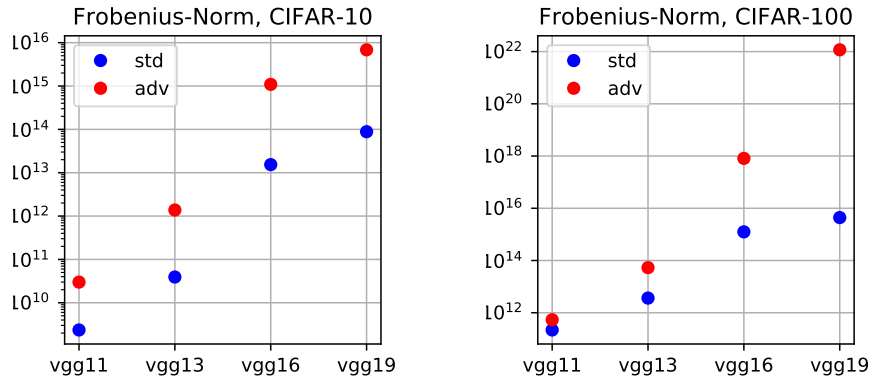


Figure 2.1: Frobienus norm for standard training and adversarial training on CIFAR-10 and CIFAR-100.

Training Settings. For both standard and adversarial training, we use the stochastic gradient descent (SGD) optimizer, along with a learning rate schedule, which is 0.1 over the first 100 epochs, down to 0.01 over the following 50 epochs, and finally be 0.001 in the last 50 epochs. Weight decay is set to be 5×10^{-4} in most of the models, which is shown to be optimal for robust accuracy. Weight decay is also set to be other values for ablation studies. For adversarial settings, we adopt the ℓ_∞ PGD adversarial training (Madry et al., 2017). The perturbation intensity is set to be 8/255. We set the number of steps as 20 and further increase it to 40 in the testing phase. For the stepsize in the inner maximization, we set it as 2/255.

Calculation of Margins. We adopt the setting in (Neyshabur et al., 2017a). In standard training, we set the margin over training set to be 5^{th}-percentile of the margins of the data points in S. i.e. $\text{Prc}_5\{f(\boldsymbol{x}_i)[y_i] - \max_{y \neq y_i} f(\boldsymbol{x})[y] | (\boldsymbol{x}_i, y_i) \in S\}$. In adversarial settings, we set the margin over training set to be 5^{th}-percentile of the margins of the PGD-adversarial examples of S. The choice of 5^{th}-percentile is because the training accuracy is 100% in all the experiments. We provide ablation studies about the percentile in the Sec. 2.10.

Adversarially-trained Models Have Larger Weight Norms. In Figure 2.1, we show the experiments of standard and adversarial training VGG on CIFAR-10 and CIFAR-100[2]. The y-axis is in the logarithm scale. In these eight cases, the weight norms of adv-trained models are larger than that of std-trained models, i.e., $W_{adv} \geq W_{std}$. Ablation studies are provided in Sec. 2.10. $W_{adv} \geq W_{std}$ can also be observed in these experiments.

Standard and Robust Generalization Gap. In Table 2.2, we show the standard and robust generalization gap of both standard-trained and adversarially-trained models. We use the results of training VGG-19 on CIFAR-10 to discuss the experiments. Firstly, we can see that $\mathcal{E}(f_{std})$ is small (=10.45%). On the other hand, an adversarial-trained model has a larger standard generalization gap ($\mathcal{E}(f_{adv})$=26.34%). It is a widely observed phenomenon that adversarial training hurts standard generalization. One possible reason is that adversarial training overfits the adversarial examples and performs worse on the original examples. Secondly, the robust generalization gap of a standard-trained model is very small ($\tilde{\mathcal{E}}(f_{std}) = 0$). It is because the standard-trained model can easily be attacked on both the training set and test set. Then, the robust training accuracy and test accuracy are closed to 0%. Therefore, the robust generalization gap is also 0. On the contrary, $\tilde{\mathcal{E}}(f_{adv})$=58.90%, i.e. the adversarial generalization is bad. This is also observed in the previous studies, and we aim to discuss this phenomenon.

Table 2.2: Comparison of the four kinds of generalization Gap. The experiments are training VGG-19 on CIFAR-10. Notice that $\tilde{\mathcal{E}}(f_{std})$=0% is a degenerated case, with training error=100%. In the other three cases, the training errors≈0%.

	Standard-trained models		Adversarially-trained models	
Types of Generalization Gaps	Standard	Robust	Standard	Robust
Training Errors	0%	100%	0%	0.02%
Test Errors	10.45%	100%	26.34%	58.92%
Generalization Gaps	$\mathcal{E}(f_{std})$=10.45%	$\tilde{\mathcal{E}}(f_{std})$=0%	$\mathcal{E}(f_{adv})$=26.34%	$\tilde{\mathcal{E}}(f_{adv})$=58.90%

[2]Larger models have more parameters and yield larger weight norm. These experiments aim to show the difference between adv-trained and std-trained models.

$\tilde{\mathcal{E}}(f_{std})$=0% is a degenerated case. In Table 2.2, we can see that the robust training error for a standard-trained model is equal to 100%. Since the model does not fit any adversarial examples in the training set, there is nothing to generalize to the adversarial examples in the test set. The generalization gap becomes meaningless. And the Rademacher complexity bound $\tilde{\mathcal{E}}(f_{std}) \leq \mathcal{O}(C_{adv}W_{std}/\sqrt{n})$ becomes a trivial bound. In the other three cases, the training errors are all ≈0%. The generalization gaps are meaningful. We aim to analyze $\tilde{\mathcal{E}}(f_{adv}) > \mathcal{E}(f_{std})$ by analyzing $\tilde{\mathcal{E}}(f_{adv}) > \mathcal{E}(f_{adv}) > \mathcal{E}(f_{std})$.

The effect of C_{adv}. We first compare the difference between $\tilde{\mathcal{E}}(f_{adv})$ and $\mathcal{E}(f_{adv})$. We can see that $\tilde{\mathcal{E}}(f_{adv}) = 58.90\% > \mathcal{E}(f_{adv}) = 26.34\%$. For an adversarially-trained model, the robust generalization gap is larger than the standard generalization gap. If we use the bounds of adversarial and standard Rademacher complexity as approximations of the robust and standard generalization gap, i.e., $\tilde{\mathcal{E}}(f_{adv}) \propto C_{adv}W_{adv}$ and $\mathcal{E}(f_{adv}) \propto C_{std}W_{adv}$, C_{adv} is possitvely related to the robust generalization gap.

The effect of W_{adv}. Similarly, we compare the difference between $\mathcal{E}(f_{adv})$ and $\mathcal{E}(f_{std})$. We can see that $\mathcal{E}(f_{adv}) = 26.34\% > \mathcal{E}(f_{std}) = 10.45\%$. This is a widely observed phenomenon that adversarial training hurts standard generalization. It can also be explained by the Rademacher bounds. If we use the bounds of standard Rademacher complexity as approximations, i.e., $\mathcal{E}(f_{adv}) \propto C_{std}W_{adv}$ and $\mathcal{E}(f_{std}) \propto C_{std}W_{std}$, W_{adv} is possitive related to the robust generalization gap.

In summary, we can use a simple formula, $C_{adv}W_{adv} > C_{std}W_{adv} > C_{std}W_{std}$, to describe the effect of these factors on generalization gap $\tilde{\mathcal{E}}(f_{adv}) > \mathcal{E}(f_{adv}) > \mathcal{E}(f_{std})$. The difficulty of adversarial generalization might come from two parts, the constant C_{adv} and the weight norms W_{adv}. The first part C_{adv} is independent of the algorithms. It comes from the minimax problem of adversarial training itself, and it cannot be avoided. The second part W_{adv} depends on the algorithms.

Weight Decay. The above analysis suggests controlling the weight norms to improve adversarial generalization. In Sec. 2.10, we provide an experiment to slightly increase

the weight decay. Larger weight decay yields smaller weight norms and smaller generalization. But it also hurts training errors. It seems to be a trade-off between training error and generalization error. When the weight decay is increased to 10^{-2}, the training fails. In this case, the weight norm of the adv-trained model is still larger than that of the std-trained model.

2.7 Conclusion

In this chapter, we provide a method and give the first bounds of adversarial Rademacher complexity of deep neural networks. Based on the analysis, the difficulty of adversarial generalization may come from two parts. One is algorithm-independent, the other one is algorithm-dependent and related to the weight norm of the neural network. We empirically study the correlation between these factors and adversarial generalization. We think our results will motivate more theoretical research to understand adversarial training and empirical research to improve the generalization of adversarial training.

Disussion on Representation Power of DNNs. In our opinion, the reason why adversarial training yields a larger weight norm might be due to the representation power of DNNs. DNNs with small weight norms cannot fit adversarial examples on the training set. Then, the algorithms will find DNNs with large weight norms. However, DNNs with large weight norms cannot generalize well, as it is suggested by the above analysis. Therefore, robust overfitting appears. To study this hypothesis requires large-scale experiments, beyond the scope of our work.

2.8 Proof of the Theorems

Proof of Theorem 2.1

Before we provide the proof, we first introduce the following definition and lemma.

Covering Number. Our method for analyzing adversarial Rademacher complexity is based on the covering number. We first provide the definition of covering number.

Definition 2.2 (ε-cover)**.** *Let $\varepsilon > 0$ and $(V, d(\cdot,\cdot))$ be a metric space, where $d(\cdot,\cdot)$ is a (pseudo)-metric. We say $\mathcal{C} \subset V$ is an ε-cover of V, if for any $v \in V$, there exists $v' \in \mathcal{C}$ s.t. $d(v, v') \leq \varepsilon$. Define the smallest $|\mathcal{C}|$ as the ε-covering number of V and denote as $\mathcal{N}(V, d(\cdot,\cdot), \varepsilon)$.*

Next, we define the ε-covering number of a function class $\mathcal{F}$. Given the sample dataset $S = \{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_n, y_n)\}$ with $\boldsymbol{x}_i \in \mathbb{R}^l$, let $\|f\|_S^2 = \frac{1}{n}\sum_{i=1}^n f(\boldsymbol{x}_i, y_i)^2$ be a pseudometric of $\mathcal{F}$. Define the ε-covering number of $\mathcal{F}$ to be $\mathcal{N}(\mathcal{F}, \|\cdot\|_S, \varepsilon)$. Let D be the diameter of $\mathcal{F}$ with $D \triangleq 2\max_{f\in\mathcal{F}} \|f\|_S$.

Proposition 2.3 (Dudley's integral)**.** *The Rademacher complexity $\mathcal{R}_S(\mathcal{F})$ satisfies*

$$\mathcal{R}_S(\mathcal{F}) \leq \inf_{\delta\geq 0}\left[8\delta + \frac{12}{\sqrt{n}}\int_\delta^{D/2}\sqrt{\log \mathcal{N}(\mathcal{F}, \|\cdot\|_S, \varepsilon)}d\varepsilon\right].$$

The proof of this proposition can be found in statistic textbooks (e.g. (Wainwright, 2019)). Based on this, we can use the bound of the covering number of the function class $\mathcal{F}$ to give an upper bound of the Rademacher complexity.

Lemma 2.1 (Covering number of norm-balls)**.** *Let $\mathcal{B}$ be a ℓ_p norm ball with radius W. Let $d(\boldsymbol{x}_1, \boldsymbol{x}_2) = \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_p$. Define the ε-covering number of B as $\mathcal{N}(\mathcal{B}, d(\cdot,\cdot), \varepsilon)$, we have*

$$\mathcal{N}(\mathcal{B}, d(\cdot,\cdot), \varepsilon) \leq (1 + 2W/\varepsilon)^d.$$

In the case of Frobenius norm ball of $m\times k$ matrices, we have the dimension $d = m\times k$ and

$$\mathcal{N}(\mathcal{B}, \|\cdot\|_F, \varepsilon) \leq (1 + 2W/\varepsilon)^{m\times k} \leq (3W/\varepsilon)^{m\times k}.$$

Lemma 2.2. *if* $\boldsymbol{x}_i^* \in \{\boldsymbol{x}_i' | \|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_p \leq \epsilon\}$*, we have*

$$\|\boldsymbol{x}_i^*\|_{r^*} \leq \max\{1, d^{1-\frac{1}{r}-\frac{1}{p}}\}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon).$$

Proof: If $p \geq r^*$, by Holder's inequality with $1/r^* = 1/p + 1/s$,

$$\|\boldsymbol{x}_i^*\|_{r^*} \leq \sup \|\mathbf{1}\|_s \|\boldsymbol{x}_i^*\|_p = \|\mathbf{1}\|_s \|\boldsymbol{x}_i^*\|_p = d^{\frac{1}{s}} \|\boldsymbol{x}_i^*\|_p = d^{1-\frac{1}{r}-\frac{1}{p}} \|\boldsymbol{x}_i^*\|_p.$$

Equality holds when all the entries are equal. If $p < r^*$, we have

$$\|\boldsymbol{x}_i^*\|_{r^*} \leq \|\boldsymbol{x}_i^*\|_p.$$

Equality holds when one of the entries of θ equals to one and the others equal to zero. Then

$$\begin{aligned} \|\boldsymbol{x}_i^*\|_{r^*} &\leq \max\{1, d^{1-\frac{1}{r}-\frac{1}{p}}\} \|\boldsymbol{x}_i^*\|_p \\ &\leq \max\{1, d^{1-\frac{1}{r}-\frac{1}{p}}\} (\|\boldsymbol{x}_i\|_p + \|\boldsymbol{x}_i - \boldsymbol{x}_i^*\|_p) \\ &\leq \max\{1, d^{1-\frac{1}{r}-\frac{1}{p}}\} (\|\boldsymbol{X}\|_{p,\infty} + \epsilon). \end{aligned}$$

□

Lemma 2.3. *Let* A *be an* $m \times k$ *matrix and* b *be a* n*-dimension vector, we have*

$$\|A \cdot b\|_2 \leq \|A\|_F \|b\|_2.$$

Proof: let A_i *be the rows of* A*,* $i = 1 \cdots m$*, we have*

$$\|A \cdot b\|_2 = \sqrt{\sum_{i=1}^{m} (A_i b)^2} \leq \sqrt{\sum_{i=1}^{m} \|A_i\|_2^2 \|b\|_2^2} = \sqrt{\sum_{i=1}^{m} \|A_i\|_2^2} \sqrt{\|b\|_2^2} = \|A\|_F \|b\|_2.$$

□

Step 1: Diameter of $\tilde{\mathcal{F}}$. We first calculate the diameter of $\tilde{\mathcal{F}}$. $\forall f \in \mathcal{F}$, given $(\boldsymbol{x}_i, y_i)$, let $\boldsymbol{x}_i^* \in \arg\inf_{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_p \le \epsilon_p} yf(\boldsymbol{x}_i')$, and we let $\boldsymbol{x}_i^l$ be the output of $\boldsymbol{x}_i^*$ pass through the first to the $l-1$ layer, we have

$$
\begin{aligned}
|\tilde{f}(\boldsymbol{x}_i, y_i)| &= |\inf_{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_p \le \epsilon_p} yf(\boldsymbol{x}_i')| \\
&= |W_l \rho(W_{l-1}\boldsymbol{x}_i^{l-1})| \\
&\overset{(i)}{\le} \|W_l\|_F \cdot \|\rho(W_{l-1}\boldsymbol{x}_i^{l-1})\|_2 \\
&= \|W_l\|_F \cdot \|\rho(W_{l-1}\boldsymbol{x}_i^{l-1}) - \rho(0)\|_2 \\
&\overset{(ii)}{\le} L_\rho M_l \|W_{l-1}(\boldsymbol{x}_i^{l-1})\|_2 \\
&\le \cdots \\
&\le L_\rho^{l-1} \prod_{j=2}^{l} M_j \|W_1 \boldsymbol{x}_i^*\|_2 \\
&\le L_\rho^{l-1} \prod_{j=1}^{l} M_j \cdot \|\boldsymbol{x}_i^*\|_2 \\
&\overset{(iii)}{\le} L_\rho^{l-1} \prod_{j=1}^{l} M_j \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon),
\end{aligned}
$$

where inequality (i) is because of Lemma 2.3, inequality (ii) is because of the Lipschitz propertiy of activation function $\rho(\cdot)$, inequality (iii) is because of Lemma 2.2. Therefore, we have

$$
2\max_{\tilde{f}\in\tilde{\mathcal{F}}} \|\tilde{f}\|_S = 2\left(\frac{1}{n}\sum_{i=1}^{n} |\tilde{f}(\boldsymbol{x}_i, y_i)|^2\right)^{\frac{1}{2}} \le 2L_\rho^{l-1} \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon) \prod_{j=1}^{l} M_j \triangleq D.
$$

Step 2: Distance to $\tilde{\mathcal{F}}^c$. Let $\mathcal{C}_j$ be δ_j-covers of $\{\|W_j\|_F \le M_j\}$, $j = 1, 2, \cdots, l$. Let

$$
\mathcal{F}^c = \{f^c : \boldsymbol{x} \to W_l^c \rho(W_{l-1}^c \rho(\cdots \rho(W_1^c \boldsymbol{x}) \cdots)), W_j^c \in \mathcal{C}_j, j = 1, 2 \cdots, l\}
$$

$$\text{and } \tilde{\mathcal{F}}^c = \{\tilde{f} : (\boldsymbol{x}, y) \to \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\|_p \leq \epsilon} yf(\boldsymbol{x}') | f \in \mathcal{F}^c\}.$$

For all $\tilde{f} \in \tilde{\mathcal{F}}$, we need to find the smallest distance to $\tilde{\mathcal{F}}^c$, i.e. we need to calculate the

$$\max_{\tilde{f} \in \tilde{\mathcal{F}}} \min_{\tilde{f}^c \in \tilde{\mathcal{F}}^c} \|\tilde{f} - \tilde{f}^c\|_S.$$

$\forall (\boldsymbol{x}_i, y_i), i = 1, \cdots, n$, given $\tilde{f}$ and $\tilde{f}^c$ with $\|W_j - W_j^c\|_F \leq \delta_j$, $j = 1, \cdots, l$, consider

$$\begin{aligned} & |\tilde{f}(\boldsymbol{x}_i, y_i) - \tilde{f}^c(\boldsymbol{x}_i, y_i)| \\ = \; & |\inf_{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_p} y_i f(\boldsymbol{x}_i') - \inf_{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_p} y_i f^c(\boldsymbol{x}_i')| \end{aligned}$$

Let

$$x_i^* = \arg \inf_{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_p} y_i f(\boldsymbol{x}_i'), \quad \text{and} \quad x_i^c = \arg \inf_{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_p} y_i f^c(\boldsymbol{x}_i'),$$

we have

$$\begin{aligned} & |\tilde{f}(\boldsymbol{x}_i, y_i) - \tilde{f}^c(\boldsymbol{x}_i, y_i)| \\ = \; & |y_i f(\boldsymbol{x}_i^*) - y_i f^c(\boldsymbol{x}_i^c)| \\ = \; & |f(\boldsymbol{x}_i^*) - f^c(\boldsymbol{x}_i^c)|. \end{aligned}$$

Let

$$\bar{\boldsymbol{x}}_i = \begin{cases} \boldsymbol{x}_i^c & if \quad f(\boldsymbol{x}_i^*) \geq f^c(\boldsymbol{x}_i^c) \\ \boldsymbol{x}_i^* & if \quad f(\boldsymbol{x}_i^*) < f^c(\boldsymbol{x}_i^c) \end{cases}$$

Then,

$$\begin{aligned} & |\tilde{f}(\boldsymbol{x}_i, y_i) - \tilde{f}_c(\boldsymbol{x}_i, y_i)| \\ = \; & |f(\boldsymbol{x}_i^*) - f^c(\boldsymbol{x}_i^c)| \\ \leq \; & |f(\bar{\boldsymbol{x}}_i) - f^c(\bar{\boldsymbol{x}}_i)|. \end{aligned}$$

Define $g_b^a(\cdot)$ as

$$g_b^a(\bar{\boldsymbol{x}}) = W_b\rho(W_{b-1}\rho(\cdots W_{a+1}\rho(W_a^c\cdots\rho(W_1^c\bar{\boldsymbol{x}})\cdots))).$$

In words, for the layers $b \geq j > a$ in $g_b^a(\cdot)$, the weight is W_j, for the layers $a \geq j \geq 1$ in $g_b^a(\cdot)$, the weight is W_j^c. Then we have $f(\bar{\boldsymbol{x}}_i) = g_l^0(\bar{\boldsymbol{x}}_i)$, $f(\bar{\boldsymbol{x}}_i) = g_l^l(\bar{\boldsymbol{x}}_i)$. We can decompose

$$\begin{aligned}
& |f(\bar{\boldsymbol{x}}_i) - f^c(\bar{\boldsymbol{x}}_i)| \\
= \; & |g_l^0(\bar{\boldsymbol{x}}_i) - g_l^l(\bar{\boldsymbol{x}}_i)| \\
= \; & |g_l^0(\bar{\boldsymbol{x}}_i) - g_l^1(\bar{\boldsymbol{x}}_i) + \cdots + g_l^{l-1}(\bar{\boldsymbol{x}}_i) - g_l^l(\bar{\boldsymbol{x}}_i)| \\
\leq \; & |g_l^0(\bar{\boldsymbol{x}}_i) - g_l^1(\bar{\boldsymbol{x}}_i)| + \cdots + |g_l^{l-1}(\bar{\boldsymbol{x}}_i) - g_l^l(\bar{\boldsymbol{x}}_i)|. \qquad (2.8.1)
\end{aligned}$$

To bound the gap $|f(\bar{\boldsymbol{x}}_i) - f^c(\bar{\boldsymbol{x}}_i)|$, we first calculate $|g_l^{j-1}(\bar{\boldsymbol{x}}_i) - g_l^j(\bar{\boldsymbol{x}}_i)|$ for $j = 1, \cdots, l$.

$$\begin{aligned}
& |g_l^{j-1}(\bar{\boldsymbol{x}}_i) - g_l^j(\bar{\boldsymbol{x}}_i)| \\
= \; & |W_l\rho(g_{l-1}^{j-1}(\bar{\boldsymbol{x}}_i)) - W_l\rho(g_{l-1}^j(\bar{\boldsymbol{x}}_i))| \\
\overset{(i)}{\leq} \; & \|W_l\|_F\|\rho(g_{l-1}^{j-1}(\bar{\boldsymbol{x}}_i)) - \rho(g_{l-1}^j(\bar{\boldsymbol{x}}_i))\|_2 \\
\overset{(ii)}{\leq} \; & L_\rho M_l\|g_{l-1}^{j-1}(\bar{\boldsymbol{x}}_i) - g_{l-1}^j(\bar{\boldsymbol{x}}_i)\|_2 \\
\overset{(iii)}{=} \; & L_\rho M_l\|W_{l-1}\rho(g_{l-2}^{j-1}(\bar{\boldsymbol{x}}_i)) - W_{l-1}\rho(g_{l-2}^j(\bar{\boldsymbol{x}}_i))\|_2 \\
\leq \; & \cdots \\
\leq \; & L_\rho^{l-j}\prod_{k=j+1}^{l} M_k\|W_j\rho(g_{j-1}^{j-1}(\bar{\boldsymbol{x}}_i)) - W_j^c\rho(g_{j-1}^{j-1}(\bar{\boldsymbol{x}}_i))\|_2,
\end{aligned}$$

where (i) is due to Lemma 2.3, (ii) is due to the bound of $\|W_j\|$ and the Lipschitz of

$\rho(\cdot)$, (iii) is because of the definition of $g_b^a(\bar{\boldsymbol{x}})$. Then

$$\begin{aligned}
& |g_l^{j-1}(\bar{\boldsymbol{x}}_i) - g_l^j(\bar{\boldsymbol{x}}_i)| \\
\leq\ & L_\rho^{l-j} \prod_{k=j+1}^{l} M_k \|W_j \rho(g_{j-1}^{j-1}(\bar{\boldsymbol{x}}_i)) - W_j^c \rho(g_{j-1}^{j-1}(\bar{\boldsymbol{x}}_i))\|_2 \\
=\ & L_\rho^{l-j} \prod_{k=j+1}^{l} M_k \|(W_j - W_j^c)\rho(g_{j-1}^{j-1}(\bar{\boldsymbol{x}}_i)))\|_2 \\
\overset{(i)}{\leq}\ & L_\rho^{l-j} \prod_{k=j+1}^{l} M_k \|W_j - W_j^c\|_F \|\rho(g_{j-1}^{j-1}(\bar{\boldsymbol{x}}_i)))\|_2 \\
\overset{(ii)}{\leq}\ & L_\rho^{l-j} \prod_{k=j+1}^{l} M_k \delta_j \|\rho(g_{j-1}^{j-1}(\bar{\boldsymbol{x}}_i)))\|_2,
\end{aligned} \tag{2.8.2}$$

where inequality (i) is due to Lemma 2.3, inequality (ii) is due to Lemma 2.3 the assumption that $\|W_j - W_j^c\|_F \leq \delta_j$. It is lefted to bound $\|\rho(g_{j-1}^{j-1}(\bar{\boldsymbol{x}}_i)))\|$, we have

$$\begin{aligned}
& \|\rho(g_{j-1}^{j-1}(\bar{\boldsymbol{x}}_i)))\|_2 \\
=\ & \rho(g_{j-1}^{j-1}(\bar{\boldsymbol{x}}_i))) - \rho(0)\|_2 \\
\leq\ & L_\rho \|g_{j-1}^{j-1}(\bar{\boldsymbol{x}}_i))\|_2 \\
=\ & L_\rho \|W_{j-1}^c \rho(g_{j-2}^{j-2}(\bar{\boldsymbol{x}}_i)))\|_2 \\
\leq\ & L_\rho \|W_{j-1}^c\|_F \|\rho(g_{j-2}^{j-2}(\bar{\boldsymbol{x}}_i)))\|_2 \\
\leq\ & L_\rho M_{j-1} \|\rho(g_{j-2}^{j-2}(\bar{\boldsymbol{x}}_i)))\|_2 \\
\leq\ & \cdots \\
\leq\ & L_\rho^{j-1} \prod_{k=1}^{j-1} M_k \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon).
\end{aligned} \tag{2.8.3}$$

combining Eq. (2.8.2) and (2.8.3), we have

$$\begin{aligned} & |g_l^{j-1}(\bar{\boldsymbol{x}}_i) - g_l^j(\bar{\boldsymbol{x}}_i)| \\ \leq\ & L_\rho^{l-1} \frac{\prod_{k=1}^{l} M_k}{M_j} \delta_j \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon) \\ =\ & \frac{D\delta_j}{2M_j}. \end{aligned} \tag{2.8.4}$$

Therefore, combining Eq. (2.8.1) and (2.8.4), we have

$$\begin{aligned} & |f(\bar{\boldsymbol{x}}_i) - f^c(\bar{\boldsymbol{x}}_i)| \\ \leq\ & |g_l^0(\bar{\boldsymbol{x}}_i) - g_l^1(\bar{\boldsymbol{x}}_i)| + \cdots + |g_l^{l-1}(\bar{\boldsymbol{x}}_i) - g_l^l(\bar{\boldsymbol{x}}_i)| \\ \leq\ & \sum_{j=1}^{l} \frac{D\delta_j}{2M_j}. \end{aligned} \tag{2.8.5}$$

Then

$$\max_{\tilde{f}\in\tilde{\mathcal{F}}} \min_{\tilde{f}^c\in\tilde{\mathcal{F}}^c} \|\tilde{f} - \tilde{f}^c\|_S \leq \sum_{j=1}^{l} \frac{D\delta_j}{2M_j}.$$

Let $\delta_j = 2M_j\varepsilon/dD$, $j = 1, \cdots, l$, we have

$$\max_{\tilde{f}\in\tilde{\mathcal{F}}} \min_{\tilde{f}^c\in\tilde{\mathcal{F}}^c} \|\tilde{f} - \tilde{f}^c\|_S \leq \sum_{j=1}^{l} \frac{D\delta_j}{2M_j} \leq \varepsilon.$$

Step 3: Covering Number of $\tilde{\mathcal{F}}$. We then calculate the ε-covering number $\mathcal{N}(\tilde{\mathcal{F}}, \|\cdot\|_S, \varepsilon)$. Because $\tilde{\mathcal{F}}^c$ is a ε-cover of $\tilde{\mathcal{F}}$. The cardinality of $\tilde{\mathcal{F}}^c$ is

$$\begin{aligned}
\mathcal{N}(\tilde{\mathcal{F}}, \|\cdot\|_S, \varepsilon) &= |\tilde{\mathcal{F}}^c| \\
&= \prod_{j=1}^{l} |\mathcal{C}_j| \\
&\overset{(i)}{\leq} \prod_{j=1}^{l} (\frac{3M_j}{\delta_j})^{h_j h_{l-1}} \\
&= (\frac{3lD}{2\varepsilon})^{\sum_{j=1}^{l} h_j h_{l-1}},
\end{aligned}$$

where inequality (i)) is due to Lemma 2.1.

Step 4, Integration. By Dudley's integral, we have

$$\begin{aligned}
& \mathcal{R}_S(\tilde{\mathcal{F}}) \\
\leq & \inf_{\delta \geq 0} \left[8\delta + \frac{12}{\sqrt{n}} \int_{\delta}^{D/2} \sqrt{\log \mathcal{N}(\mathcal{F}, \|\cdot\|_S, \varepsilon)} d\varepsilon \right] \\
\leq & \inf_{\delta \geq 0} \left[8\delta + \frac{12}{\sqrt{n}} \int_{\delta}^{D/2} \sqrt{(\sum_{j=1}^{l} h_j h_{j-1}) \log(3lD/2\varepsilon)} d\varepsilon \right] \\
= & \inf_{\delta \geq 0} \left[8\delta + \frac{12D\sqrt{\sum_{j=1}^{l} h_j h_{j-1}}}{\sqrt{n}} \int_{\delta/D}^{1/2} \sqrt{\log(3l/2\varepsilon)} d\varepsilon \right]. \qquad (2.8.6)
\end{aligned}$$

Let $\delta \to 0$.[3] Integration by part, we have

[3]Simply let $\delta = D/\sqrt{n}$, we can obtain a bound in $\mathcal{O}(\sqrt{\log n / n})$. To get rid of the $\log n$ term, we can let $\delta \to 0$.

$$
\begin{aligned}
& \int_0^{1/2} \sqrt{\log(3l/2\varepsilon)} d\varepsilon \\
= & \frac{1}{2}\left(\frac{3l}{2}\sqrt{\pi}\mathrm{erfc}(\sqrt{\log 3l}) + \sqrt{\log 3l}\right) \\
\leq & \frac{1}{2}\left(\frac{3l}{2}\sqrt{\pi}\exp(-\sqrt{\log 3l}^2) + \sqrt{\log 3l}\right) \\
= & \frac{1}{2}\left(\frac{\sqrt{\pi}}{2} + \sqrt{\log 3l}\right) \\
\leq & \frac{1}{2}\left(2\sqrt{\log 3l}\right) \\
= & \sqrt{\log 3l}. \qquad (2.8.7)
\end{aligned}
$$

Plugging Eq. (2.8.7) to Eq. (2.8.6), we have

$$
\mathcal{R}_S(\tilde{\mathcal{F}}) \leq \frac{24}{\sqrt{n}} \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon) L_\rho^{l-1} \sqrt{\sum_{j=1}^{l} h_j h_{j-1} \log(3l)} \prod_{j=1}^{l} M_j.
$$

Proof of Theorem 2.2

The proof is similar to the proof of the Frobenius norm bound with two difference. The first one is inequality in Lemma 2.3 is replaced by Lemma 2.4 below. The second one is that the C_j covering sets are for each neurons in this case. While C_j cover each matrices in the previous case. Below we provide the complete proof. We first introduce the following inequality.

Lemma 2.4. *Let A be a $m \times k$ matrix and b be a n-dimension vector, we have*

$$
\|A \cdot b\|_\infty \leq \|A\|_{1,\infty} \|b\|_\infty.
$$

Proof: let A_i be the rows of A, $j = 1 \cdots m$, we have

$$
\|A \cdot b\|_\infty = \max |A_i b| \leq \max \|A_i\|_1 \|b\|_\infty = \|A\|_{1,\infty} \|b\|_\infty.
$$

Step 1: Diameter of $\tilde{\mathcal{F}}$. We first calculate the diameter of $\tilde{\mathcal{F}}$. $\forall f \in \mathcal{F}$, given $(\boldsymbol{x}_i, y_i)$, let $\boldsymbol{x}_i^* \in \arg\inf_{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_p \le \epsilon_p} y f(\boldsymbol{x}_i')$, and we let $\boldsymbol{x}_i^l$ be the output of $\boldsymbol{x}_i^*$ pass through the first to the $l-1$ layer, we have

$$
\begin{aligned}
& |\tilde{f}(\boldsymbol{x}_i, y_i)| \\
= \; & |\inf_{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|_p \le \epsilon_p} y f(\boldsymbol{x}_i')| \\
= \; & |W_l \rho(W_{l-1} \boldsymbol{x}_i^{l-1})| \\
\overset{(i)}{\le} \; & \|W_l\|_{1,\infty} \cdot \|\rho(W_{l-1}\boldsymbol{x}_i^{l-1})\|_\infty \\
= \; & \|W_l\|_{1,\infty} \cdot \|\rho(W_{l-1}\boldsymbol{x}_i^{l-1}) - \rho(0)\|_\infty \\
\overset{(ii)}{\le} \; & L_\rho M_l \|W_{l-1}(\boldsymbol{x}_i^{l-1})\|_\infty \\
\le \; & \cdots \\
\le \; & L_\rho^{l-1} \prod_{j=1}^{l} M_j \cdot \|\boldsymbol{x}_i^*\|_\infty \\
\overset{(iii)}{\le} \; & L_\rho^{l-1} \prod_{j=1}^{l} M_j (\|\boldsymbol{X}\|_{p,\infty} + \epsilon),
\end{aligned}
$$

where inequality (i) is because of Lemma 2.4, inequality (ii) is because of the Lipschitz propertiy of activation function $\rho(\cdot)$, inequality (iii) is because of Lemma 2.2. Therefore, we have

$$
2 \max_{\tilde{f} \in \tilde{\mathcal{F}}} \|\tilde{f}\|_S = 2 \left(\frac{1}{n} \sum_{i=1}^{n} |\tilde{f}(\boldsymbol{x}_i, y_i)|^2 \right)^{\frac{1}{2}} \le 2 L_\rho^{l-1} (\|\boldsymbol{X}\|_{p,\infty} + \epsilon) \prod_{j=1}^{l} M_j \triangleq D.
$$

Step 2: Distance to $\tilde{\mathcal{F}}^c$. Let $\mathcal{C}_j^m$ be δ_j-covers of $\{\|W_j^m\|_1 \le M_j\}$, $j = 1, 2, \cdots, l$, $m = 1, \cdots, h_j$, where W_j^m is the m^{th} row of W_j^m. Let

$$
\mathcal{F}^c = \{f^c : \boldsymbol{x} \to W_l^c \rho(W_{l-1}^c \rho(\cdots \rho(W_1^c \boldsymbol{x}) \cdots)), W_j^{cm} \in \mathcal{C}_j^m, m = 1, \cdots, h_j, j = 1, 2 \cdots, l\}
$$

$$\text{and } \tilde{\mathcal{F}}^c = \{\tilde{f} : (\boldsymbol{x}, y) \to \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\|_p \leq \epsilon} yf(\boldsymbol{x}') | f \in \mathcal{F}^c\}.$$

For all $\tilde{f} \in \tilde{\mathcal{F}}$, we need to find the smallest distance to $\tilde{\mathcal{F}}^c$, i.e. we need to calculate the

$$\max_{\tilde{f} \in \tilde{\mathcal{F}}} \min_{\tilde{f}^c \in \tilde{\mathcal{F}}^c} \|\tilde{f} - \tilde{f}^c\|_S.$$

$\forall (\boldsymbol{x}_i, y_i), j = 1, \cdots, n$, given $\tilde{f}$ and $\tilde{f}^c$ with $|W_j^m - W_j^{cm}| \leq \delta_j$, $m = 1, \cdots, h_j$, $j = 1, \cdots, l$, we have $\|W_j - W_j^c\|_{1,\infty} \leq \delta_j$. By the same argument as the step 2 of the proof o Theorem 2.1, we have

$$\max_{\tilde{f} \in \tilde{\mathcal{F}}} \min_{\tilde{f}^c \in \tilde{\mathcal{F}}^c} \|\tilde{f} - \tilde{f}^c\|_S \leq \sum_{j=1}^{l} \frac{D\delta_j}{2M_j}.$$

Let $\delta_j = 2M_j\varepsilon/dD$, $j = 1, \cdots, l$, we have

$$\max_{\tilde{f} \in \tilde{\mathcal{F}}} \min_{\tilde{f}^c \in \tilde{\mathcal{F}}^c} \|\tilde{f} - \tilde{f}^c\|_S \leq \sum_{j=1}^{l} \frac{D\delta_j}{2M_j} \leq \varepsilon.$$

Step 3: Covering Number of $\tilde{\mathcal{F}}$. We then calculate the ε-covering number $\mathcal{N}(\tilde{\mathcal{F}}, \|\cdot\|_S, \varepsilon)$. Because $\tilde{\mathcal{F}}^c$ is a ε-cover of $\tilde{\mathcal{F}}$. The cardinality of $\tilde{\mathcal{F}}^c$ is

$$\mathcal{N}(\tilde{\mathcal{F}}, \|\cdot\|_S, \varepsilon) = |\tilde{\mathcal{F}}^c| = \prod_{j=1}^{l} \prod_{m=1}^{h_j} |\mathcal{C}_j^m| \overset{(i)}{\leq} \prod_{j=1}^{l} (\frac{3M_j}{\delta_j})^{h_j h_{j-1}} = (\frac{3lD}{2\varepsilon})^{\sum_{j=1}^{l} h_j h_{j-1}},$$

where inequality (i) is due to Lemma 2.1.

Step 4, Integration. By the same argument as the step 4 of the proof o Theorem 2.1, integration by part, we have

$$\mathcal{R}_S(\tilde{\mathcal{F}}) \leq \frac{24}{\sqrt{n}}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon)L_\rho^{l-1}\sqrt{\sum_{j=1}^{l} h_j h_{j-1}\log(3l)}\prod_{j=1}^{l} M_j.$$

Proof of Theorem 2.3

The proof of the above Theorem is based on constructing a linear network. By the definition of Rademacher complexity, if $\mathcal{H}'$ is a subset of $\mathcal{H}$, we have

$$\mathcal{R}_{\mathcal{S}}(\mathcal{H}') = \mathbb{E}_\sigma \frac{1}{n}\big[\sup_{h\in\mathcal{H}'}\sum_{i=1}^{n}\sigma_i h(\boldsymbol{x}_i, y_i)\big] \leq \mathbb{E}_\sigma \frac{1}{n}\big[\sup_{h\in\mathcal{H}}\sum_{i=1}^{n}\sigma_i h(\boldsymbol{x}_i, y_i)\big] = \mathcal{R}_{\mathcal{S}}(\mathcal{H}).$$

Therefore, it is enough to lower bound the complexity of $\tilde{\mathcal{F}}'$ in a particular distribution $\mathcal{D}$, where $\tilde{\mathcal{F}}'$ is a subset of $\tilde{\mathcal{F}}$. Let

$$\tilde{\mathcal{F}}' = \{\boldsymbol{x} \to \inf_{\|\boldsymbol{x}'-\boldsymbol{x}\|_p\leq\epsilon} yM_l \cdot M_2 w^T\boldsymbol{x} | w \in \mathbb{R}^q, \|w\|_2 \leq M_1\}.$$

We first prove that $\tilde{\mathcal{F}}'$ is a subset of $\tilde{\mathcal{F}}$. In $\tilde{\mathcal{F}}$, we let the activation function $\rho(\cdot)$ be a identity mapping. Let

$$W_1 = \begin{bmatrix} w \\ 0 \\ \vdots \\ 0 \end{bmatrix} \in \mathbb{R}^{h_1\times h_0}, \quad W_j = \begin{bmatrix} M_j & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{h_j\times h_{j-1}}, \quad j = 2, \cdots, l. \tag{2.8.8}$$

Then, we have $\|W_j\| \leq M_j$ and $\tilde{\mathcal{F}}$ with additional constraint in Eq. (2.8.8) reduce to $\tilde{\mathcal{F}}'$. In other words, $\tilde{\mathcal{F}}'$ is a subset of $\tilde{\mathcal{F}}$.

It turns out that we need to lower bound the adversarial Rademacher complexity of linear hypothesis. The results are given by the work of (Yin et al., 2019; Awasthi et al., 2020). Below we state the result.

Proposition 2.4. *Given the function class*

$$\mathcal{G} = \{\boldsymbol{x} \to yw^T\boldsymbol{x} | w \in \mathbb{R}^q, \|w\|_r \leq W\}$$

and

$$\tilde{\mathcal{G}} = \{\boldsymbol{x} \to \inf_{\|\boldsymbol{x}'-\boldsymbol{x}\|_r \leq \epsilon} yw^T\boldsymbol{x} | w \in \mathbb{R}^q, \|w\|_r \leq W\},$$

the adversarial Rademacher complexity $\mathcal{R}_S(\tilde{\mathcal{G}})$ *satisfies*

$$\mathcal{R}_S(\tilde{\mathcal{G}}) \geq \max\left\{\mathcal{R}_S(\mathcal{G}), \frac{\epsilon \max\{1, d^{1-\frac{1}{r}-\frac{1}{p}}\}W}{2\sqrt{n}}\right\}.$$

Since the standard Rademacher complexity

$$\mathcal{R}_S(\mathcal{G}) = \frac{W}{m}\mathbb{E}_\sigma\|\sum_{i=1}^n \sigma_i\boldsymbol{x}_i\|_{r*},$$

let $\|\boldsymbol{x}_i\| = B$ with equal entries for $i = 1, \cdots, n$, by Lemma 2.2, we have

$$\mathcal{R}_S(\mathcal{G}) = \frac{W}{m}\mathbb{E}_\sigma|\sum_{i=1}^n \sigma_i| \max\{1, d^{1-\frac{1}{r}-\frac{1}{p}}\}B.$$

By Khintchine's inequality, we know that there exists a universal constant $c > 0$ such that

$$\mathbb{E}_\sigma|\sum_{i=1}^n \sigma_i| \geq c\sqrt{n}.$$

Then, we have

$$\mathcal{R}_S(\mathcal{G}) = \frac{cW}{\sqrt{n}} \max\{1, d^{1-\frac{1}{r}-\frac{1}{p}}\}B.$$

Therefore,

$$\begin{aligned}\mathcal{R}_S(\tilde{\mathcal{G}}) &\geq \max\left\{\mathcal{R}_S(\mathcal{G}), \frac{\epsilon \max\{1, d^{1-\frac{1}{r}-\frac{1}{p}}\}W}{2\sqrt{n}}\right\} \\ &\geq \frac{1}{1+2c}\mathcal{R}_S(\mathcal{G}) + \frac{2c}{1+2c} \times \frac{\epsilon \max\{1, d^{1-\frac{1}{r}-\frac{1}{p}}\}W}{2\sqrt{n}} \\ &\geq \frac{c}{1+2c}\left(\frac{(B+\epsilon)\max\{1, d^{1-\frac{1}{r}-\frac{1}{p}}\}W}{\sqrt{n}}\right).\end{aligned}$$

Let $W = \prod_{j=1}^{l} M_j$, we have

$$\mathcal{R}_S(\tilde{\mathcal{F}}) \geq \Omega\left(\frac{\max\{1, d^{1-\frac{1}{r}-\frac{1}{p}}\}(B+\epsilon)\prod_{j=1}^{l} M_j}{\sqrt{n}}\right),$$

where $r = 2$ for frobenius norm bound and $r = 1$ for $\|\cdot\|_{1,\infty}$-norm bound. □

Proof of Theorem 2.4

Proof: Firstly, we have

$$\begin{aligned}&\tilde{\ell}(f(\boldsymbol{x}), y) \\ =\ & \max_{\|\boldsymbol{x}-\boldsymbol{x}'\|\leq\epsilon} \phi_\gamma(M(f(\boldsymbol{x}'), y)) \\ =\ & \phi_\gamma(\inf_{\|\boldsymbol{x}-\boldsymbol{x}'\|\leq\epsilon} M(f(\boldsymbol{x}'), y)) \\ =\ & \phi_\gamma(\inf_{\|\boldsymbol{x}-\boldsymbol{x}'\|\leq\epsilon} ([f(\boldsymbol{x}')]_y - \max_{y'\neq y}[]f(\boldsymbol{x}')]_{y'})) \\ =\ & \phi_\gamma(\inf_{\|\boldsymbol{x}-\boldsymbol{x}'\|\leq\epsilon} \inf_{y'\neq y} ([f(\boldsymbol{x}')]_y - [f(\boldsymbol{x}')]_{y'})) \\ =\ & \phi_\gamma(\inf_{y'\neq y} \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\|\leq\epsilon} ([f(\boldsymbol{x}')]_y - [f(\boldsymbol{x}')]_{y'})). \\ =\ & \max_{y'\neq y} \phi_\gamma(\inf_{\|\boldsymbol{x}-\boldsymbol{x}'\|\leq\epsilon} ([f(\boldsymbol{x}')]_y - [f(\boldsymbol{x}')]_{y'})).\end{aligned}$$

Define

$$h^k(\boldsymbol{x}, y) = \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\|\leq\epsilon} ([f(\boldsymbol{x}')]_y - [f(\boldsymbol{x}')]_k) + \gamma \mathbb{1}(y = k),$$

we now prove that

$$\max_{y' \neq y} \phi_\gamma(\inf_{\|\boldsymbol{x}-\boldsymbol{x}'\| \leq \epsilon} ([f(\boldsymbol{x}')]_y - [f(\boldsymbol{x}')]_{y'})) = \max_k \phi_\gamma(h^k(\boldsymbol{x}, y)).$$

If

$$\inf_{y' \neq y} \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\| \leq \epsilon} ([f(\boldsymbol{x}')]_y - [f(\boldsymbol{x}')]_{y'}) \leq \gamma,$$

we have

$$\inf_{y' \neq y} \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\| \leq \epsilon} ([f(\boldsymbol{x}')]_y - f(\boldsymbol{x}')]_{y'})) = \inf_k h^k(\boldsymbol{x}, y).$$

If

$$\inf_{y' \neq y} \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\| \leq \epsilon} ([f(\boldsymbol{x}')]_y - [f(\boldsymbol{x}')]_{y'}) > \gamma,$$

we have

$$\phi_\gamma(\inf_{y' \neq y} \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\| \leq \epsilon} ([f(\boldsymbol{x}')]_y - [f(\boldsymbol{x}')]_{y'})) = \phi_\gamma(\inf_k h^k(\boldsymbol{x}, y)) = 0.$$

Therefore, we have

$$\tilde{\ell}(f(\boldsymbol{x}), y) = \phi_\gamma(\inf_k h^k(\boldsymbol{x}, y)) = \max_k \phi_\gamma(h^k(\boldsymbol{x}, y)).$$

Define $\mathcal{H}^k = \{h^k(\boldsymbol{x}, y) = \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\| \leq \epsilon}([f(\boldsymbol{x}')]_y - f(\boldsymbol{x}')]_k) + \gamma \mathbb{1}(y = k) | f \in \mathcal{F}\}$, we have

$$\mathcal{R}_S(\tilde{\ell}_\mathcal{F}) \overset{(i)}{\leq} K\mathcal{R}_S(\phi_\gamma \circ \mathcal{H}^k) \overset{(ii)}{\leq} \frac{K}{\gamma}\mathcal{R}_S(\mathcal{H}^k), \tag{2.8.9}$$

where inequality (i) is the Lemma 9.1 of (Mohri et al., 2018), inequality (ii) is due to the Lipschitz property of $\phi_\gamma(\cdot)$. Now, define $f^k(\boldsymbol{x}, y) = \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\| \leq \epsilon}([f(\boldsymbol{x}')]_y - f(\boldsymbol{x}')]_k)$,

we have $h^k(\boldsymbol{x}, y) = f^k(\boldsymbol{x}, y) + \gamma \mathbb{1}(y = k)$. Define the function class

$$\mathcal{F}^k = \{f^k(\boldsymbol{x}, y) = \inf_{\|\boldsymbol{x}-\boldsymbol{x}'\| \leq \epsilon} ([f(\boldsymbol{x}')]_y - [f(\boldsymbol{x}')]_k) | f \in \mathcal{F}\}.$$

We have

$$\begin{aligned}
\mathcal{R}_S(\mathcal{H}^k) &= \frac{1}{n}\mathbb{E}_\sigma \sup_{h^k \in \mathcal{H}^k} \sum_{i=1}^n \sigma_i h^k(\boldsymbol{x}_i, y_i) \\
&= \frac{1}{n}\mathbb{E}_\sigma \sup_{h^k \in \mathcal{H}^k} \sum_{i=1}^n \sigma_i \left[f^k(\boldsymbol{x}_i, y_i) + \gamma \mathbb{1}(y = k) \right] \\
&= \frac{1}{n}\mathbb{E}_\sigma \sup_{h^k \in \mathcal{H}^k} \sum_{i=1}^n \sigma_i f^k(\boldsymbol{x}_i, y_i) + \frac{1}{n}\mathbb{E}_\sigma \sum_{i=1}^n \sigma_i \gamma \mathbb{1}(y = k) \\
&= \frac{1}{n}\mathbb{E}_\sigma \sup_{f^k \in \mathcal{F}^k} \sum_{i=1}^n \sigma_i f^k(\boldsymbol{x}_i, y_i) \\
&= \mathcal{R}_S(\mathcal{F}^k)
\end{aligned}$$

Finally, we need to bound the Rademacher complexity of $\mathcal{R}_S(\mathcal{F}^k)$. Notice that

$$[f(\boldsymbol{x})]_y - [f(\boldsymbol{x})]_k = (W_l^y - W_l^k)\rho(W_{l-1}(\rho(\cdots W_1(\boldsymbol{x})\cdots))),$$

and we have $\|W_l^y - W_l^k\|_F \leq 2M_j$. By Theorem 2.1 (the results in binary classification case), we have

$$\mathcal{R}_S(\mathcal{F}^k) \leq \frac{48}{\sqrt{n}} \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon) L_\rho^{l-1} \sqrt{\sum_{j=1}^l h_j h_{j-1} \log(3l)} \prod_{j=1}^l M_j. \quad (2.8.10)$$

Combining Eq. (2.8.10) and (2.8.9), we obtain that

$$\mathcal{R}_S(\tilde{\ell}_\mathcal{F}) \leq \frac{48K}{\gamma\sqrt{n}} \max\{1, d^{\frac{1}{2}-\frac{1}{p}}\}(\|\boldsymbol{X}\|_{p,\infty} + \epsilon) L_\rho^{l-1} \sqrt{\sum_{j=1}^l h_j h_{j-1} \log(3l)} \prod_{j=1}^l M_j.$$

2.9 Discussion on Existing Methods for Rademacher Complexity

In this section, we discuss the related work, discuss the existing methods in calculating Rademacher complexity, and identify the difficulty of analyzing adversarial Rademacher complexity.

Existing Bounds for (Standard) Rademacher Complexity

Layer Peelings. The main idea of calculating the Rademacher complexity of multi-layers neural networks is the 'peeling off' technique (Neyshabur et al., 2015). We denote $g \circ \mathcal{F}$ as the function class $\{g \circ f | f \in \mathcal{F}\}$. By Talagrand' s Lemma, we have $\mathcal{R}_{\mathcal{S}}(g \circ f) \leq L_g \mathcal{R}_{\mathcal{S}}(\mathcal{F})$. Based on this property, we can obtain $\mathcal{R}_{\mathcal{S}}(\mathcal{F}_l) \leq 2L_\rho M_j \mathcal{R}_{\mathcal{S}}(\mathcal{F}_{l-1})$, where $\mathcal{F}_l$ is the function class of l-layers neural networks. Since the Rdemacher complexity of linear function class is bounded by $\mathcal{O}(BM_1/\sqrt{n})$, we can get the upper bound $\mathcal{O}(B2^l L_\rho^{l-1} \prod_{j=1}^l M_j/\sqrt{n})$ by induction. We can remove the L_ρ by assuming that $L_\rho = 1$ (e.g. Relu activation function).

(Golowich et al., 2018) improves the dependence on depth-l from 2^l to $\sqrt{d}$. The main idea is to rewrite the Rademacher complexity $\mathbb{E}_\sigma[\cdot]$ as $\mathbb{E}_\sigma \exp\ln[\cdot]$. Then, we can peel off the layer inside the $\ln(\cdot)$ function and the 2^l now appears inside the $\ln(\cdot)$.

Covering Number. (Bartlett et al., 2017) uses a covering numbers argument to show that the generalization gap scale as

$$\tilde{\mathcal{O}}\Big(\frac{B\prod_{j=1}^l \|W_j\|}{\sqrt{n}}\Big(\sum_{j=1}^l \frac{\|W_j\|_{2,1}^{2/3}}{\|W_j\|^{2/3}}\Big)^{3/2}\Big),$$

where $\|\cdot\|$ is the spectral norm. The proof is based on the induction on layers. Let W_j be the weight matrix of the present layer and X_j be the output of X pass through the first to the $l-1$ layer. Then, one can compute the matrix covering number $\mathcal{N}(\{W_j X_j\}, \|\cdot\|_2, \epsilon)$ by induction.

Existing Bounds for (Variants) of Adversarial Rademacher Complexity

In linear cases, the upper bounds can be directly derived by the definition of ARC (Khim & Loh, 2018; Yin et al., 2019). Below we discuss the attempts of analyzing adversarial Rademacher complexity in multi-layers cases. We start from the work using modified adversarial loss.

Tree Transformation Loss. The work of (Khim & Loh, 2018) introduced a tree transformation T and showed that $\max_{\|x-x'\|\leq\epsilon} \ell(f(\boldsymbol{x}), y) \leq \ell(Tf(\boldsymbol{x}), y)$. Then, we have the following upper bound for the adversarial population risk. For $\delta \in (0, 1)$,

$$R_{\mathcal{D}}(f) \leq R(Tf) \leq R_n(Tf) + 2L\mathcal{R}_S(T \circ \mathcal{F}) + 3\sqrt{\frac{\log \frac{2}{\delta}}{2n}}.$$

It gives an upper bound of the robust population risk by the empirical risk and the standard Rademacher complexity of $T \circ f$. $\mathcal{R}_S(T \circ \mathcal{F})$ can be viewed as an approximation of adversarial Rademacher complexity. However, the empirical risk $R_n(Tf)$ is not the objective in practice. This analysis does not provide a guarantee for robust generalization gaps.

SDP Relaxation Surrogate Loss. In the work of (Yin et al., 2019), the authors defined the SDP surrogate loss as

$$\hat{\ell}(f(\boldsymbol{x}), y) = \phi_\gamma\left(M(f(\boldsymbol{x}), y) - \frac{\epsilon}{2} \max_{k\in[K], z=\pm 1} \max_{P\succeq 0, diag(P)\leq 1} \langle zQ(w_{2,k}, W_1), P\rangle\right)$$

to approximate the adversarial loss for two-layer neural nets. Therefore, the adversarial Rademacher complexity is approximated by the Rademacher complexity on this loss function.

FGSM Attack Loss. The work of (Gao & Wang, 2021) tries to provide an upper bound for adversarial Rademacher complexity. To deal with the max operation in the adversarial loss, they consider FGSM adversarial examples. By some assumptions

on the gradient, they provide an upper bound for adversarial Rademacher complexity using the loss $\ell(f(\boldsymbol{x}_{FGSM}), y)$. However, the bound includes some parameters of the assumptions on the gradients, it is not a clean bound.

The following work provided a bound using original adversarial loss.

Massart's Lemma Bound. The work of (Awasthi et al., 2020) showed that the ARC is bounded by

$$\mathcal{O}\left(\frac{(B+\epsilon)\sqrt{h_1 q}\sqrt{\log m}M_1 M_2}{\sqrt{n}}\right)$$

in two-layers cases.

Difficulty of Using Layer Peeling to Bound Adversarial Rademacher Complexity

We first take a look at the layer peeling technique.

$$\begin{aligned}
\mathcal{R}_{\mathcal{S}}(\mathcal{H}) &= \mathbb{E}_\sigma \frac{1}{n}\big[\sup_{h\in\mathcal{H}} \sum_{i=1}^{n} \sigma_i h(\boldsymbol{x}_i)\big] \\
&= \mathbb{E}_\sigma \frac{1}{n}\big[\sup_{h'\in\mathcal{H}_{l-1}, \|W_l\|\le M_l} \sum_{i=1}^{n} \sigma_i W_l \rho(h'(\boldsymbol{x}_i))\big] \\
&\le M_l \mathbb{E}_\sigma \frac{1}{n}\left[\sup_{h'\in\mathcal{H}_{l-1}} \left\| \sum_{i=1}^{n} \sigma_i \rho(h'(\boldsymbol{x}_i))\right\|\right] \\
&\le 2M_l L_\rho \mathbb{E}_\sigma \frac{1}{n}\left[\sup_{h'\in\mathcal{H}_{l-1}} \sum_{i=1}^{n} \sigma_i h'(\boldsymbol{x}_i)\right] \\
&= 2M_l L_\rho \mathcal{R}_{\mathcal{S}}(\mathcal{H}_{l-1}),
\end{aligned}$$

In adversarial settings, if we directly apply the layer peeling technique, we have

$$\begin{aligned}
\mathcal{R}_{\mathcal{S}}(\tilde{\mathcal{H}}) &= \mathbb{E}_\sigma \frac{1}{n}\big[\sup_{h\in\mathcal{H}} \sum_{i=1}^{n} \sigma_i \max_{\|\boldsymbol{x}_i - \boldsymbol{x}_i'\|\le\epsilon} h(\boldsymbol{x}_i')\big] \\
&= \mathbb{E}_\sigma \frac{1}{n}\big[\sup_{h'\in\mathcal{H}_{l-1}, \|W_l\|\le M_l} \sum_{i=1}^{n} \sigma_i W_l \rho(h'(\boldsymbol{x}_i^*(h)))\big] \\
&\le M_l \mathbb{E}_\sigma \frac{1}{n}\Big[\sup_{h'\in\mathcal{H}_{l-1}} \Big\| \sum_{i=1}^{n} \sigma_i \rho(h'(\boldsymbol{x}_i^*(h)))\Big\|\Big] \\
&\le 2M_l L_\rho \mathbb{E}_\sigma \frac{1}{n}\Big[\sup_{h'\in\mathcal{H}_{l-1}} \sum_{i=1}^{n} \sigma_i h'(\boldsymbol{x}_i^*(h))\Big] \\
&\neq 2M_l L_\rho \mathbb{E}_\sigma \frac{1}{n}\Big[\sup_{h'\in\mathcal{H}_{l-1}} \sum_{i=1}^{n} \sigma_i h'(\boldsymbol{x}_i^*(h'))\Big] \\
&= 2M_l L_\rho \mathcal{R}_{\mathcal{S}}(\tilde{\mathcal{H}}_{l-1}),
\end{aligned}$$

where $\boldsymbol{x}_i^*(h)$ is the optimal adversarial example given a l-layers neural networks, $\boldsymbol{x}_i^*(h')$ is the optimal adversarial example given a $l-1$-layers neural networks. $\boldsymbol{x}_i^*(h) \neq \boldsymbol{x}_i^*(h')$ is the main reason why layer peeling cannot be directly extended to the adversarial settings.

2.10 Additional Experiments

In this section, we provide additional experiments.

Experiments on VGG-11 and VGG-13

In Figure 2.2, we show the experiments on VGG-11 and VGG-13. In Figure 2.3, we show the experiments on VGG-16 and VGG-19. he gap of product of Frobenius norm between standard and adversarial training is large, which yields bad generalization.

$\|\cdot\|_{1,\infty}$-Norm Bounds. The $\|\cdot\|_{1,\infty}$-norm bounds are shown in Figure 2.4. Similar to the Frobenius norm bounds, the gap of $\prod_{j=1}^{l}\|W_j\|_{1,\infty}$ between adversarial training and standard training are large. But the magnitude of $\prod_{j=1}^{l}\|W_j\|_{1,\infty}$ is larger than

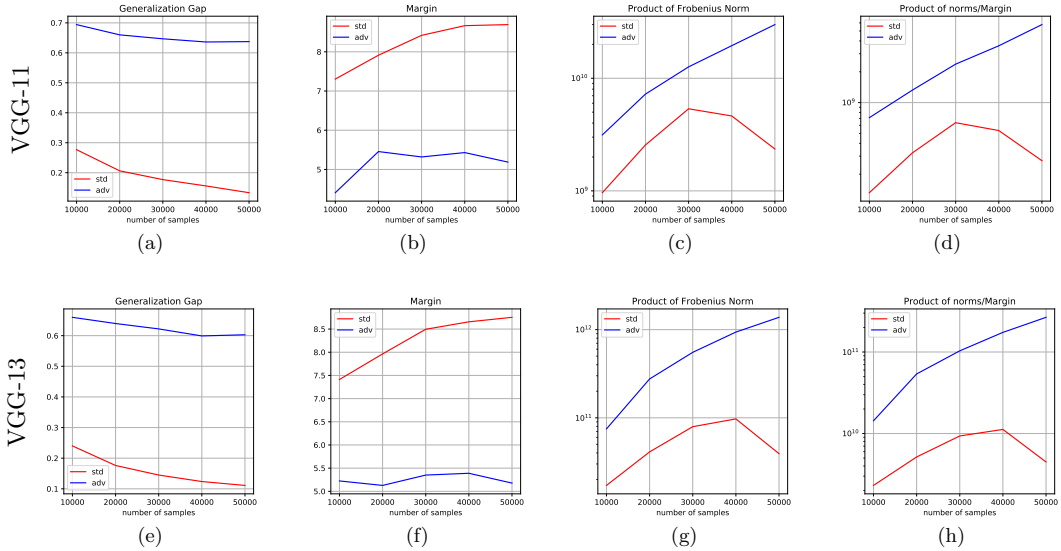


Figure 2.2: Product of the Frobenius norm in the experiments on VGG networks. The red lines are the results of standard training. The blue lines are the results of adversarial training. The first row are the experiments on VGG-11. The second row are the experiments on VGG-13. (a) and (e): Generalization gap. (b) and (f): Margin γ over training set. (c) and (g): $\prod_{j=1}^{l} \|W_j\|_F$ of the neural networks. (d) and (h): $\prod_{j=1}^{l} \|W_j\|_F / \gamma$ of the neural networks.

the magnitude of $\prod_{j=1}^{l} \|W_j\|_F$.

Ablation Study of Margins

In Figure 2.5, we show the results of the margins in 1^{th}, 3^{th}, and, 5^{th}-percentile of the training dataset. Since the (robust) training accuracy is 100%, the choice of percentile will not affect the results. As we can see in the Figure, in all the cases, the margins of standard training are larger than the margins of adversarial training. Since the margins appear in the divider in the upper bound of Rademacher complexity, the margins of the training dataset have some small effects on the bad generalization of adversarial training.

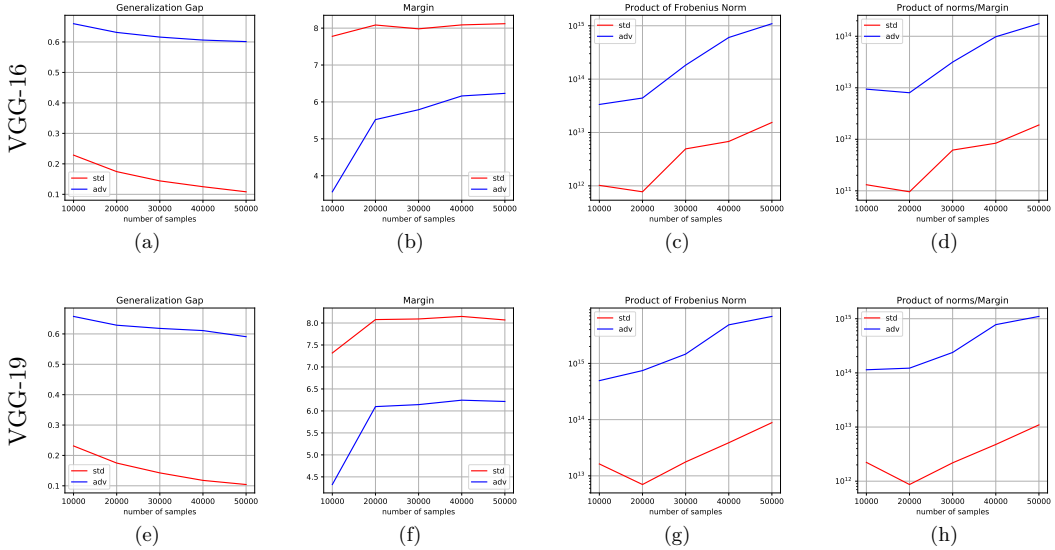


Figure 2.3: Product of the Frobenius norm in the experiments on CIFAR-10. The red lines are the results of standard training. The blue lines are the results of adversarial training. The first row is the experiments on VGG-16. The second row is the experiments on VGG-19. (a) and (e): Generalization gap. (b) and (f): Margin. (c) and (g): $\prod_{j=1}^{l} \|W_j\|_F$ of the neural networks. (d) and (h): $\prod_{j=1}^{l} \|W_j\|_F/\gamma$ of the neural networks.

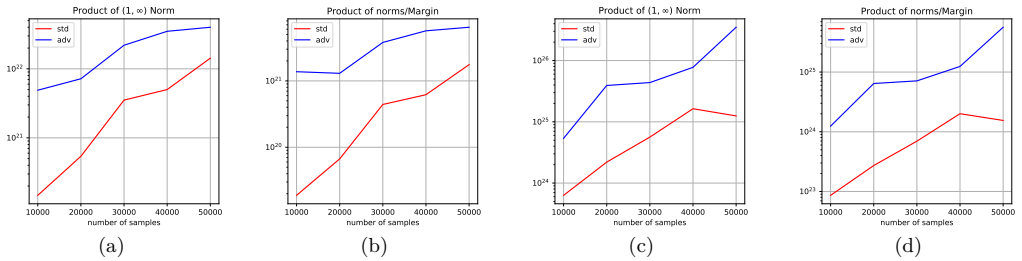


Figure 2.4: Product of the $\|\cdot\|_{1,\infty}$-Norm in the experiments on CIFAR-10. The red lines are the results of standard training. The blue lines are the results of adversarial training. (a) $\prod_{j=1}^{l} \|W_j\|_{1,\infty}$ of VGG-16 networks. (b) $\prod_{j=1}^{l} \|W_j\|_{1,\infty}/\gamma$ of VGG-16 networks. (c) $\prod_{j=1}^{l} \|W_j\|_{1,\infty}$ of VGG-19 networks. (d) $\prod_{j=1}^{l} \|W_j\|_{1,\infty}/\gamma$ of VGG-19 networks.

Experiments on CIFAR-100

Perfomance. In Table 2.3, we show the performance of standard training and adversarial training on CIFAR-100 using VGG-16 and 19 networks. We can see that using smaller number of training samples is unable to train an acceptable VGG-networks on CIFAR-100. Therefore it is hard use only 50000 training samples to study the trends of the weight norm using the experiments on CIFAR-100. We compare the product of weight norm between standard and adversarial training.

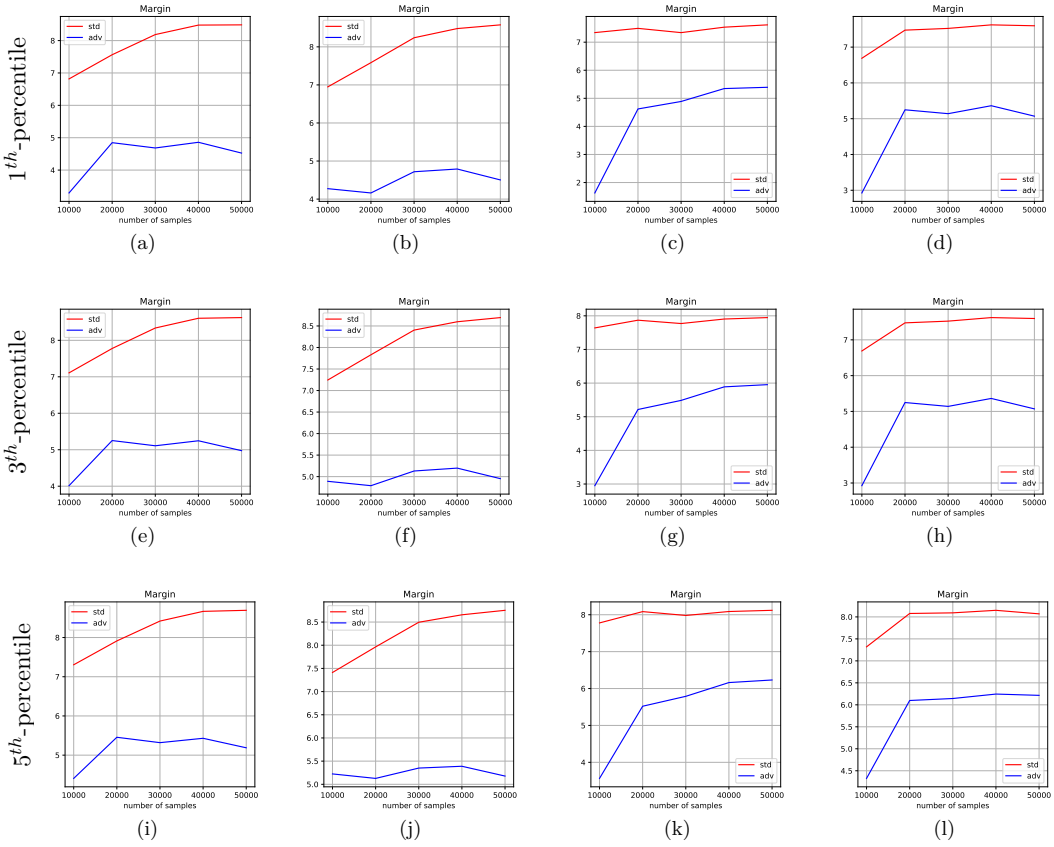


Figure 2.5: Ablation study of margins. The first to the 4^{th} rows are the experiments on VGG-11, 13, 16, and 19, respectively.

Product of Weight Norms. In Figure 2.6, we show the results of on training VGG-19-16 and VGG-19 on CIFAR-100. Similar to the experiments on CIFAR-10, we can see

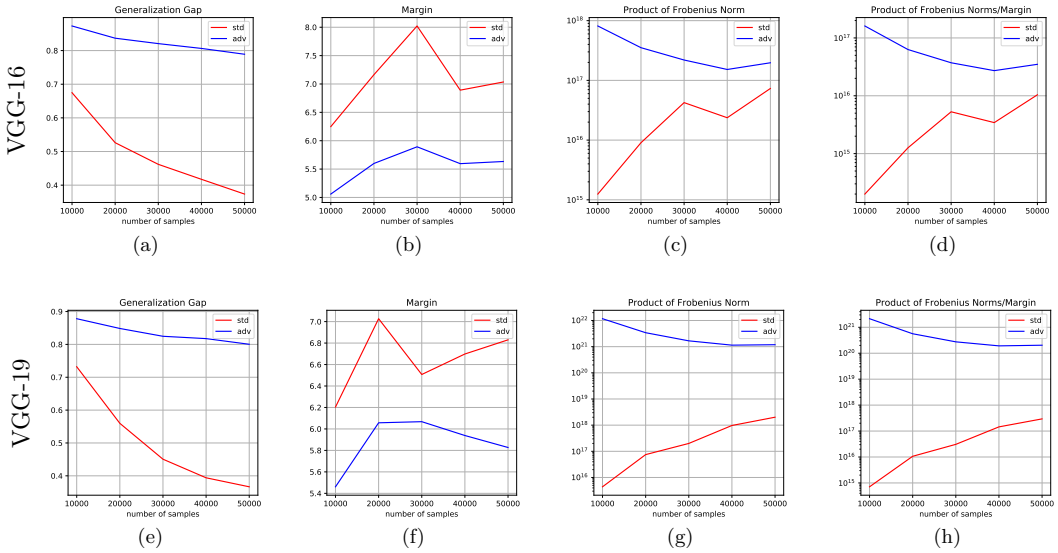
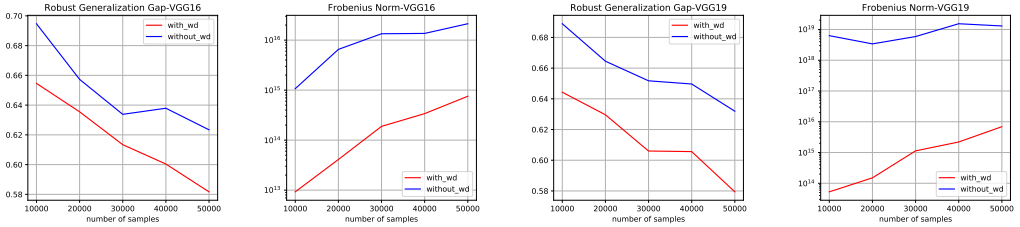


Figure 2.6: Product of the Frobenius norm in the experiments on VGG networks on CIFAR-100. The red lines are the results of standard training. The blue lines are the results of adversarial training. The first row are the experiments on VGG-16. The second row are the experiments on VGG-19. (a) and (e): Generalization gap. (b) and (f): Margin γ over training set. (c) and (g): $\prod_{j=1}^{l} \|W_j\|_F$ of the neural networks. (d) and (h): $\prod_{j=1}^{l} \|W_j\|_F / \gamma$ of the neural networks.

that the adversarially trained models have larger weight norm that that of the standard trained model.

Table 2.3: Accuracy of standard and adversarial training on CIFAR-100 using VGG-16 and 19 networks. For standard training model, we shows the clean accuracy. For adversarial training model, we show the robust accuracy against PGD attacks.

No. of Samples	10000	20000	30000	40000	50000
VGG-16-STD	0.26	0.44	0.54	0.60	0.63
VGG-16-ADV	0.12	0.15	0.17	0.18	0.19
VGG-19-STD	0.32	0.47	0.53	0.58	0.62
VGG-19-ADV	0.12	0.16	0.17	0.19	0.21



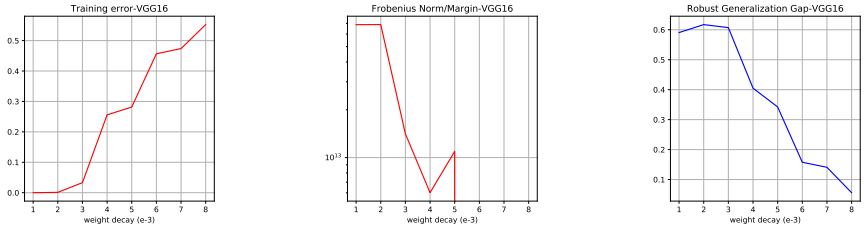
(a) (b) (c) (d)

Figure 2.7: Experiments on the effects of weight decay. (a) Robust generalization gap with or without weight decay on VGG-16. (b) Frobenius norm with or without weight decay on VGG-16. (c) Robust generalization gap with or without weight decay on VGG-19. (d) Frobenius norm with or without weight decay on VGG-19.

Weight Decay

The upper bounds of adversarial Rademacher complexity suggest adding a regularization term on the weights to improve generalization, which is essentially weight decay. In Figure 2.7, we provide the experiments of adversarial training with and without weight decay. In Figure 2.7 (a) and (c), we can see that adversarial training with weight decay has a smaller robust generalization gap. In Figure 2.7 (b) and (d), adversarial training with weight decay have a smaller product of weight norms. These experiments show the relationship between the robust generalization gap and the product of weight norms.

In Figure 2.8, we increase the weight decay ranging from 1×10^{-3} to 9×10^{-3}. In this range, the training becomes bad. From 6e-3, the margin becomes negative (which



(a) (b) (c)

Figure 2.8: Experiments on the effects of weight decay ranging from 1×10^{-3} to 9×10^{-3}. (a) Training error. (b) Frobenius norm. (c) Robust generalization gap.

means that training error is large). Then, weight norm/margin becomes negative. Form 9e-3, training totally fail, training error = test error = 90%. Therefore, the 'optimal' choice of weight decay (for weight norm) is from [1-5]e-3. We see the smallest one is 6.00848997e+12 (in the experiments using wd = 4e-3). This is still large than that of standard training, which is 1.8961088e+12.

□ **End of chapter.**

Chapter 3

Uniform Stability Analysis

Summary

In this chapter, we study the robust overfitting issue of adversarial training by using tools from uniform stability. One major challenge is that the outer function (as a maximization of the inner function) is nonsmooth, so the standard technique (*e.g.,* (Hardt et al., 2016)) cannot be applied. Our approach is to consider η-approximate smoothness: we show that the outer function satisfies this modified smoothness assumption with η being a constant related to the adversarial perturbation ϵ. Based on this, we derive stability-based generalization bounds for stochastic gradient descent (SGD) on the general class of η-approximate smooth functions, which covers the adversarial loss. Our results suggest that robust test accuracy decreases in ϵ when T is large, with a speed between $\Omega(\epsilon\sqrt{T})$ and $\mathcal{O}(\epsilon T)$. This phenomenon is also observed in practice. Additionally, we show that a few popular techniques for adversarial training (*e.g.,* early stopping, cyclic learning rate, and stochastic weight averaging) are stability-promoting in theory.

3.1 Introduction

In Chapter 2, we have studied the uniform convergence of adversarial training, which considers the generalization of the worst function in the function class

$$\sup_{f \in \mathcal{F}} [R_{\mathcal{D}}(f) - R_S(f)].$$

Uniform convergence analysis is algorithm-independent and focuses on the hypothesis function class. In this chapter, we take the algorithms into consideration. We study algorithm-based generalization gaps,

$$R_{\mathcal{D}}(\hat{w}) - R_S(\hat{w}),$$

through the lens of uniform algorithmic stability.

Uniform stability analysis (Bousquet & Elisseeff, 2002) in learning problems has been introduced to measure generalization gap instead of uniform convergence analysis such as classical VC-dimension (Vapnik & Chervonenkis, 2015) and Rademacher complexity (Bartlett & Mendelson, 2002). Generalization gap can be bounded in terms of uniform argument stability (UAS). Formally, UAS is the gap between the output parameters w of running an algorithm A on two datasets S and S' differ in at most one sample, denoted as $\delta(S, S') = \|w(S) - w(S')\|$. In a standard training problem with n training samples, assuming that the loss function is convex, L-Lipschitz and β-gradient Lipschitz, running stochastic gradient descent (SGD) with step size $\alpha \leq 1/\beta$ for T steps, the UAS is bounded by $\mathcal{O}(L\alpha T/n)$ (Hardt et al., 2016). Since the generalization gap is controlled by the number of samples n, this bound (at least partially) explains the good generalization of a standard training problem. Beyond (Hardt et al., 2016), Bassily et al. (2020) considered the case that the loss function is non-smooth. Without the β-gradient Lipschitz assumption, they showed that the UAS is bounded by $\mathcal{O}(L\alpha\sqrt{T} + L\alpha T/n)$ and provided a lower bound to show that the additional term $\mathcal{O}(L\alpha\sqrt{T})$ is unavoidable. In adversarial settings, two works have discussed the stability

of adversarial training to our knowledge.

Firstly, Farnia & Ozdaglar (2021) considered the stability of minimax problems. Their work includes a discussion on an algorithm called GDmax (gradient descent on the maximization of the inner problem), which can be viewed as a general form of adversarial training. It provides the stability of GDmax when the inner problem is further assumed to be strongly concave. Under the strongly concave assumption, the outer function is smooth. Then, the generalization bound $\mathcal{O}(L\alpha T/n)$ can be applied in this case. However, the inner problem is not strongly concave in practice. The bound $\mathcal{O}(L\alpha T/n)$ does not match the poor generalization of adversarial training observed in practice.

Another stability analysis of adversarial training is the work of (Xing et al., 2021b). Before they propose their algorithm, they use the stability bound $\mathcal{O}(L\alpha\sqrt{T} + L\alpha T/n)$ (Bassily et al., 2020) to characterize the issue of adversarial generalization. However, the bound is ϵ-independent. Let us Consider two cases, $\epsilon \to 0$ and ϵ is large (*e.g.,* 16/255). In the first case, adversarial training is very close to standard training and has good generalization ability. In the second case, adversarial generalization is very poor. The bound is the same in these two different cases. Therefore, it cannot provide an interpretation of adversarial generalization.

In summary, existing stability-based generalization bounds (Hardt et al., 2016; Bassily et al., 2020) have limited interpretation on adversarial generalization. In this chapter, we provide a new stability analysis of adversarial training using the notion of η-approximate smoothness. We first show that, under the same assumptions as (Hardt et al., 2016; Xing et al., 2021b), even though the outer function (adversarial loss) is non-smooth, it is η-approximately smooth (see Definition 3.2), where η is a constant linearly depending on the gradient Lipschitz of the inner function and the perturbation intensity ϵ. Then, we derive stability-based generalization bounds (Thm. 3.2 to 3.5) for stochastic gradient descent (SGD) on this general class of η-approximate smooth functions, which covers the adversarial loss. Our main result can be summarized in the following equation. Running SGD on adversarial loss for T steps with step size

$\alpha \leq 1/\beta$, the excess risk, which is the sum of generalization error and optimization error, satisfies

$$\text{Excess Risk} \leq \mathcal{E}_{gen} + \mathcal{E}_{opt} \leq \underbrace{\overbrace{L\eta T\alpha}^{\text{additional}} + \overbrace{\frac{2L^2T\alpha}{n} + \frac{D^2}{T\alpha} + L^2\alpha}^{\text{for standard training}}}_{\text{for adversarial training}}. \tag{3.1.1}$$

The excess risk of adversarial training has an additional term $L\eta T\alpha$. We also provide a lower bound of UAS of running SGD on adversarial loss. Our results suggest that robust test accuracy decreases in ϵ when T is large, with a speed between $\Omega(\epsilon\sqrt{T})$ and $\mathcal{O}(\epsilon T)$. This phenomenon is also observed in practice. It provides an understanding of robust overfitting from the perspective of uniform stability. Additionally, we show that a few popular techniques for adversarial training (*e.g.,* early stopping, cyclic learning rate, and stochastic weight averaging) are stability-promoting in theory and also empirically improve adversarial training. Experiments on SVHN, CIFAR-10, CIFAR-100, and ImageNet confirm our results.

The chapter is organized as follows. The first part is the technical part to derive stability bounds. The second part is to analyze robust overfitting. Specifically, in the first part, Sec. 3.2 introduces the preliminary knowledge about UAS. Sec. 3.3 provides the Lemma and properties of approximately smooth functions and Sec. 3.4 gives the stability bounds. In the second part, Sec. 3.5 analyzes the robust overfitting in the theoretical settings and Sec. 3.6 presents the experiments.

3.2 Preliminaries of Stability

To bound the generalization gap of a model $\hat{w} = A(S)$ trained by a randomized algorithm A, we employ the following notion of *uniform stability*.

Definition 3.1. *A randomized algorithm A is* ε-uniformly stable *if for all data sets*

$S, S' \in \mathcal{Z}^n$ such that S and S' differ in at most one example, we have

$$\sup_z \mathbb{E}_A \left[h(A(S); z) - h(A(S'); z) \right] \leq \varepsilon \,. \tag{3.2.1}$$

Here, the expectation is taken only over the internal randomness of A. We recall the important theorem that uniform stability implies *generalization in expectation* (Hardt et al., 2016).

Theorem 3.1 (Generalization in expectation)**.** *Let A be ε-uniformly stable. Then, the expected generalization gap satisfies*

$$|\mathcal{E}_{gen}| = |\mathbb{E}_{S,A}[R_{\mathcal{D}}[A(S)] - R_S[A(S)]]| \leq \varepsilon \,.$$

Therefore, we turn to the properties of iterative algorithms that control their uniform stability.

3.3 Stability of Adversarial Training

Adversarial Surrogate Loss. In adversarial training, we consider the following surrogate loss

$$h(w; z) = \max_{\|z - z'\|_p \leq \epsilon} g(w; z'), \tag{3.3.1}$$

where $g(w; z)$ is the loss function of the standard counterpart, $\|\cdot\|_p$ is the ℓ_p-norm, $p \geq 1$. Usually, g can also be written in the form of $\ell(f_w(\boldsymbol{x}); y)$, where f_w is the neural network to be trained and $(\boldsymbol{x}, y)$ is the input-label pair. We assume the loss function g satisfies the following smoothness assumption.

Assumption 3.1. *The function g satisfies the following Lipschitzian smoothness conditions:*

$$\|g(w_1, z) - g(w_2, z)\| \leq L\|w_1 - w_2\|,$$
$$\|\nabla_w g(w_1, z) - \nabla_w g(w_2, z)\| \leq L_w\|w_1 - w_2\|,$$
$$\|\nabla_w g(w, z_1) - \nabla_w g(w, z_2)\| \leq L_z\|z_1 - z_2\|_p,$$

where $\|\cdot\|$ *is Euclidean norm.*

Assumption 3.1 assumes that the loss function is smooth, which are also used in the stability literature (Farnia & Ozdaglar, 2021; Xing et al., 2021b), as well as the convergence analysis literature (Wang et al., 2019; Liu et al., 2020). While ReLU activation function is non-smooth, recent works (Allen-Zhu et al., 2019; Du et al., 2019) showed that the loss function of over-parameterized DNNs is semi-smooth. It helps justify Assumption 3.1. Under Assumption 3.1, the loss function of adversarial training satisfies the following Lemma (Liu et al., 2020).

Lemma 3.1. *Let* h *be the adversarial loss defined in Eq. (3.3.1) and* g *satisfies Assumption 3.1.* $\forall w_1, w_2$ *and* $\forall z \in \mathcal{Z}$*, the following properties hold.*

1. *(Lipschitz function.)* $\|h(w_1, z) - h(w_2, z)\| \leq L\|w_1 - w_2\|$.
2. *For all subgradient* $d(w, z) \in \partial_w h(w, z)$*, we have* $\|d(w_1, z) - d(w_2, z)\| \leq L_w\|w_1 - w_2\| + 2L_z\epsilon$.

If we further assume that $g(w, z)$ is μ-strongly concave in z, the adversarial surrogate loss $h(w, z)$ is also smooth (Sinha et al., 2017). Therefore, the uniform stability of adversarial training follows (Hardt et al., 2016; Farnia & Ozdaglar, 2021), see Sec. 3.9. However, $g(w, z)$ is non-strongly concave in practice. The above results provide limited explanations of the poor generalization of adversarial training. We discuss the generalization properties of adversarial training under Lemma 3.1.2.

3.3.1 Basic Properties of Approximate Smoothness

To simplify the notation, we use $h(w)$ as a shorthand notation of $h(w, z)$. To simplify the argument, we consider differentiable function h. The results can be extended to non-differentiable cases. Lemma 3.1 motivates us to analyze a function with the following modified smoothness assumption, which we call approximate smoothness assumption.

Definition 3.2. *Let $\beta > 0$ and $\eta > 0$. We say a differentiable function $h(w)$ is η-approximately β-gradient Lipschitz, if $\forall w_1$ and w_2, we have*

$$\|\nabla h(w_1) - \nabla h(w_2)\| \leq \beta\|w_1 - w_2\| + \eta.$$

In definition 3.2, η controls the smoothness of the loss function $h(\cdot)$. When $\eta = 0$, the function h is gradient Lipschitz. When $\eta \to +\infty$, h is a general non-smooth function. As our discussion before, the adversarial surrogate loss is $2L_z\epsilon$-approximately smooth. As far as we know, this assumption is rarely discussed in the optimization literature since it cannot improve the convergence rate from a general non-smooth assumption. But it affects uniform stability, as we will discuss later. We need to develop the basic properties of approximate smoothness first.

Lemma 3.2. *Assume that the function h is η-approximately β-gradient Lipschitz. $\forall w_1, w_2$ and $\forall z \in \mathcal{Z}$, the following properties hold.*

1. *(η-approximate descent lemma.)*

$$h(w_1) - h(w_2) \leq \nabla h(w_2)^T(w_1 - w_2) + \frac{\beta}{2}\|w_1 - w_2\|^2 + \eta\|w_1 - w_2\|.$$

2. *(η-approximately co-coercive.) Assume in addition that $h(w, z)$ is convex in w for all $z \in \mathcal{Z}$. Let $[\cdot]_+ = \max(0, \cdot)$. We have*

$$\langle \nabla h(w_1) - \nabla h(w_2), w_1 - w_2 \rangle \geq \frac{1}{\beta}\left[[\|\nabla h(w_1) - \nabla h(w_2)\| - \eta]_+\right]^2.$$

We defer the proof to Sec. 3.8. Note that the loss function is L-Lipschitz for every example z, we have $\mathbb{E}|h(w_1, z) - h(w_2, z)| \leq L\mathbb{E}\|w_1 - w_2\|$, for all $z \in \mathcal{Z}$. To obtain the stability generalization bounds, we need to analysis the difference $\|w_1^T - w_2^T\|$, where w_1^T and w_2^T are the outputs of running SGD on adversarial surrogate loss for T iterations on two datasets with only one different sample. Next we provide the recursive bounds under the approximate smoothness assumption.

Algorithms. We consider the stochastic gradient descent on the adversarial surrogate loss. i.e.,

$$w^{t+1} = w^t - \alpha_t \nabla_w h(w^t, z_{i_t}), \tag{3.3.2}$$

where α_t is the step size in iteration t, z_{i_t} is the sample chosen in iteration t. We consider two popular schemes for choosing the examples' indices i_t. *Sampling with replacement*: One is to pick $i_t \sim \text{Uniform}\{1, \cdots, n\}$ at each step. *Fixed permutation*: The other is to choose a random permutation over $\{1, \cdots, n\}$ and cycle through the examples repeatedly in the order determined by the permutation. Our results hold for both variants.

Properties of Update Rules. We define $G_{\alpha,z}(w) = w - \alpha \nabla h(w, z)$ be the update rule of SGD. The following lemma holds.

Lemma 3.3. *Assume that the function h is η-approximately β-gradient Lipschitz. $\forall w_1, w_2$ and $\forall z \in \mathcal{Z}$,we have*

1. *($\alpha\eta$-approximately $(1+\alpha\beta)$-expansive.)* $\|G_{\alpha,z}(w_1) - G_{\alpha,z}(w_2)\| \leq (1+\alpha\beta)\|w_1 - w_2\| + \alpha\eta$.

2. *($\alpha\eta$-approximately non-expansive.) Assume in addition that $h(w, z)$ is convex in w for all $z \in \mathcal{Z}$, for $\alpha \leq 1/\beta$, we have* $\|G_{\alpha,z}(w_1) - G_{\alpha,z}(w_2)\| \leq \|w_1 - w_2\| + \alpha\eta$.

3. *($\alpha\eta$-approximately $(1-\alpha\gamma)$-contractive.) Assume in addition that $h(w, z)$ is γ-strongly convex in w for all $z \in \mathcal{Z}$, for $\alpha \leq 1/\beta$, we have* $\|G_{\alpha,z}(w_1) - G_{\alpha,z}(w_2)\| \leq (1-\alpha\gamma)\|w_1 - w_2\| + \alpha\eta$.

The proof of Lemma 3.3 is based on Lemma 3.2 and is deferred to Sec. 3.8. Lemma 3.3 provides the recursive distance $\|w_1 - w_2\|$ from iteration t to $t+1$. Based on this, we can recursively derive the distance $\|w_1^T - w_2^T\|$. Then, we can obtain the stability generalization bounds.

3.4 Stability Generalization Bounds

In the previous section, we have discussed the properties of approximate smoothness and developed tools we need to use. In this section, we discuss the stability bounds.

3.4.1 Convex Optimization

We first consider the case that $h(w, z)$ is convex in w for all $z \in \mathcal{Z}$.

Theorem 3.2. *Assume that $h(w, z)$ is convex, L-Lipschitz, and η-approximately β-gradient Lipschitz in w for all given $z \in \mathcal{Z}$. Suppose that we run SGD with step sizes $\alpha_t \leq 1/\beta$ for T steps. Then,*

$$\mathcal{E}_{gen} = \mathbb{E}[R_{\mathcal{D}}(w^T) - R_S(w^T)] \leq L\left(\eta + \frac{2L}{n}\right)\sum_{t=1}^{T}\alpha_t. \tag{3.4.1}$$

The proof is mainly based on Lemma 3.3 that the update rule is approximately non-expansive in this case. We defer it to Sec. 3.8. We have shown that the adversarial surrogate loss is $2L_z\epsilon$-approximately L_w-gradient Lipschitz. Let $\eta = 2L_z\epsilon$ in Eq. (3.4.1), we directly obtain the stability bounds for adversarial training. In practice, the solution of the inner problem is sub-optimal. Let $\Delta\epsilon$ be the maximum error between the optimal and sub-optimal attacks in each iteration. We have the following Corollary.

Corollary 3.1 (Uniform stability for sub-optimal attacks adversarial training.)**.** *Under Assumption 3.1, assume in addition that $g(w, z)$ is convex in w for all given $z \in \mathcal{Z}$. Suppose that we run adversarial training with step sizes $\alpha_t \leq 1/L_w$ for T steps. Then, adversarial training satisfies uniform stability with*

$$\mathcal{E}_{gen} \leq \left(2LL_z(\epsilon + \Delta\epsilon) + \frac{2L^2}{n}\right)\sum_{t=1}^{T}\alpha_t \leq \mathcal{O}\left(L(L_z\epsilon + \frac{L}{n})\sum_{t=1}^{T}\alpha_t\right). \tag{3.4.2}$$

Remark: Corollary 3.1 shows that adversarial training with weak attacks (large $\Delta\epsilon$) have worse adversarial generalization than that with strong attacks. This is also observed in practice. However, $\Delta\epsilon$ is at most 2ϵ, the upper bound of AT with different

attacks have the same order. It might be due to the weakness of Assumption 3.1 or uniform stabilty framework.

Interpreting Robust Generalization. If we let $\epsilon = 0$ in Eq. (3.4.2) , it reduced to the generalization bound in (Hardt et al., 2016) for standard training. Therefore, the additional generalization error of adversarial training comes from the additional term $L_z\epsilon$. The global gradient Lipschitz L_z with respect to z plays an important role in generalization. Even though the perturbation ϵ is small, it is amplified by the Lipschitz L_z and finally hurts the robust generalization.

3.4.2 Further Discussion on the Generalization Bounds

We first provide a lower bound. Then we compare our bounds with the existing bounds in Table 3.1.

Theorem 3.3 (Lower Bound)**.** *There exists functions $h(w, z)$, s.t. h is convex, L-Lipschitz, and η-approximately β-gradient Lipschitz in w for all given $z \in \mathcal{Z}$. Exists S and S' differ in one sample. Suppose that we run SGD with step fixed step sizes $\alpha \leq 1/\beta$ for T steps. Then,*

$$\mathbb{E}[\delta(S, S')] \geq \Omega\left(\eta\alpha\sqrt{T} + \frac{L\alpha T}{n}\right). \tag{3.4.3}$$

Table 3.1: Comparison of the upper and lower bounds of $\mathbb{E}[\delta(S, S')]$. Comparing with the previous results, we replace L by η and provide the matching lower bound in η.

	Assumption	Upper bounds	Lower bounds
Farnia & Ozdaglar (2021)	convex-strongly concave	$\mathcal{O}(LT\alpha/n)$	$\Omega(LT\alpha/n)$
Xing et al. (2021b)	convex-nonconcave	$\mathcal{O}(L\sqrt{T}\alpha + LT\alpha/n)$	$\Omega(\sqrt{T}\alpha + LT\alpha/n)$
Ours	convex-nonconcave	$\mathcal{O}(\eta T\alpha + LT\alpha/n)$	$\Omega(\eta\sqrt{T}\alpha + LT\alpha/n)$

Comparison with the Existing UAS Bounds. Compared with the work of (Farnia & Ozdaglar, 2021), they assume that the inner problem is strongly concave. Thus the bounds are not comparable. Strongly concave is a strong assumption in practice. Therefore, the work of (Xing et al., 2021b) and our analysis focus on the

nonconcave cases. Comparing with the bound $\mathcal{O}(L\sqrt{T}\alpha+LT\alpha/n)$, our bound captures a critical aspect of adversarial generalization bound: ϵ-dependent. As observed in practice, the adversarial generalization gap reduces to the standard generalization gap as $\epsilon \to 0$. Our bound consists with this observation. On the contrary, the bound $\mathcal{O}(L\sqrt{T}\alpha + LT\alpha/n)$ is very large when $\epsilon \to 0$. Additionally, we provide a matching lower bound w.r.t η. The comparison is provided in Table 3.1.

Comparison with the Work of (Xing et al., 2021b). The work of (Xing et al., 2021b) argued that the max function is not smooth even though the standard counterpart is smooth. Therefore, they followed the bound in non-smooth cases (Bassily et al., 2020). Then, they aimed to solve the non-smooth issue. They design a noise-injected algorithm and show its effectiveness in tackling the non-smooth issue. Our work focus on providing better bounds to interpret robust overfitting.

3.4.3 Non-convex Optimization and Strongly Convex Optimization

Next, we consider the case that the loss function h is general non-convex and strongly convex. By Lemma 3.3, we have

Theorem 3.4. *Assume that $h(w,z)$ is L-Lipschitz, and η-approximately β-gradient Lipschitz in w for all given $z \in \mathcal{Z}$. Assume in addition that $0 \le h(w,z) \le B$ for all w and z. Suppose that we run SGD with diminishing step sizes $\alpha_t \le 1/(\beta t)$ for T steps. Then*

$$\mathcal{E}_{gen} \le \frac{BL_w + (2L^2 + L\eta n)T}{\beta(n-1)}. \tag{3.4.4}$$

Theorem 3.5. *Assume that $h(w,z)$ is γ-strongly convex, L-Lipschitz, and η-approximately β-gradient Lipschitz in w for all given $z \in \mathcal{Z}$. Suppose that we run SGD with step sizes $\alpha_t \le 1/\beta$ for T steps. Then*

$$\mathcal{E}_{gen} = \mathbb{E}[R_{\mathcal{D}}(w^T) - R_S(w^T)] \le \frac{L\eta}{\gamma} + \frac{2L^2}{\gamma n}. \tag{3.4.5}$$

Remark: The bound in non-convex cases provides a similar interpretation of robust generalization to the analysis in the convex case. In uniform stability analysis, whether

the loss function is convex or non-convex does not give a major difference. Therefore, we provide the analysis of the non-convex case in Sec. 3.9. We also provide our convergence analysis of running SGD on a η-approximate smoothness, non-convex function in Theorem 3.9. Strongly convex is a strong assumption. We leave the analysis of the bound in strongly convex cases in Sec. 3.9.

3.5 Excess Risk Minimization

Based on the risk decomposition, we have Excess Risk $\leq \mathcal{E}_{gen} + \mathcal{E}_{opt}$, we need to minimize $\mathcal{E}_{gen} + \mathcal{E}_{opt}$ to achieve better performance. Per our previous discussion, whether the loss function is convex or non-convex does not give a major difference in stability analysis. We study the convex case in this section. We leave the discussion on the non-convex and strongly convex cases in Sec. 3.9 and 3.9, respectively. We first introduce the optimization error.

Optimization Analysis. The convergence analysis of SGD on a L-Lipschitz, convex function is discussed in (Nemirovski et al., 2009). The convergence rate cannot be improved if we further assume that the function h is gradient Lipschitz. Therefore, a weaker condition, approximately gradient Lipschitz, cannot improve the convergence rate. We use the following convergence error bound (adopted from (Nemirovski et al., 2009)) for the optimization error of both adversarial training and standard training.

Theorem 3.6. *Assume that $h(w, z)$ is L-Lipschitz and convex in w for all given $z \in \mathcal{Z}$. Let $D = \|w^0 - w^*\|$, where w^0 is the initialization of SGD. Suppose that we run SGD with step sizes α_t for T steps. Then, $\exists k \leq T$, s.t.*

$$\mathcal{E}_{opt}(w^k) \leq \frac{D^2 + L^2 \sum_{t=1}^{T} \alpha_t^2}{\sum_{t=1}^{T} \alpha_t}. \tag{3.5.1}$$

If we let $\alpha_t = 1/\sqrt{T}$, we have $\mathcal{E}_{opt} \leq \mathcal{O}(1/\sqrt{T})$, which is the convergence rate of SGD on convex function. Since we need to consider the generalization and optimization errors simultaneously, we keep the α_t in Theorem 3.6.

Generalization-Optimization Trade-off. We have now discussed the optimization and generalization errors of adversarial training. We aim to find the optimal trade-off between generalization and optimization in terms of α_t and T. However, finding the optimal α_t and T simultaneously is challenging. We consider the settings we use in practice.

Fixed Step Size. We first consider the simplest case, the step size α is fixed. Then, combining Eq. (3.4.1) and Eq. (3.5.1), we have

$$\mathcal{E}_{gen} + \mathcal{E}_{opt} \leq \underbrace{\overbrace{L\eta T\alpha}^{additional} + \overbrace{\frac{2L^2T\alpha}{n} + \frac{D^2}{T\alpha} + L^2\alpha}^{\text{for standard training}}}_{\text{for adversarial training}}. \tag{3.5.2}$$

Interpretation of Robust Overfitting. In standard training, overfitting is rarely observed in practice. The optimization error and generalization error are both small. In Eq. (3.5.2), the second to the fourth terms are for standard training. The second term is controlled by the number of samples n, which is small if we have sufficient training samples. The third term is controlled by T, and the last term is fixed given a small α. This bound partially explains the good performance of standard training. However, we have an additional term $L\eta T\alpha$ for excess risk in adversarial training. Then, after a particular iteration that $L\eta T\alpha$ dominates Eq. (3.5.2), robust overfitting appears. This is consistent with the training procedure in practice. Therefore, the $\eta = 2L_z\epsilon$ approximate smoothness of adversarial loss provides a possible explanation of robust overfitting. To achieve better performance, we need to stop training the model earlier.

Early Stopping. It is shown that early stopping is an important training technique for adversarial training (Rice et al., 2020). In Eq. (3.5.2), if we optimize the right-hand-side with respect to T, we have

$$T^* = \frac{\|w^0 - w^*\|}{\alpha\sqrt{L\eta + 2L^2/n}}, \quad \mathcal{E}_{gen} + \mathcal{E}_{opt} \leq 2\sqrt{L\eta + \frac{2L^2}{n}}D + L^2\alpha.$$

Therefore, it is the best to stop training at T^*. However, T^* is unknown in practice. It is important to select stopping criteria. For example, we can use a validation set to determine when to stop.

Varying Step Size. We discuss one popular varying step size schedule, cyclic learning rate, which is also called super-converge learning rate for adversarial training. It is shown that it can speed up adversarial training with fewer epochs (Wong et al., 2020). The Super-converge learning rate follows the following rules. In the first phase (warm-up), the step size increase from 0 to α' linearly. In the second phase (cold down), the step size decreases back to 0 linearly. It is unclear (to our knowledge) why this schedule can speed up convergence in optimization theory. But the generalization part can partially be explained by UAS. If we set $\alpha' = 2\alpha$, it is easy to check that $\mathcal{E}_{gen} \leq (L\eta + 2L^2/n)T\alpha$ in this case, which is the same as the bound in the fixed learning rate case. Notice that the cyclic learning rate usually requires fewer steps T to converge. Then, the generalization gap is smaller. Cyclic learning rate can be viewed as another form of early stopping from the perspective of UAS.

Stochastic Weight Averaging. SWA is also a useful training technique for adversarial training (Hwang et al., 2021). Instead of using the last checkpoint, SWA suggests using the average of the checkpoints for inference. It is shown that SWA can find a model with better generalization since it leads to wider minima (Izmailov et al., 2018). Below we study SWA from the perspective of UAS.

Theorem 3.7. *Assume that $h(w, z)$ is convex, L-Lipschitz, and η-approximately β-gradient Lipschitz in w for all given $z \in \mathcal{Z}$. Suppose that we run SGD with step sizes $\alpha_t \leq 1/\beta$ for T steps. Let $\bar{w}$ be the average of the trajectory. Then,*

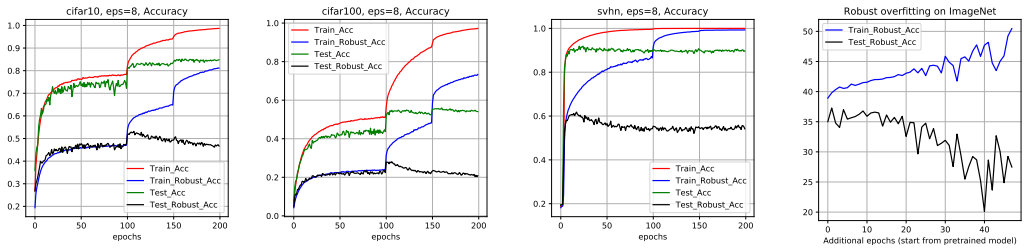
$$\mathcal{E}_{gen}(\bar{w}) \leq \left(\frac{L\eta}{2} + \frac{L^2}{n}\right)\sum_{t=1}^{T}\alpha_t, \quad \mathcal{E}_{opt}(\bar{w}) \leq \frac{\|w^0 - w^*\|^2 + L^2\sum_{t=1}^{T}\alpha_t^2}{\sum_{t=1}^{T}\alpha_t}. \tag{3.5.3}$$

In words, SWA reduces the generalization error bound to one-half of the one without

SWA. But the training error bound remains unchanged.

3.6 Experiments

Training Settings. We mainly consider the experiments on CIFAR-10 (Krizhevsky et al., 2009), CIFAR-100, and SVHN (Netzer et al., 2011). We also provide one experiment on ImageNet (Deng et al., 2009). For the first three datasets, we conduct the experiments on training PreActResNet-18, which follows (Rice et al., 2020), For the experiment on ImageNet, we use ResNet-50 (He et al., 2016), following the experiment of (Madry et al., 2017). For the inner problems, we adopt the ℓ_∞ PGD adversarial training in (Madry et al., 2017), the step size in the inner maximization is set to be $\epsilon/4$ on CIFAR-10 and CIFAR100 and is set to be $\epsilon/8$ on SVHN. Weight decay is set to be 5×10^{-4}. Additional experiments are provided in Sec. 3.10. [1]



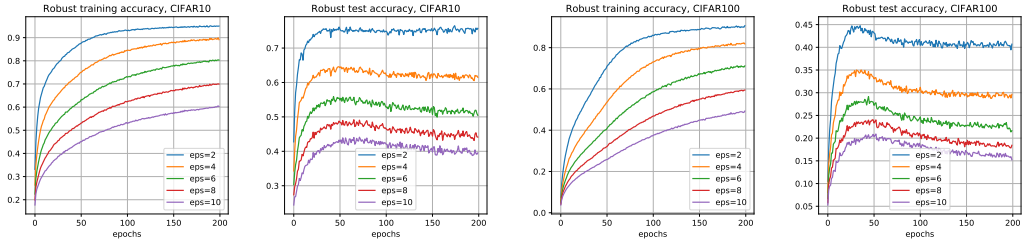
(a) (b) (c) (d)

Figure 3.1: Robust overfitting in the experiments on (a) CIFAR-10, (b) CIFAR-100, (c) SVHN and (d) ImageNet.

Robust Overfitting on Common Dataset. In Fig. 3.1 (a), (b), and (c), we show the experiments on the piece-wise learning rate schedule, which is 0.1 over the first 100 epochs, down to 0.01 over the following 50 epochs, and finally be 0.001 in the last 50 epochs, on CIFAR-10, CIFAR-100, and SVHN. Experiments on different ϵ are shown in Sec. 3.10. *Robust Overfitting on ImageNet.* We provide one experiment on ImageNet in Fig. 3.1 (d). We start from a pre-trained model from Madry's Lab and

[1] https://github.com/JiancongXiao/Stability-of-Adversarial-Training

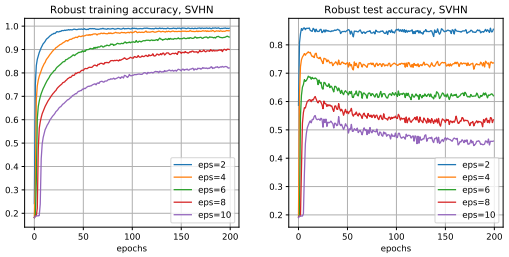
keep running 50 more epochs. *Robust overfitting* can be observed in these experiments. After a particular epoch (around the 100^{th} epoch), the robust training accuracy is still increasing, but the robust test accuracy starts to decrease.



(a) (b) (c) (d)

Figure 3.2: Experiments of adversarial training on CIFAR-10 and CIFAR-100 with a fixed learning rate. (a) Robust training accuracy on CIFAR-10. (b) Robust test accuracy on CIFAR-10. (c) Robust training accuracy on CIFAR-100. (d) Robust test accuracy on CIFAR-100. ϵ are set to be 2, 4, 6, 8, and 10.

Fixed Step Size. To better understand robust overfitting and match the theoretical settings (in Eq. (3.5.2)), we consider the fixed learning rate schedule. In Fig. 3.2, we show the experiments of adversarial training using a fixed learning rate 0.01. The perturbation intensity ϵ is set to be 2, 4, 6, 8, and 10. respectively. Fig. 3.2 (a) and (b) show the experiments on CIFAR-10. Fig. 3.2 (c) and (d) show the experiments on CIFAR-100, respectively. Fig. 3.3 shows the experiments on SVHN.



(a) (b)

Figure 3.3: Experiments of adversarial training on SVHN with a fixed learning rate. (a) Robust training accuracy. (b) Robust test accuracy.

Generalization Error Dominates Training Error. In the robust overfitting phase, the robust generalization error dominates the robust training error. This phenomenon corresponds to the Eq. (3.5.2) that the first term dominates the other terms with large

T.

Robust Test Accuracy Decreases in ϵ. Comparing different ϵ in Fig. 3.2, we can see that the robust test accuracy decreases faster when ϵ is larger. This corresponds to the robust overfitting rule in the theoretical setting that the test performance decreases in ϵ. These phenomena are similar in theoretical and practical settings. Therefore, the stability analysis provides a different perspective on understanding robust overfitting.

Robust Test Accuracy Decreases in a Rate between $\Omega(\epsilon\sqrt{T})$ and $\mathcal{O}(\epsilon T)$. If ϵ is small, *e.g.*, $\epsilon = 2$, the decrease rate is close to $\mathcal{O}(\epsilon T)$. If ϵ is large, the decrease rate is more likely to be $\Omega(\epsilon\sqrt{T})$. This is also the gap between the upper bound and lower bound in Sec. 3.4.

3.7 Conclusion

Limitations and Future Work. Firstly, in Fig. 3.2, we can see that the decrease rate of robust overfitting is close to $\mathcal{O}(T)$ when ϵ is small and is close to $\Omega(\sqrt{T})$ when ϵ is large. One possible direction is to figure out the relation. Secondly, one might improve adversarial training by controlling L_z. L_z depends on the loss, the network architecture, and the dataset. One possible direction is to design a smoother loss or smooth activation function (e.g., SiLU) for a lower L_z. Notice that L_z is uniform for all w. If we view $L_z(w)$ locally with respect to w, we might use an (approximated) second-order penalty term on its magnitude to control it.

In this chapter, we show that the adversarial loss satisfies η-approximate smoothness, and we derive stability-based generalization bounds on this general class of η-approximate smooth functions. Our bounds give a different perspective on understanding robust overfitting. The robust test accuracy decreases in η, and experimental results confirm this phenomenon. We think our work will inspire more theoretical and empirical research to improve adversarial training.

3.8 Proof of the Theorem

Proof of Theorem 3.1

The proof can be found in (Hardt et al., 2016). We provide the proof here for reference. Denote by $S = (z_1, \ldots, z_n)$ and $S' = (z'_1, \ldots, z'_n)$ two independent random samples and let $S^{(i)} = (z_1, \ldots, z_{i-1}, z'_i, z_{i+1}, \ldots, z_n)$ be the sample that is identical to S except in the i'th example where we replace z_i with z'_i. With this notation, we get that

$$\begin{aligned}
\mathbb{E}_S \mathbb{E}_A \left[R_S[A(S)]\right] &= \mathbb{E}_S \mathbb{E}_A \left[\frac{1}{n}\sum_{i=1}^{n} h(A(S); z_i)\right] \\
&= \mathbb{E}_S \mathbb{E}_{S'} \mathbb{E}_A \left[\frac{1}{n}\sum_{i=1}^{n} h(A(S^{(i)}); z'_i)\right] \\
&= \mathbb{E}_S \mathbb{E}_{S'} \mathbb{E}_A \left[\frac{1}{n}\sum_{i=1}^{n} h(A(S); z'_i)\right] + \delta \\
&= \mathbb{E}_S \mathbb{E}_A \left[R_{\mathcal{D}}[A(S)]\right] + \delta,
\end{aligned}$$

where we can express δ as

$$\delta = \mathbb{E}_S \mathbb{E}_{S'} \mathbb{E}_A \left[\frac{1}{n}\sum_{i=1}^{n} h(A(S^{(i)}); z'_i) - \frac{1}{n}\sum_{i=1}^{n} h(A(S); z'_i)\right].$$

Furthermore, taking the supremum over any two data sets S, S' differing in only one sample, we can bound the difference as

$$|\delta| \leq \sup_{S,S',z} \mathbb{E}_A \left[h(A(S); z) - h(A(S'); z)\right] \leq \epsilon,$$

by our assumption on the uniform stability of A. The claim follows. □

Proof of Lemma 3.1

Let the adversarial examples for parameter w_1 and w_2 be

$$z_1 \in \arg \max_{\|z-z'\|_p \leq \epsilon} g(w_1, z')$$

$$z_2 \in \arg \max_{\|z-z'\|_p \leq \epsilon} g(w_2, z'),$$

then we have

$$\begin{aligned}
&\|h(w_1, z) - h(w_2, z)\| \\
=&|g(w_1, z_1) - g(w_2, z_2)| \\
\leq& \max\{|g(w_1, z_1) - g(w_2, z_1)|, |g(w_1, z_2) - g(w_2, z_2)|\} \\
\leq& L\|w_1 - w_2\|,
\end{aligned}$$

where the first inequality is based on the fact that $g(w_1, z_1) \geq g(w_1, z_2)$ and $g(w_2, z_2) \geq g(w_2, z_1)$, the second inequality is based on Assumption 3.1. This proves Lemma 3.1.1.

For all subgradient $d(w, z) \in \partial_w h(w, z)$, we have

$$\begin{aligned}
&\|d(w_1, z) - d(w_2, z)\| \\
=&\|\nabla_w g(w_1, z_1) - \nabla_w g(w_2, z_2)\| \\
\leq&\|\nabla_w g(w_1, z_1) - \nabla_w g(w_2, z_1)\| + \|\nabla_w g(w_2, z_1) - \nabla_w g(w_2, z_2)\| \\
\leq& L_w\|w_1 - w_2\| + L_z\|z_1 - z_2\|_p \\
\leq& L_w\|w_1 - w_2\| + L_z[\|z_1 - z\|_p + \|z - z_2\|_p] \\
\leq& L_w\|w_1 - w_2\| + 2L_z\epsilon
\end{aligned}$$

where the first and the third inequality is due to triangle inequality, the second inequality is based on Assumption 3.1. This proves the second inequality (non-gradient Lipschitz) in Lemma 3.1. □

Proof of Lemma 3.2

Proof of Lemma 3.2.1 (η-approximate descent Lemma).

Let $\tilde{w}$ be a point in the line segment of w_1 and w_2, $\tilde{w}(u) = w_2 + u(w_1 - w_2)$, then

$$
\begin{aligned}
& h(w_1) - h(w_2) \\
=& \int_0^1 \langle w_1 - w_2, \nabla_w h(\tilde{w}(u)) \rangle du \\
=& \int_0^1 \langle w_1 - w_2, \nabla_w h(w_2) + \nabla_w h(\tilde{w}(u)) - \nabla_w h(w_2) \rangle du \\
=& \langle \nabla_w h(w_2), w_1 - w_2 \rangle + \int_0^1 \langle w_1 - w_2, \nabla_w h(\tilde{w}(u)) - \nabla_w h(w_2) \rangle du \\
\leq& \langle \nabla_w h(w_2), w_1 - w_2 \rangle + \int_0^1 \|w_1 - w_2\| \|\nabla_w h(\tilde{w}(u)) - \nabla_w h(w_2)\| du \\
\leq& \langle \nabla_w h(w_2), w_1 - w_2 \rangle + \int_0^1 \|w_1 - w_2\| [\beta \|\tilde{w}(u) - w_2\| + \eta] du \\
=& \langle \nabla_w h(w_2), w_1 - w_2 \rangle + \int_0^1 \|w_1 - w_2\| [\beta u \|w_1 - w_2\| + \eta] du \\
=& \langle \nabla_w h(w_2), w_1 - w_2 \rangle + \beta \|w_1 - w_2\|^2 \int_0^1 u du + \eta \|w_1 - w_2\|] \\
=& \langle \nabla_w h(w_2), w_1 - w_2 \rangle + \frac{\beta}{2} \|w_1 - w_2\|^2 + \eta \|w_1 - w_2\|.
\end{aligned}
$$

□

Proof of Lemma 3.2.2 (η-approximate co-coercive).

By Lemma 3.2.1 (η-approximate descent Lemma), we have

$$
h(w_1) \leq h(w_2) + \langle \nabla_w h(w_2), w_1 - w_2 \rangle + \frac{\beta}{2} \|w_1 - w_2\|^2 + \eta \|w_1 - w_2\|.
$$

Let w^* be a minimizer of h, then

$$
\begin{aligned}
h(w^*) = \inf_{w_1} h(w_1) &\leq \inf_{w_1} \left(h(w_2) + \langle \nabla_w h(w_2), w_1 - w_2 \rangle + \frac{\beta}{2} \|w_1 - w_2\|^2 + \eta \|w_1 - w_2\| \right) \\
&= \inf_{\|v\|=1} \inf_{t \geq 0} \left(h(w_2) + t \nabla_w h(w_2)^T v + \frac{\beta t^2}{2} + \eta t \right),
\end{aligned}
$$

where $t = \|w_1 - w_2\|$ and $v = (w_1 - w_2)/\|w_1 - w_2\|$. Then

$$\begin{aligned}
&\inf_{\|v\|=1} \inf_{t\geq 0} \left(h(w_2) + t\nabla_w h(w_2)^T v + \frac{\beta t^2}{2} + \eta t \right) \\
=&\inf_{t\geq 0} \left(h(w_2) - t\|\nabla_w h(w_2)\| + \frac{\beta t^2}{2} + \eta t \right) \\
=&h(w_2) + \inf_{t\geq 0} \left(-t(\|\nabla_w h(w_2)\| - \eta) + \frac{\beta t^2}{2} \right).
\end{aligned}$$

If $\|\nabla_w h(w_2)\| - \eta \leq 0$, the quaduatic function is optimized when $t = 0$. Then

$$\begin{aligned}
&h(w_2) + \inf_{t\geq 0} \left(-t(\|\nabla_w h(w_2)\| - \eta) + \frac{\beta t^2}{2} \right) \\
=&h(w_2).
\end{aligned}$$

If $\|\nabla_w h(w_2)\| - \eta \geq 0$, then

$$\begin{aligned}
&h(w_2) + \inf_{t\geq 0} \left(-t(\|\nabla_w h(w_2)\| - \eta) + \frac{\beta t^2}{2} \right) \\
=&h(w_2) - \frac{1}{2\beta}[\|\nabla_w h(w_2)\| - \eta]^2.
\end{aligned}$$

Therefore, we obtain that

$$h(w^*) - h(w) \leq -\frac{1}{2\beta}\left[[\|\nabla h(w)\| - \eta]_+ \right]^2. \tag{3.8.1}$$

Define

$$h_1(w) = h(w) - \nabla h(w_1)^T w$$

and

$$h_2(w) = h(w) - \nabla h(w_2)^T w.$$

Firstly, it is easy to see that $h_1(w)$ and $h_2(w)$ are both η-approximate β-gradient Lipschitz, which satisfies inequatily in Eq. (3.8.1). Secondly, $w = w_1$ minimizes $h_1(w)$.

Then

$$\begin{aligned} & h(w_2) - h(w_1) - \nabla h(w_1)^T (w_2 - w_1) \\ =& h_1(w_2) - h_1(w_1) \\ \geq& \frac{1}{2\beta} \Big[[\|\nabla h_1(w_2)\| - \eta]_+ \Big]^2 \\ =& \frac{1}{2\beta} \Big[[\|\nabla h(w_1) - \nabla h(w_2)\| - \eta]_+ \Big]^2. \end{aligned} \tag{3.8.2}$$

Similarly, we have

$$h(w_1) - h(w_2) - \nabla h(w_2)^T (w_1 - w_2) \geq \frac{1}{2\beta} \Big[[\|\nabla h(w_1) - \nabla h(w_2)\| - \eta]_+ \Big]^2. \tag{3.8.3}$$

Take the summation of Eq. (3.8.2) and Eq. (3.8.3), we have

$$\langle \nabla h(w_1) - \nabla h(w_2), w_1 - w_2 \rangle \geq \frac{1}{\beta} \Big[[\|\nabla h(w_1) - \nabla h(w_2)\| - \eta]_+ \Big]^2.$$

□

Proof of Lemma 3.3

Proof of Lemma 3.3.1 ($\alpha\eta$-approximately $(1 + \alpha\beta)$-expansive).

$$\begin{aligned} & \|G_{\alpha,z}(w_1) - G_{\alpha,z}(w_2)\| \\ =& \|w_1 - w_2 - \alpha(\nabla h(w_1) - \nabla h(w_2))\| \\ \leq& \|w_1 - w_2\| + \|\alpha(\nabla h(w_1) - \nabla h(w_2))\| \\ \leq& \|w_1 - w_2\| + \alpha(\beta\|w_1 - w_2\| + \eta) \\ \leq& (1 + \alpha\beta)\|w_1 - w_2\| + \alpha\eta. \end{aligned}$$

□

Proof of Lemma 3.3.2 ($\alpha\eta$-approximately non-expansive.) Let t $= \|\nabla h(w_1) - \nabla h(w_2)\|$. If $t \leq \eta$, we have

$$\begin{aligned}
&\|G_{\alpha,z}(w_1) - G_{\alpha,z}(w_2)\| \\
=&\|w_1 - w_2 - \alpha(\nabla h(w_1) - \nabla h(w_2))\| \\
\leq&\|w_1 - w_2\| + \alpha\|\nabla h(w_1) - \nabla h(w_2)\| \\
\leq&\|w_1 - w_2\| + \alpha\eta.
\end{aligned}$$

If $t \geq \eta$, we have

$$\begin{aligned}
&\|G_{\alpha,z}(w_1) - G_{\alpha,z}(w_2)\|^2 \\
=&\|w_1 - w_2 - \alpha(\nabla h(w_1) - \nabla h(w_2))\|^2 \\
=&\|w_1 - w_2\|^2 - 2\alpha(\nabla h(w_1) - \nabla h(w_2)^T(w_1 - w_2) + \alpha^2 t^2 \\
\leq&\|w_1 - w_2\|^2 - \frac{2\alpha}{\beta}(t-\eta)^2 + \alpha^2 t^2 \\
=&\|w_1 - w_2\|^2 - \frac{2\alpha}{\beta}(t-\eta)^2 - \frac{2\alpha\eta}{\beta}(t-\eta) + \alpha^2 t^2 + \frac{2\alpha\eta}{\beta}(t-\eta) \\
=&\|w_1 - w_2\|^2 - \frac{2\alpha t}{\beta}(t-\eta) + \alpha^2 t^2 + \frac{2\alpha\eta}{\beta}(t-\eta).
\end{aligned}$$

Let $\alpha \leq 1/\beta$, then

$$\begin{aligned}
&\|w_1 - w_2\|^2 - \frac{2\alpha t}{\beta}(t-\eta) + \alpha^2 t^2 + \frac{2\alpha\eta}{\beta}(t-\eta) \\
\leq&\|w_1 - w_2\|^2 - 2\alpha^2 t(t-\eta) + \alpha^2 t^2 + \frac{2\alpha\eta}{\beta}(t-\eta) \\
\leq&\|w_1 - w_2\|^2 - \alpha^2(t+\eta)(t-\eta) + \alpha^2 t^2 + \frac{2\alpha\eta}{\beta}(t-\eta) \\
\leq&\|w_1 - w_2\|^2 + \alpha^2\eta^2 + \frac{2\alpha\eta}{\beta}(t-\eta).
\end{aligned}$$

By the definition of η-approximate smoothness,

$$\frac{1}{\beta}(t-\eta) \leq \|w_1 - w_2\|.$$

Then

$$\begin{aligned}&\|w_1 - w_2\|^2 + \alpha^2\eta^2 + \frac{2\alpha\eta}{\beta}(t-\eta)\\ \leq&\|w_1 - w_2\|^2 + \alpha^2\eta^2 + 2\alpha\eta\|w_1 - w_2\|\\ =&(\|w_1 - w_2\| + \alpha\eta)^2.\end{aligned}$$

Therefore, we obtain that

$$\|G_{\alpha,z}(w_1) - G_{\alpha,z}(w_2)\| \leq \|w_1 - w_2\| + \alpha\eta.$$

□

Proof of Lemma 3.3.3 ($\alpha\eta$-approximately $(1-\alpha\gamma)$-contraction.).

Firstly, if $h(w)$ is a γ-strongly convex, η-approximately β-gradient Lipschitz function, $\phi(w) = h(w) - \frac{\gamma}{2}\|w\|^2$ is a convex, η-approximate $(\beta-\gamma)$-gradient Lipschitz function. The proof of convexity follows the definition. To see the second claim, since

$$\begin{aligned}&\phi(w_1) - \phi(w_2)\\ =&h(w_1) - h(w_2) - (\frac{\gamma}{2}\|w_1\|^2 - \frac{\gamma}{2}\|w_2\|^2)\\ \leq&\langle\nabla_w h(w_2), w_1 - w_2\rangle + \frac{\beta}{2}\|w_1 - w_2\|^2 + \eta\|w_1 - w_2\| - (\frac{\gamma}{2}\|w_1\|^2 - \frac{\gamma}{2}\|w_2\|^2)\\ \leq&\langle\nabla_w \phi(w_2), w_1 - w_2\rangle + \frac{\beta}{2}\|w_1 - w_2\|^2 + \eta\|w_1 - w_2\| - (\frac{\gamma}{2}\|w_1\|^2 - \frac{\gamma}{2}\|w_2\|^2) + \gamma w_2^T(w_1 - w_2)\\ \leq&\langle\nabla_w \phi(w_2), w_1 - w_2\rangle + \frac{\beta}{2}\|w_1 - w_2\|^2 + \eta\|w_1 - w_2\| - \frac{\gamma}{2}\|w_1 - w_2\|^2\\ \leq&\langle\nabla_w \phi(w_2), w_1 - w_2\rangle + \frac{\beta-\gamma}{2}\|w_1 - w_2\|^2 + \eta\|w_1 - w_2\|.\end{aligned}$$

Therefore, $\phi(w)$ satisfies the η-approximate $(\beta-\gamma)$-descent Lemma. Let $t = \|\nabla\phi(w_1) - \nabla\phi(w_2)\|$.

If $t \leq \eta$, we have

$$\begin{aligned}&\|G_{\alpha,z}(w_1) - G_{\alpha,z}(w_2)\|\\ =&\|w_1 - w_2 - \alpha(\nabla h(w_1) - \nabla h(w_2))\|\\ =&\|w_1 - w_2 - \alpha(\nabla\phi(w_1) + \gamma w_1 - \nabla\phi(w_2) - \gamma w_2)\|\\ \leq&\|(1-\alpha\gamma)(w_1 - w_2)\| + \alpha\|\nabla\phi(w_1) - \nabla\phi(w_2)\|\\ \leq&(1-\alpha\gamma)\|w_1 - w_2\| + \alpha\eta.\end{aligned}$$

If $t \geq \eta$, we have

$$\begin{aligned}
&\|G_{\alpha,z}(w_1) - G_{\alpha,z}(w_2)\|^2 \\
=&\|w_1 - w_2 - \alpha(\nabla h(w_1) - \nabla h(w_2))\|^2 \\
=&\|w_1 - w_2 - \alpha(\nabla \phi(w_1) + \gamma w_1 - \nabla \phi(w_2) - \gamma w_2)\|^2 \\
\leq&(1-\alpha\gamma)^2\|w_1 - w_2\|^2 - 2\alpha(1-\alpha\gamma)(\nabla\phi(w_1) - \nabla\phi(w_2)^T(w_1 - w_2) + \alpha^2 t^2 \\
\leq&(1-\alpha\gamma)^2\|w_1 - w_2\|^2 - \frac{2\alpha(1-\alpha\gamma)}{\beta-\gamma}(t-\eta)^2 + \alpha^2 t^2 \\
\leq&(1-\alpha\gamma)^2\|w_1 - w_2\|^2 - \frac{2\alpha(1-\alpha\gamma)}{\beta-\gamma}(t-\eta)^2 - \frac{2\alpha(1-\alpha\gamma)\eta}{\beta-\gamma}(t-\eta) + \alpha^2 t^2 + \frac{2\alpha(1-\alpha\gamma)\eta}{\beta-\gamma}(t-\eta).
\end{aligned}$$

Since $\alpha \leq 1/\beta$, we have $(1-\alpha\gamma)/(\beta-\gamma) \geq \alpha$, then

$$\begin{aligned}
&(1-\alpha\gamma)^2\|w_1 - w_2\|^2 - \frac{2\alpha(1-\alpha\gamma)}{\beta-\gamma}(t-\eta)^2 - \frac{2\alpha(1-\alpha\gamma)\eta}{\beta-\gamma}(t-\eta) + \alpha^2 t^2 + \frac{2\alpha(1-\alpha\gamma)\eta}{\beta-\gamma}(t-\eta) \\
\leq&(1-\alpha\gamma)^2\|w_1 - w_2\|^2 - \alpha^2 t(t-\eta) + \alpha^2 t^2 + \frac{2\alpha(1-\alpha\gamma)\eta}{\beta-\gamma}(t-\eta) \\
\leq&(1-\alpha\gamma)^2\|w_1 - w_2\|^2 + \alpha^2\eta^2 + 2\alpha(1-\alpha\gamma)\eta\|w_1 - w_2\| \\
\leq&\Big((1-\alpha\gamma)\|w_1 - w_2\| + \alpha\eta\Big)^2.
\end{aligned}$$

Therefore, we obtain that

$$\|G_{\alpha,z}(w_1) - G_{\alpha,z}(w_2)\| \leq (1-\alpha\gamma)\|w_1 - w_2\| + \alpha\eta.$$

□

Proof of Theorem 3.2

The proof follows the standard techniques for uniform stability. We need to replace the non-expansive property used in standard analysis by the approximately non-expansive property. Let S and S' be two samples of size n differing in only a single example. Consider two trajectories $w_1^1, \ldots, w_1^T$ and $w_2^1, \ldots, w_2^T$ induced by running SGD on sample S and S', respectively. Let $\delta_t = \|w_1^t - w_2^t\|$.

Fixing an example $z \in Z$ and apply the Lipschitz condition on $h(\cdot\,; z)$, we have

$$\mathbb{E}\left|h(w_1^T; z) - h(w_2^T; z)\right| \leq L\,\mathbb{E}\left[\delta_T\right]. \tag{3.8.4}$$

Observe that at step t, with probability $1 - 1/n$, the example selected by SGD is

the same in both S and S'. With probability $1/n$ the selected example is different. Therefore, by the $\alpha\eta$-approximate non-expansive property, we have

$$\mathbb{E}\left[\delta_{t+1}\right] \leq \left(1-\frac{1}{n}\right)\left(\mathbb{E}\left[\delta_t\right]+\alpha_t\eta\right)+\frac{1}{n}\mathbb{E}\left[\delta_t\right]+\frac{2\alpha_t L}{n} \leq \mathbb{E}\left[\delta_t\right]+\left(\eta+\frac{2L}{n}\right)\alpha_t\,. \tag{3.8.5}$$

Unraveling the recursion gives

$$\mathbb{E}\left[\delta_T\right] \leq \left(\eta+\frac{2L}{n}\right)\sum_{t=1}^{T}\alpha_t\,.$$

Plugging this back into Eq. (3.8.4), we obtain

$$\mathcal{E}_{gen} \leq L\left(\eta+\frac{2L}{n}\right)\sum_{t=1}^{T}\alpha_t\,.$$

Since this bounds holds for all S, S' and z, we obtain the desired bound on the uniform stability. □

Proof of Theorem 3.3

Proof: The construction of function h is adopted from the construction in the work of (Bassily et al., 2020).

Let $T \leq d$, and $v, K \geq 0$. Considering $\mathcal{Z} = \{0, 1\}$, and the objective function

$$h(w,z) = \begin{cases} \eta \max\{0, x_1 - v, \cdots, x_T - v\} & \text{if } z = 0 \\ \langle r, x\rangle / K & \text{if } z = 1, \end{cases} \tag{3.8.6}$$

where $r = (-1, \cdots, -1, 0, \cdots, 0)$ (i.e., equals to -1 for the first T components). Function h is η-approximately smooth since the first case is a piece-wise linear function. For the

dataset S and S' differ in at most one sample, the empirical objective functions are

$$R_S(w) = \frac{1}{nK}\langle r, x\rangle + \frac{n-1}{n}\eta \max\{0, x_1 - v, \cdots, x_T - v\},$$

and

$$R_{S'}(w) = \eta \max\{0, x_1 - v, \cdots, x_T - v\}.$$

Let w_1^T and w_2^T be the trajectories running algorithm on dataset S and S', initialized on $w_1^0 = w_2^0 = 0$. Clearly, $w_2^t = 0$ for all t. It is easy to obtain $w_1^t = -\frac{t\alpha r}{nk} - \alpha\eta\frac{n-1}{n}\sum_{s=1}^{t-1} e_s$ recursively. By the orthogonality of the subgradients, we have

$$\delta(S, S') = \|w_1^T - w_2^T\| = \|w_1^T\| \geq \Omega\left(\alpha\eta\left\|\sum_{s=1}^{T} e_s\right\|\right) = \Omega(\alpha\eta\sqrt{T}). \tag{3.8.7}$$

On the other hand, the work of (Hardt et al., 2016) provided a lower bound for general non-smooth function

$$\delta(S, S') \geq \Omega\left(\frac{L\alpha T}{n}\right). \tag{3.8.8}$$

Combining Eq. (3.8.7) and Eq. (3.8.8), we have

$$\delta(S, S') \geq \Omega\left(\alpha\eta\sqrt{T} + \frac{L\alpha T}{n}\right). \tag{3.8.9}$$

□

Proof of Theorem 3.4

We consider a general form of Theorem 3.4.

Theorem 3.8 (Non-convex)**.** *Assume that $h(w, z)$ is L-Lipschitz, and η-approximately β-gradient Lipschitz in w for all given $z \in \mathcal{Z}$. Assume in addition that $0 \leq g(w, z) \leq B$ for all w and z. Suppose that we run SGD on the adversarial surrogate loss with step sizes $\alpha_t \leq c/t$ for T steps, where $c > 0$. Then, for all $t_0 \in \{1, 2, \cdots, n\}$, adversarial*

training satisfies uniform stability with

$$\mathcal{E}_{gen} = \mathbb{E}[R_{\mathcal{D}}(w^T) - R_S(w^T)] \leq \frac{Bt_0}{n-1} + \frac{1}{\beta(n-1)}\left(2L^2 + L\eta n\right)\left(\frac{T}{t_0}\right)^{\beta c}. \quad (3.8.10)$$

Let $q = \beta c$. For small T (s.t. $t_0 \leq n$ when we set the first term equals to the second term). we select t_0 to optimize the right hand side, then

$$\mathcal{E}_{gen} \leq \frac{2}{n-1} B^{\frac{q}{q+1}} \left(\frac{2L^2 + L\eta n}{\beta}\right)^{\frac{1}{q+1}} T^{\frac{q}{q+1}}. \quad (3.8.11)$$

For arbitrary T, the optimal $t_0 > n$. We simply let $t_0 = 1$, then

$$\mathcal{E}_{gen} \leq \frac{BL_w + (2L^2 + L\eta n)T^q}{\beta(n-1)}. \quad (3.8.12)$$

Proof: Let S and S' be two samples of size n differing in only a single example. Consider two trajectories $w_1^1, \ldots, w_1^T$ and $w_2^1, \ldots, w_2^T$ induced by running SGD on sample S and S', respectively. Let $\delta_t = \|w_1^t - w_2^t\|$. Let $t_0 \in \{0, 1, \ldots, n\}$, be the iteration that $\delta_{t_0} = 0$, but SGD picks two different samples form S and S' in iteration $t_0 + 1$, then

$$\mathcal{E}_{gen} \leq \frac{t_0}{n} B + L\,\mathbb{E}\left[\delta_T \mid \delta_{t_0} = 0\right]. \quad (3.8.13)$$

Let $\Delta_t = \mathbb{E}\left[\delta_t \mid \delta_{t_0} = 0\right]$. Observe that at step t, with probability $1 - 1/n$, the example selected by SGD is the same in both S and S'. With probability $1/n$ the selected example is different. Therefore, by the $\alpha\eta$-approximate $(1 + \alpha\beta)$-expansive property, for every $t \geq t_0$,

$$\begin{aligned}
\Delta_{t+1} &\leq \left(1 - \frac{1}{n}\right)(1 + \alpha_t\beta)\Delta_t + \frac{1}{n}\Delta_t + \left(\eta + \frac{2L}{n}\right)\alpha_t \\
&\leq \left(\frac{1}{n} + (1 - 1/n)(1 + c\beta/t)\right)\Delta_t + \left(\eta + \frac{2L}{n}\right)\frac{c}{t} \\
&= \left(1 + (1 - 1/n)\frac{c\beta}{t}\right)\Delta_t + \left(\eta + \frac{2L}{n}\right)\frac{c}{t} \\
&\leq \exp\left((1 - 1/n)\frac{c\beta}{t}\right)\Delta_t + \left(\eta + \frac{2L}{n}\right)\frac{c}{t}.
\end{aligned}$$

Here we used the fact that $1 + x \leq \exp(x)$ for all x.

Using the fact that $\Delta_{t_0} = 0$, we can unwind this recurrence relation from T down to $t_0 + 1$. This gives

$$\begin{aligned}
\Delta_T &\leq \sum_{t=t_0+1}^{T} \left\{ \prod_{k=t+1}^{T} \exp\left((1-\tfrac{1}{n})\tfrac{\beta c}{k}\right) \right\} \left(\eta + \frac{2L}{n}\right)\frac{c}{t} \\
&= \sum_{t=t_0+1}^{T} \exp\left((1-\tfrac{1}{n})\beta c \sum_{k=t+1}^{T} \tfrac{1}{k}\right) \left(\eta + \frac{2L}{n}\right)\frac{c}{t} \\
&\leq \sum_{t=t_0+1}^{T} \exp\left((1-\tfrac{1}{n})\beta c \log(\tfrac{T}{t})\right) \left(\eta + \frac{2L}{n}\right)\frac{c}{t} \\
&= \left(\eta + \frac{2L}{n}\right) c T^{\beta c(1-1/n)} \sum_{t=t_0+1}^{T} t^{-\beta c(1-1/n)-1} \\
&\leq \left(\eta + \frac{2L}{n}\right) \frac{1}{(1-1/n)\beta c} c \left(\frac{T}{t_0}\right)^{\beta c(1-1/n)} \\
&\leq \frac{\eta n + 2L}{\beta(n-1)} \left(\frac{T}{t_0}\right)^{\beta c},
\end{aligned}$$

Plugging this bound into (3.8.13), we get

$$\mathcal{E}_{gen} \leq \frac{B t_0}{n-1} + \frac{L\eta n + 2L^2}{\beta(n-1)} \left(\frac{T}{t_0}\right)^{\beta c}.$$

Let $q = \beta c$. For small T (s.t. $t_0 \leq n$ when we set the first term equals to the second term). we select t_0 to optimize the right hand side, then

$$\mathcal{E}_{gen} \leq \frac{2}{n-1} B^{\frac{q}{q+1}} \left(\frac{2L^2 + L\eta n}{\beta}\right)^{\frac{1}{q+1}} T^{\frac{q}{q+1}}.$$

For arbitrary T, the optimal $t_0 > n$. We simply let $t_0 = 1$, then

$$\mathcal{E}_{gen} \leq \frac{B L_w + (2L^2 + L\eta n) T^q}{\beta(n-1)}.$$

Since the bound we just derived holds for all S, S' and z, we immediately get the claimed upper bound on the uniform stability. Let $q = 1$, we obtain the result of

Theorem 3.4. □

Proof of Theorem 3.5

The proof follows the idea in convex case. By the $\alpha\eta$-approximately $(1-\alpha\gamma)$-contraction, for every t,

$$\begin{aligned}\mathbb{E}\,\delta_{t+1} &\le \left(1-\frac{1}{n}\right)(1-\alpha\gamma)\,\mathbb{E}\,\delta_t + \frac{1}{n}(1-\alpha\gamma)\,\mathbb{E}\,\delta_t + \left(\eta+\frac{2L}{n}\right)\alpha \\ &= (1-\alpha\gamma)\,\mathbb{E}\,\delta_t + \left(\eta+\frac{2L}{n}\right)\alpha\,.\end{aligned} \tag{3.8.14}$$

Unraveling the recursion gives

$$\mathbb{E}\,\delta_T \le \left(\eta+\frac{2L}{n}\right)\alpha\sum_{t=0}^{T}(1-\alpha\gamma)^t \le \frac{\eta}{\gamma}+\frac{2L}{\gamma n}\,.$$

Plugging the above inequality into Eq. (3.8.4), we obtain

$$\mathcal{E}_{gen} \le \frac{L\eta}{\gamma}+\frac{2L^2}{\gamma n}\,.$$

Since this bounds holds for all S, S' and z, the Theorem follows. □

Proof of Theorem 3.7

Let $\bar{w} = \frac{1}{T}\sum_{t=1}^{T} w^t$ denote the average of the stochastic gradient iterates. Since

$$w^t = \sum_{k=1}^{t}\alpha_k \nabla h(w^k; z_k)\,,$$

we have

$$\bar{w} = \sum_{t=1}^{T}\alpha_t\frac{T-t+1}{T}\nabla h(w^k; z_k)$$

Using the $\alpha\eta$-approximate non-expansive, we have

$$\delta_t \le (1-1/n)\delta_{t-1} + \frac{1}{n}\left(\delta_{t-1} + (\eta n + 2L)\alpha_t\frac{T-t+1}{T}\right)\,.$$

which implies

$$\delta_T \leq \left(\eta + \frac{2L}{n}\right) \sum_{t=1}^{T} \alpha_t \frac{T-t+1}{T} = \left(\frac{\eta}{2} + \frac{L}{n}\right) \sum_{t=1}^{T} \alpha_t \,.$$

Since f is L-Lipschitz, we have

$$\mathcal{E}_{gen}(\bar{w}) \leq \left(\frac{L\eta}{2} + \frac{L^2}{n}\right) \sum_{t=1}^{T} \alpha_t. \tag{3.8.15}$$

Here the expectation is taken over the algorithm and hence the claim follows by our definition of uniform stability. $\mathcal{E}_{opt}$ follows (Nemirovski et al., 2009). □

3.9 Discussion on Non-convex and Strongly Convex Case

Discussion on Non-convex Case

To discuss the generalization-optimization trade-off in the non-convex case. We first give the optimization error bound.

Theorem 3.9. *Assume that h is η-approximate β-gradient Lipschitz and given $0 < \tau < 1$. Without loss of generality, assume the stochastic gradient $\nabla\tilde{h}(w)$ be unbiased and have a bounded variance σ^2. Let the stochastic gradient descent (SGD) update be $w_{t+1} = w_t - \alpha\nabla\tilde{h}(w_t)$ with a constant step size $\alpha = 1/\sqrt{T}$ for number of iterations $T \geq (\beta/2(1-\tau))^2$. $\exists t \leq T$, s.t.*

$$\mathbb{E}\|\nabla h(w_t)\|^2 \leq \frac{\eta^2}{\tau^2} + \frac{2\eta\sigma}{\tau} + \mathcal{O}(\frac{1}{\sqrt{T}}). \tag{3.9.1}$$

Proof:

Assume that the stochastic gradient $\nabla h(w)$ be unbiased and have a bounded variance σ^2.

$$\mathbb{E}[\nabla\tilde{h}(w)] = \nabla h(w),$$

$$\mathbb{E}\|\nabla\tilde{h}(w)\|^2 \leq \|\nabla h(w)\|^2 + \sigma^2.$$

Notice that when w is a random vector, the above expectation is condition on w. Let the stochastic gradient descent (SGD) update be $w_{t+1} = w_t - \alpha\tilde{h}(w_t)$ with a constant step size $\alpha = 1/\sqrt{T}$. By η-approximately Descent Lemma, we have

$$\begin{aligned}
&h(w_{t+1}) - h(w_t) \\
=& -\alpha\langle\nabla h(w_t), w_{t+1} - w_t\rangle + \frac{\beta}{2}\|w_{t+1} - w_t\|^2 + \eta\|w_{t+1} - w_t\| \\
=& -\alpha\langle\nabla h(w_t), \nabla\tilde{h}(w_t)\rangle + \frac{\beta\alpha^2}{2}\|\nabla\tilde{h}(w_t)\|^2 + \eta\alpha\|\nabla\tilde{h}(w_t)\|.
\end{aligned}$$

Given w_t, take the conditional expectation over the noised introduced by SGD, we have

$$\begin{aligned}
&\mathbb{E}[h(w_{t+1})] - h(w_t) \\
=& -\alpha\langle\nabla h(w_t), \mathbb{E}[\nabla\tilde{h}(w_t)]\rangle + \frac{\beta\alpha^2}{2}\mathbb{E}\|\nabla\tilde{h}(w_t)\|^2 + \eta\alpha\mathbb{E}\|\nabla\tilde{h}(w_t)\| \\
\leq& -\alpha\|\nabla h(w_t)\|^2 + \frac{\beta\alpha^2}{2}\big[\|\nabla h(w_t)\|^2 + \sigma^2\big] + \eta\alpha\sqrt{\big[\mathbb{E}\|\nabla\tilde{h}(w_t)\|\big]^2} \\
\leq& -\alpha\|\nabla h(w_t)\|^2 + \frac{\beta\alpha^2}{2}\big[\|\nabla h(w_t)\|^2 + \sigma^2\big] + \eta\alpha\sqrt{\|\nabla h(w_t)\|^2 + \sigma^2} \\
\leq& -\alpha\|\nabla h(w_t)\|^2 + \frac{\beta\alpha^2}{2}\big[\|\nabla h(w_t)\|^2 + \sigma^2\big] + \eta\alpha\big[\|\nabla h(w_t)\| + \sigma\big] \\
=& -\alpha\|\nabla h(w_t)\|^2 + \frac{\beta\alpha^2}{2}\|\nabla h(w_t)\|^2 + \eta\alpha\|\nabla_w h(w_t)\| + \frac{\beta\alpha^2\sigma^2}{2} + \eta\alpha\sigma \\
\leq& -\tau\alpha\|\nabla h(w_t)\|^2 + \eta\alpha\|\nabla h(w_t)\| + \frac{\beta\alpha^2\sigma^2}{2} + \eta\alpha\sigma,
\end{aligned}$$

where the first inequality is the assumption of SGD, the second inequality is the Jensen's inequality, the third one is the assumption of SGD, the fourth one is the Cauchy-Schwartz inequality, and the last one is because of the size of step size α. Take the expectation over the trajectory $w_0, w_1, \cdots, w_T$, and take the average aver $t = 0, 1, \cdots, T$, we have

$$\begin{aligned}
&\frac{1}{T}\sum_{t=0}^{T}\big[\tau\alpha\mathbb{E}\|\nabla h(w_t)\|^2 - \eta\alpha\mathbb{E}\|\nabla h(w_t)\|\big] \\
\leq& \frac{1}{T}\sum_{t=0}^{T}\big[\mathbb{E}[h(w_t)] - \mathbb{E}h(w_{t+1})\big] + \frac{\beta\alpha^2\sigma^2}{2} + \eta\alpha\sigma \\
\leq& \frac{1}{T}\big[\mathbb{E}[h(w_0)] - h(w_*)\big] + \frac{\eta\alpha^2\sigma^2}{2} + \eta\alpha\sigma.
\end{aligned}$$

Let $\mathbb{E}[h(w_0)] - h(w_*) = D$ and divide α on both side. $\exists t \leq T$, *s.t.*

$$\tau\mathbb{E}\|\nabla h(w_t)\|^2 - \eta\mathbb{E}\|\nabla h(w_t)\| \leq \frac{D}{T\alpha} + \frac{\beta\alpha\sigma^2}{2} + \eta\sigma.$$

Since $\alpha = 1/\sqrt{T}$, we have

$$\begin{aligned}
&\tau\mathbb{E}\|\nabla h(w_t)\|^2 - \eta\mathbb{E}\|\nabla h(w_t)\| \leq \frac{D}{\sqrt{T}} + \frac{\beta\sigma^2}{2\sqrt{T}} + \eta\sigma \\
\Leftrightarrow &\mathbb{E}\|\nabla h(w_t)\|^2 - \frac{\eta}{\tau}\mathbb{E}\|\nabla h(w_t)\| + (\frac{\eta}{2\tau})^2 \leq \frac{1}{\sqrt{T}}\left[\frac{D}{\tau} + \frac{\beta\sigma^2}{2\tau}\right] + \frac{\eta\sigma}{\tau} + (\frac{\eta}{2\tau})^2 \\
\Leftrightarrow &\left|\mathbb{E}\|\nabla h(w_t)\| - \frac{\eta}{2\tau}\right| \leq \sqrt{\frac{1}{\sqrt{T}}\left[\frac{D}{\tau} + \frac{\beta\sigma^2}{2\tau}\right] + \frac{\eta\sigma}{\tau} + (\frac{\eta}{2\tau})^2}.
\end{aligned}$$

Then we remove the absolute value, and obtain

$$\begin{aligned}
&\mathbb{E}\|\nabla h(w_t)\| - \frac{\eta}{2\tau} \leq \sqrt{\frac{1}{\sqrt{T}}\left[\frac{D}{\tau} + \frac{\beta\sigma^2}{2\tau}\right] + \frac{\eta\sigma}{\tau} + (\frac{\eta}{2\tau})^2} \\
\Leftrightarrow &\mathbb{E}\|\nabla h(w_t)\| \leq \frac{\eta}{2\tau} + \sqrt{\frac{1}{\sqrt{T}}\left[\frac{D}{\tau} + \frac{\beta\sigma^2}{2\tau}\right] + \frac{\eta\sigma}{\tau} + (\frac{\eta}{2\tau})^2} \\
\Leftrightarrow &\mathbb{E}\|\nabla h(w_t)\|^2 \leq \left[\frac{\eta}{2\tau} + \sqrt{\frac{1}{\sqrt{T}}\left[\frac{D}{\tau} + \frac{\beta\sigma^2}{2\tau}\right] + \frac{\beta\sigma}{\tau} + (\frac{\eta}{2\tau})^2}\right]^2.
\end{aligned}$$

By Cauchy-Schwartz inequality, we have

$$\begin{aligned}
\mathbb{E}\|\nabla h(w_t)\|^2 \leq &\left[\frac{\eta}{2\tau} + \sqrt{\frac{1}{\sqrt{T}}\left[\frac{D}{\tau} + \frac{\beta\sigma^2}{2\tau}\right] + \frac{\eta\sigma}{\tau} + (\frac{\eta}{2\tau})^2}\right]^2 \\
\leq &2\left[(\frac{\eta}{2\tau})^2 + \frac{1}{\sqrt{T}}\left[\frac{D}{\tau} + \frac{\beta\sigma^2}{2\tau}\right] + \frac{\beta\sigma}{\tau} + (\frac{\eta}{2\tau})^2\right] \\
= &\frac{\eta^2}{\tau^2} + \frac{2\eta\sigma}{\tau} + \frac{2}{\sqrt{T}}\left[\frac{D}{\tau} + \frac{\beta\sigma^2}{2\tau}\right] \\
= &\frac{\eta^2}{\tau^2} + \frac{2\eta\sigma}{\tau} + \mathcal{O}(\frac{1}{\sqrt{T}}).
\end{aligned}$$

□

In words, running SGD on an approximately smooth non-convex function, the algorithm cannot converge to a stationary point but with an additional constant term. Notice

that this is an error bound for gradient norm. If we need an error bound for the optimality gap, we need an additional PL condition. Combining the optimization error and generalization error, we have

$$\mathcal{E}_{opt} + \mathcal{E}_{gen} \leq \mathcal{O}\left(\eta T + \frac{T}{n} + \frac{1}{\sqrt{T}}\right) + \text{constant},$$

where the first term is an additional term for adversarial training, which induces robust overfitting. Therefore, we can see that the analysis of the convex and non-convex cases do not have a major difference.

Discussion on Strongly Convex Case

By (Nemirovski et al., 2009),

$$\mathcal{E}_{opt} \leq \frac{LD^2}{T} = \mathcal{O}(\frac{1}{T})$$

in the strongly convex case. Whether the function is smooth does not affect the convergence rate. Therefore,

$$\mathcal{E}_{opt} + \mathcal{E}_{gen} \leq \mathcal{O}\left(\eta + \frac{1}{n} + \frac{1}{T}\right).$$

This result shows that robust overfitting will disappear if the loss function is strongly convex. But the performance of adversarial training is still worse than the performance of standard training in $\mathcal{O}(\eta)$ in this strong assumption.

Discussion on Strongly Concave Assumption on the Inner Problem

In this subsection, we discuss the case that $g(w, z)$ is μ-strongly concave in z.

Assumption 3.2. *The function g satisfies the following Lipschitzian smoothness con-*

ditions:

$$\|g(w_1, z) - g(w_2, z)\| \leq L\|w_1 - w_2\|,$$
$$\|\nabla_w g(w_1, z) - \nabla_w g(w_2, z)\| \leq L_w\|w_1 - w_2\|,$$
$$\|\nabla_w g(w, z_1) - \nabla_w g(w, z_2)\| \leq L_z\|z_1 - z_2\|,$$
$$\|\nabla_z g(w_1, z) - \nabla_w g(w_2, z)\| \leq L_{zw}\|w_1 - w_2\|.$$

Assumption 3.2 assumes that the loss function is smooth (in zeroth-order and first-order), which are also used in the stability literature (Farnia & Ozdaglar, 2021; Xing et al., 2021b), as well as the convergence analysis literature (Wang et al., 2019; Liu et al., 2020). Comparing with Assumption 3.1, Assumption 3.2 requires one more gradient Lipschitz $\|\nabla_z g(w_1, z) - \nabla_w g(w_2, z)\| \leq L_{zw}\|w_1 - w_2\|$.

Lemma 3.4. *Under Assumption 3.2, assume in addition that $g(w, z)$ is μ-strongly concave in z. $\forall w_1, w_2$ and $\forall z \in \mathcal{Z}$, the following properties hold.*

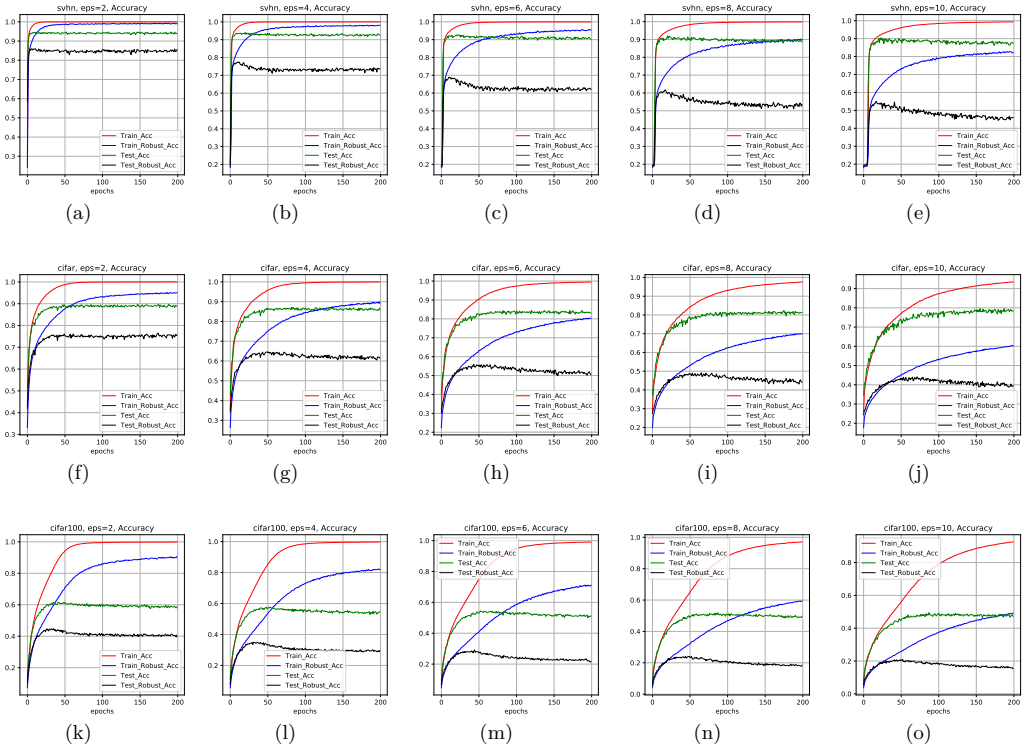
1. *(Lipschitz function.)* $\|h(w_1, z) - h(w_2, z)\| \leq L\|w_1 - w_2\|$.
2. *(gradient Lipschitz.)* $\|\nabla_w h(w_1, z) - \nabla_w h(w_2, z)\| \leq \beta_2\|w_1 - w_2\|$, *where*

$$\beta_2 = \frac{L_z L_{zw}}{\mu} + L_w.$$

The proof can be found in (Sinha et al., 2017; Wang et al., 2019). Therefore, the adversarial surrogate loss is β_2-gradient Lipschitz. The stability generalization bounds follows (Hardt et al., 2016) by replacing β by β_2 (for the choice of step size α).

3.10 Additional Experiments

In this section, we provide additional experiments on SVHN, CIFAR-10, and CIFAR-100. In Fig. 3.4, we show the experiments of adversarial training using a fixed learning rate. In Fig. 3.5, we show the experiments of adversarial training using a standard piece-wise linear learning rate. In Fig. 3.6, we show the experiments of adversarial training using cyclic learning rate.

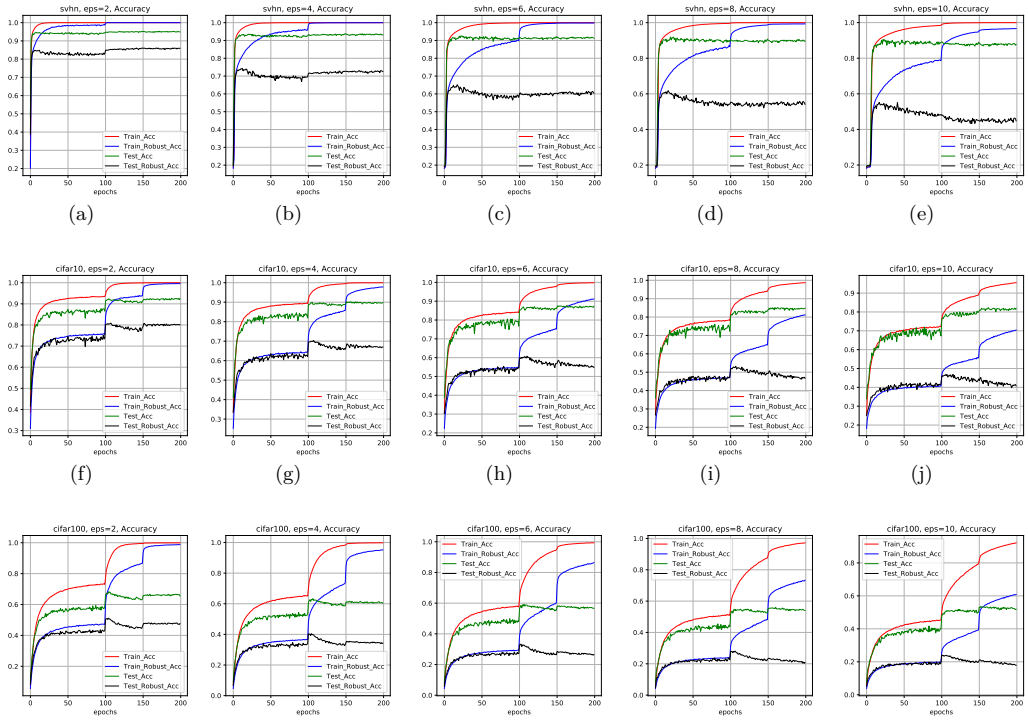


(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

(k) (l) (m) (n) (o)

Figure 3.4: Accuracy of adversarial training with fixed learning rate = 0.01. The first row is the experiments on SVHN. The second row is the experiments on CIFAR-10. The last row is the experiments on CIFAR-100. The first column to the last column are the experiments of ϵ equal to 2, 4, 6, 8, and 10, respectively.



(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

(k) (l) (m) (n) (o)

Figure 3.5: Accuracy of adversarial training with piece-wise linear learning rate. The first row is the experiments on SVHN. The second row is the experiments on CIFAR-10. The last row is the experiments on CIFAR-100. The first column to the last column are the experiments of ϵ equal to 2, 4, 6, 8, and 10, respectively.

Cyclic Learning Rate. We illustrate the experiments of adversarial training using cyclic learning rate. The learning rate increases linearly in the first 80 epochs and decreases to zero in the last 120 epochs. This learning rate mainly contributes to optimization. In terms of generalization, as discussed in theoretical settings, the generalization bound is no larger than that of the previous cases.

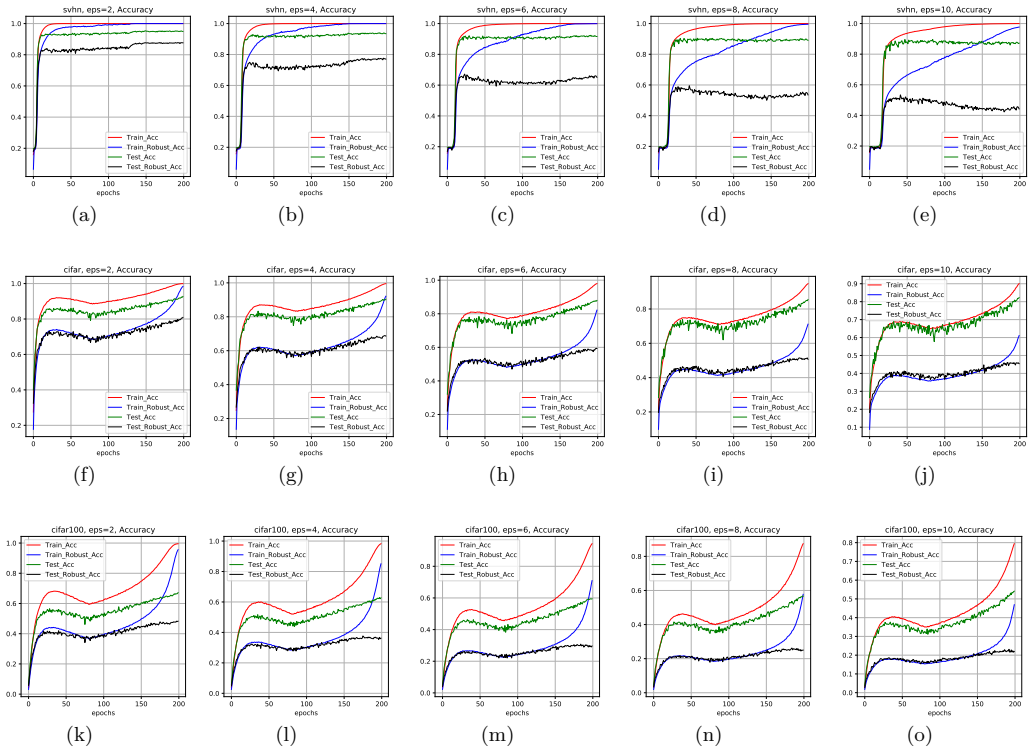


Figure 3.6: Accuracy of adversarial training with super-converge learning rate. The first row is the experiments on SVHN. The second row is the experiments on CIFAR-10. The last row is the experiments on CIFAR-100. The first column to the last column are the experiments of ϵ equal to 2, 4, 6, 8, and 10, respectively.

□ **End of chapter.**

Chapter 4

Eliminating Generalization Error Floor

Summary

In Chapter 3, we show that adversarial loss is non-smooth. Applying SGD on non-smooth loss induces generalization error. A more general question arises: can we eliminate the non-vanishing term? This chapter provides an affirmative answer. We prove that the non-smooth loss minimization problem can achieve a generalization bound in $\mathcal{O}(T\alpha/n)$, which matches the bound in smooth cases. Additionally, our analysis can be extended to weakly-convex cases, while existing uniform stability analysis on non-smooth loss requires a convexity assumption. The main idea is to use tools from Moreau envelopes, and the key technical ingredient to obtain the bound is an error bound on the worst-case change in the optimal solutions of Moreau envelopes. We refer to this variant of SGD as Moreau envelope (ME)-SGD. Combining with the optimization gap, we show that ME-SGD achieves the minimax lower bound. We conduct experiments on three non-smooth losses and show that ME-SGD reduces the generalization gap in practice. In particular, we show the effectiveness of ME-SGD in adversarial training problems.

4.1 Introduction

In Chapter 3, we show that adversarial loss is non-smooth. Applying SGD on non-smooth loss induces generalization error. In this chapter, we consider a more general setting: uniform stability in non-smooth loss minimization problems. Then, we directly apply the results to adversarial settings.

The generalization bound for the non-smooth loss problem is not good enough. Bassily et al. (2020) provided an upper bound on uniform stability in $\mathcal{O}(\alpha\sqrt{T}+\alpha T/n)$ and provided a lower bound to show that the additional term $\mathcal{O}(L\alpha\sqrt{T})$ is unavoidable. The term $\mathcal{O}(L\alpha\sqrt{T})$ is undesirable. It means that we may not have zero generalization error even though we have infinitely many training samples, *i.e.,* $n \to +\infty$. We refer to the non-vanishing term as *generalization error floor*. Generalization error floor might be one of the reasons for the overfitting issue observed in practice. Ever since the work of (Bassily et al., 2020), a few works show that some variants of SGD still induce generalization error floor, *e.g.,* Markov chain-SGD. The comparison of the bounds is listed in Table 4.1. A natural question arises:

Table 4.1: Uniform Stability for different algorithms with normalize Lipschitz in non-smooth convex minimization problem. Here T is the number of iterations, n is the number of samples, and $\alpha > 0$ is the step size.

	Algorithms	Upper Bounds	Lower Bounds	No Error Floor
(Bassily et al., 2020)	GD (full batch)	$\mathcal{O}(\sqrt{T}\alpha + T\alpha/n)$	$\Omega(\sqrt{T}\alpha + T\alpha/n)$	✗
(Bassily et al., 2020)	SGD (w/replacement)	$\mathcal{O}(\sqrt{T}\alpha + T\alpha/n)$	$\Omega(\min\{1, T/n\}\sqrt{T}\alpha + T\alpha/n)$	✗
(Bassily et al., 2020)	SGD (fixed permutation)	$\mathcal{O}(\sqrt{T}\alpha + T\alpha/n)$	$\Omega(\min\{1, T/n\}\sqrt{T}\alpha + T\alpha/n)$	✗
(Yang et al., 2021)	SGD (Pairwise Learning)	$\mathcal{O}(\sqrt{T}\alpha + T\ln T\alpha/n)$	/	✗
(Wang et al., 2022)	Markov chain-SGD	$\mathcal{O}(\sqrt{T}\alpha + T\alpha/n)$	/	✗
Ours	Moreau Envelope-SGD	$\mathcal{O}(T\alpha/n)$	$\Omega(T\alpha/n)$	✓

Can we eliminate the generalization error floor in non-smooth minimization problems?

Observing that the generalization error floor comes from the non-smoothness of the loss function, we employ a smoothing technique using tools from the Moreau envelope function to smooth the loss and perform gradient descent to this smooth surrogate.

We refer to this variant of SGD as Moreau envelops-SGD. We prove that ME-SGD achieves $\mathcal{O}(T\alpha/n)$ uniform stability in a non-smooth loss minimization problem, which improves the bound induced by SGD in $\mathcal{O}(\sqrt{T}\alpha + T\alpha/n)$. Our bound matches the bound in smooth cases (Hardt et al., 2016). Finally, we study the minimax optimality of the algorithm. The minimax lower bound of excess risk on the Lipschiz-convex problem (Nemirovskij & Yudin, 1983). By combining the upper bound of optimization error, we show that the obtained upper bound matches the minimax lower bound.

Technical challenge. Since the Moreau envelopes function itself is in the form of a minimization problem, the whole problem is in a min-min formulation, which is not a standard min-sum formulation in machine learning. Therefore, standard analytical tools (e.g., (Hardt et al., 2016)) cannot be applied. To this end, we develop error bounds on the worst-case change in the optimal solutions of Moreau envelopes functions such that we can obtain the stability bounds.

Weakly-Convex Cases. Notice that all the works we listed in Table 4.1 assume that the loss function is convex. It is unclear how to extend the results to non-convex cases. Our analysis can generalize to weakly-convex cases. We prove that the generalization bound still does not contain the error floor in this case. The analysis is provided in Sec. 4.4.

Additionally, it is widely observed in practice that simply taking the average on the weights of the training trajectory can generalize better than a single output model (Izmailov et al., 2018; Hwang et al., 2021). In ME-SGD, the inner min problem is to train the weights of the model, and the outer min problem is to do weights averaging. Therefore, a by-product of our analysis is that we provide a different theoretical understanding of model averaging in our framework. See more discussion in Sec. 4.3.4.

4.2 Preliminaries: Stability Analysis for Generalization Gap

We first review the definition of population and empirical risk defined in introduction.

Let $\mathcal{D}$ be an unknown distribution in the sample space $\mathcal{Z}$. Our goal is to find a model w with small population risk, defined as:

$$R_{\mathcal{D}}(w) = \mathbb{E}_{z\sim\mathcal{D}}\, h(w, z),$$

where $h(\cdot,\cdot)$ is the loss function which is possibly nonsmooth. Since we cannot get access to the objective $R_{\mathcal{D}}(w)$ directly due to the unknown distribution $\mathcal{D}$, we instead minimize the empirical risk built on a training dataset. Let $S = \{z_1, \ldots, z_n\} \sim \mathcal{D}^n$ be an sample dataset drawn i.i.d. from $\mathcal{D}$. The empirical risk function is defined as:

$$R_S(w) = \frac{1}{n}\sum_{i=1}^{n} h(w, z_i).$$

Let $\bar{w}$ be the optimal solution of $R_S(w)$. Then, for the algorithm output $\hat{w} = A(S)$, we define the expected generalization gap as

$$\mathcal{E}_{gen}(A, h, n, \mathcal{D}) = \mathbb{E}_{S\sim\mathcal{D}^n, A}[R_{\mathcal{D}}(A(S)) - R_S(A(S))]. \tag{4.2.1}$$

We define the the expected optimization gap as

$$\mathcal{E}_{opt}(A, h, n, \mathcal{D}) = \mathbb{E}_{S\sim\mathcal{D}^n, A}[R_S(A(S)) - R_S(\bar{w})]. \tag{4.2.2}$$

We use $\mathcal{E}_{gen}$ and $\mathcal{E}_{opt}$ as short hand notations of the above definition. To bound the generalization gap of a model $\hat{w} = A(S)$ trained by a randomized algorithm A, we employ the following notion of *uniform stability*.

Definition 4.1. *A randomized algorithm A is* ε-uniformly stable *if for all data sets*

$S, S' \in \mathcal{Z}^n$ such that S and S' differ in at most one example, we have

$$\sup_z \mathbb{E}_A \left[h(A(S); z) - h(A(S'); z) \right] \leq \varepsilon \,. \tag{4.2.3}$$

The following theorem shows that expected generalization gap can be attained from uniform stability.

Theorem 4.1 (Generalization in expectation (Hardt et al., 2016))**.** *Let A be ε-uniformly stable. Then, the expected generalization gap satisfies*

$$|\mathcal{E}_{gen}| = |\mathbb{E}_{S,A}[R_{\mathcal{D}}[A(S)] - R_S[A(S)]]| \leq \varepsilon \,.$$

Uniform Argument Stability (UAS). If h is L-Lipschitz, *i.e.*, $|h(w_1; z) - h(w_2; z)| \leq L\|w_1 - w_2\|$, we can use UAS$= \mathbb{E}\|A(S) - A(S)'\|$ to measure the generalization gap.

Hypothesis Class. Following the work of (Bassily et al., 2020), we mainly consider the following function class of convex, non-smooth, and Lipschitz functions in Section 4.3.

$$\mathcal{H} = \{h : W \times \mathcal{Z} \to \mathbb{R} \mid \quad h \text{ is convex, } L\text{-Lipschitz in } w, |W| = D_W\}. \tag{4.2.4}$$

Note that the convexity assumption can be relaxed. In Section 4.4, we also extend the results to weakly-convex, non-smooth cases.

Generalization Error Floor. The uniform argument stability induced by SGD is given in (Bassily et al., 2020). Running SGD (with replacement and fixed permutation) on $h \in \mathcal{H}$, the UAS satisfies

$$\Omega(\sqrt{T}\alpha + T\alpha/n) \leq \mathbb{E}\delta(S, S') \leq \mathcal{O}(L(\sqrt{T}\alpha + T\alpha/n)).$$

4.3 Moreau envelope-SGD: Eliminating Generalization Error Floor

Observing that the generalization error floor comes from the non-smoothness of the loss function, we aim to use a smooth surrogate loss and apply algorithms on this surrogate loss to eliminate the error floor. The designated smooth surrogate loss should satisfy the following principles.

- **Principle 1.** The optimization problem on the smooth surrogate loss has the same global solutions as the original problem.

The designated algorithms should satisfy the following two principles.

- **Principle 2.** The algorithms have the same convergence guarantee as SGD.
- **Principle 3.** The algorithms can eliminate the generalization error floor, achieving $\mathcal{O}(T\alpha/n)$ uniform stability.

Principle 1 is necessary and important. Otherwise, the optimization problem is changed. Principles 2 and 3 are our goals. We aim to find an algorithm that eliminates the generalization error floor without sacrificing the convergence guarantee.

4.3.1 Smooth Surrogate Loss

Inspired by the work of (Zhang & Luo, 2020), we use the Moreau envelope function to smooth the loss. Let

$$K(w,u;z) = h(w;z) + \frac{p}{2}\|w-u\|^2. \tag{4.3.1}$$

We can choose $p > 0$ to insure that $K(w, u; z)$ is strongly convex with respect to w. We define the Moreau envelope function:

$$\begin{aligned} M(u; S) &= \min_{w \in W} K(w, u; S) \\ &= \min_{w \in W} \frac{1}{n} \sum_{z \in S} K(w, u; z), \qquad (4.3.2) \\ w(u; S) &= \arg\min_{w \in W} K(w, u; S). \qquad (4.3.3) \end{aligned}$$

Then, $M(u; S)$ is a smooth function. Formally, we state the theoretical results as follows.

Lemma 4.1. *Assume that $h \in \mathcal{H}$. Let $p > 0$. Then, $M(u; S)$ satisfies*

1. *$\min_u M(u; S)$ has the same global solution set as $\min_w R_S(w)$.*
2. *The gradient of $M(u; S)$ is $\nabla_u M(u; S) = p(u - w(u; S))$.*
3. *$M(u; S)$ is convex in u.*
4. *$M(u; S)$ is $2p$-gradient Lipschitz continuous.*
5. *$M(u; S)$ has bounded gradient norm L.*

The proof of Lemma 4.1 is due to (Rockafellar, 1976) and also provided in Sec. 4.7.1.

Remark 1: Since $\min_u M(u; S)$ has the same global solutions as $\min_w R_S(w)$, Principle 1 is ensured. As for algorithms, a natural way is to perform gradient descent to $M(u; S)$. Classical convergence results of GD on smooth loss ensure Principle 2. What remains unclear is Principle 3. It is because

$$\begin{aligned} \min_u M(u; S) &= \min_u \min_{w \in W} \frac{1}{n} \sum_{z \in S} K(w, u; z) \qquad (4.3.4) \\ &\neq \min_u \frac{1}{n} \sum_{z \in S} \min_{w \in W} K(w, u; z). \qquad (4.3.5) \end{aligned}$$

Table 4.2: Comparison of the choice of using Moreau envelopes.

	Principle 1	Principle 2	Principle 3
Eq. (4.3.4)	✓	✓	unclear
Eq. (4.3.5)	✗	✓	✓

$\min_u M(u;S)$ is not in a standard finite sum formulation, but in a min-min formulation. Then, the analytical tools of gradient descent on smooth loss (Hardt et al., 2016) cannot be applied.

Remark 2. To apply the analytical tools of (Hardt et al., 2016), an alternative choice is to apply SGD to the problem in Eq. (4.3.5). However, Eq. (4.3.5) has different global solutions from the original problem. We list the choice of using Moreau envelopes in Table 4.2.

It suffices to prove the 'unclear' part in Table 4.2. By Lemma 4.1, the estimate of the gradient requires the estimate of the solution of the minimization problem $\min_w K(w,u;S)$. Depending on whether we solve the subproblems exactly or not, we have the exact approach and inexact approach.

4.3.2 Exact Approach

We first consider the exact approach, which is the gradient descent to $M(u;S)$.

Theorem 4.2. *Assume h is a convex, L-Lipschitz function. Suppose we run GD on the smoothed surrogate adversarial loss $M(u;S)$ defined in Eq. (4.3.2) with fixed stepsize $\alpha \le 1/\sqrt{T}$ for $T \ge 4p^2$ steps. Then, the optimization and generalization gap satisfies*

$$\mathcal{E}_{opt} \le \mathcal{O}(1/T\alpha) \quad \text{and} \quad \mathcal{E}_{gen} \le \left(\frac{2L^2T\alpha}{n}\right). \tag{4.3.6}$$

Remark 3: The proof is deferred to Sec. 4.7.2. In summary, to prove Thm. 4.2 is to build the recursion from $\|u_S^t - u_{S'}^t\|$ to $\|u_S^{t+1} - u_{S'}^{t+1}\|$. To this aim, the key ingredient is the following error bound of $\|w(u;S) - w(u;S')\|$.

Lemma 4.1. *Assume that $h \in \mathcal{H}$. Let $p > 0$. For neighbouring S and S', we have*

$$\|w(u; S) - w(u; S')\| \leq 2L/(np).$$

Thm. 4.2 is our first main result. It shows that the exact approach eliminates the generalization error floor.

However, the exact approach requires the exact minimization of $K(w, u; S)$, which is sometimes computationally intractable. To address this issue, we consider the inexact approach below.

4.3.3 The Inexact Approach

The inexact approach is to estimate $\nabla_u M(u; S)$ by inexactly solving $\min_w K(w, u; S)$. To this aim, we perform multiple steps of SGD to the subproblem $\min_w K(w, u; S)$, attaining an estimate $\bar{w}(u)$ of the true $w(u)$, and then use $\bar{w}(u)$ to estimate $\nabla_u M(u; S)$. We refer to this algorithm as Moreau envelope-SGD (ME-SGD).

Algorithm 1 Moreau envelope-SGD

1: Initialize w^0, u^0;
2: Choose stepsize $c_s^t > 0$ and $\alpha_t > 0$;
3: **for** $t = 0, 1, 2, \ldots, T$ **do**
4: Let $w_0^t = w^t$;
5: **for** $s = 0, 1, 2, \cdots, N$ **do**
6: Draw a sample z_s^t from S uniformly;
7: $w_{s+1}^t = P_W(w_s^t - c_s^t \nabla_w K(w_s^t, u^t; z_s^t))$;
8: **end for**
9: $w^{t+1} = w_N^t$;
10: $u^{t+1} = u^t + \alpha_t p(w^{t+1} - u^t)$;
11: **end for**

In Step 7 in Alg. 1, we run SGD on $K(w, u, S)$ *w.r.t* w to find a solution given u. In step 10, we run GD on $K(w, u, S)$ *w.r.t* u. To provide the upper bounds of the

optimization gap and generalization gap of Alg. 1, we need the following Lemma for the inner optimization.

Lemma 4.2. *Given t and u^t, suppose we run SGD on $K(w, u^t, S)$ w.r.t. w with stepsize $c_s^t \leq 1/ps$ for N steps. w_N^t is approximately the minimizer with an error C_1^2/N, i.e.,*

$$E\|w_N^t - w(u^t)\|^2 \leq \frac{C_1^2}{N},$$

where $C_1 = (L + pD_W)/p$.

Lemma 4.2 provides the optimization error of the inner loop. In words, if we run the inner loop for sufficient steps, we can approximate the smoothed loss $M(u; S)$. Below we provide the convergence guarantee and uniform stability of ME-SGD with sufficient number of steps in the inner loop.

Theorem 4.3 (Convergence guarantee of ME-SGD)**.** *Suppose h is convex and L-Lipschitz. In Alg. 1, if we choose inner stepsize $c_s^t \leq 1/ps$, number of steps in inner loop $N = T$, outer stepsize $\alpha \leq 1/\sqrt{T}$, $T \geq 4p^2$, the optimization gap satisfies*

$$\mathcal{E}_{opt} \leq \frac{\|u^0 - u^*\|^2 + 2pC_1D_W + (L + pD_W)^2}{2T\alpha} = \frac{C_2}{T\alpha}, \tag{4.3.7}$$

where $C_2 = \|u^0 - u^*\|^2/2 + pC_1D_W + (L + pD_W)^2/2$.

Theorem 4.4 (Generalization bound of ME-SGD)**.** *Assume that h is convex and L-Lipschitz. In Alg. 1, if we choose inner stepsize $c_s^t \leq 1/ps$, number of steps in inner loop $N = n^2$, outer stepsize $\alpha_t \leq 1/\sqrt{T}$, $T \geq 4p^2$, the generalization gap satisfies*

$$\mathcal{E}_{gen} \leq L\left(\frac{2C_1p}{n} + \frac{2L}{n}\right)\sum_{t=1}^{T}\alpha_t = \frac{C_3}{n}\sum_{t=1}^{T}\alpha_t, \tag{4.3.8}$$

where $C_3 = L(4L + 2pD_W)$.

Thm. 4.3 and 4.4 are the main results of this chapter. For fixed stepsize $\alpha_t = \alpha$, it shows that Alg. 1 has training loss $\mathcal{O}(1/T\alpha)$ and has optimal generalization bound in $\mathcal{O}(T\alpha/n)$.

Interpretation of Number of Steps. In practice, if we use batch size 1 and go through the whole dataset in each epochs, T can be viewed as the number of epochs, and N can be viewed as the number of samples.

Minimax Optimal. A minimax lower bound of the excess risk for the function class $\mathcal{H}$ is given in (Nemirovskij & Yudin, 1983):

$$\min_{w} \max_{\mathcal{D}} \mathbb{E}_{S\sim\mathcal{D}^n}[R_{\mathcal{D}}(w) - \min_{w\in W} R_{\mathcal{D}}(w)] \geq \frac{LD_W}{C_4\sqrt{n}},$$

where C_4 is a universal constant. SGD does not achieve the minimax lower bound because of the error floor. In ME-SGD, let $T\alpha = \sqrt{C_2 n/C_3}$, we obtain the optimal excess risk with respect to T and α, *i.e.,* Excess risk $\leq \mathcal{E}_{opt} + \mathcal{E}_{gen} \leq 2\sqrt{\frac{C_2C_3}{n}}$. Therefore, ME-SGD with optimal stopping criterion achieves the minimax lower bound in $\Omega(1/\sqrt{n})$ of excess risk. Inversely, by combining the lower bound of excess risk and the upper bound of optimization error, we can obtain a lower bound in $\Omega(T\alpha/n)$ for the generalization gap.

4.3.4 Further Comparison with Existing Algorithms

Table 4.3: Comparison of SGD, weight decay, proximal update, stochastic weight averaging, and ME-SGD. Only Moreau envelope-SGD reduces the error floor in the generalization bound.

	Operation on w	Operation on u	Uniform Stability	No error floor
SGD	Minimize w on $R_S(w)$	No operation on u	$\mathcal{O}(\sqrt{T}\alpha + T\alpha/n)$	✗
Weight decay	Minimize w on $K(w,u;S)$	Set $u=0$	$\mathcal{O}(\sqrt{T}\alpha + T\alpha/n)$	✗
Proximal update	Minimize w on $R_S(w)$	Update rule in Eq. (4.3.9)	$\mathcal{O}(\sqrt{T}\alpha + T\alpha/n)$	✗
SWA	Minimize w on $R_S(w)$	Minimize u on $K(w,u;S)$	$\mathcal{O}(\sqrt{T}\alpha + T\alpha/n)$	✗
ME-SGD	Minimize w on $K(w,u;S)$	Minimize u on $K(w,u;S)$	$\mathcal{O}(T\alpha/n)$	✓

In Alg. 1, Step 7 is just to run SGD on $K(w,u;z) = h(w;z) + p\|w-u\|^2/2$ instead of $h(w;z)$. The additional term can be viewed as a regularization term similar to weight decay. Step 10 is a model averaging step similar to stochastic weight averaging (SWA). We compare ME-SGD with some existing similar algorithms in detail. The summary of the comparison is provided in Table 4.3. We can see that only ME-SGD can reduce

the generalization error floor.

Weight Decay. Weight decay (WD) is to add a ℓ_2 regularization to the empirical loss. The loss function with WD is $h(w;z)+p\|w\|^2/2$. Therefore, if we replace Step 10 by $u=0$ in Alg. 1, ME-SGD reduces to a simple weight decay regularization technique. Since the loss function with WS is not smooth, following the analysis in (Bassily et al., 2020) we will obtain the same uniform stability bounds.

Proximal Update. The proximal update is to apply an update rule,

$$P_{f,\alpha}(w)=\arg\min_u \frac{1}{2}\|w-u\|^2+\alpha f(u), \tag{4.3.9}$$

after a stochastic gradient update. Both proximal update and ME-SGD use the Moreau envelope function, but the algorithms are different. The stability analysis of the proximal update is given in (Hardt et al., 2016), Def. 4.5 and Lemma 4.6. It is proved that the proximal update is 1-expansive if $f(\cdot)$ is convex. Therefore, the generalization bound of the proximal update is no larger than that of SGD. In non-smooth cases, SGD induces an error floor. The proximal update induces the same uniform stability bounds as SGD in non-smooth settings and might not eliminate generalization error floor.

Stochastic Weight Averaging. Stochastic weight averaging (Izmailov et al., 2018) suggests using the weighted average of the iterates rather than the final one for inference. The update rules of SWA is $u^{t+1}=\tau^t u^t+(1-\tau^t)w^{t+1}$. By simply applying the analytical tools (Lemma 3.1) in the work of (Bassily et al., 2020), we can see that SWA is not guarantee to eliminate the generalization error floor.

SWA can be regarded as ME-SGD ($p\to 0$). In Alg. 1, if we denote $\tau^t=1-\alpha^t p$, Step 10 can be view as a weight averaging step. In Thm. 4.4, it is required that $\alpha^t\le 1/2p$. Then, $\tau^t=(1-\alpha^t p)\ge 1/2$. Therefore, by fixing $\alpha^t p$ to be constant and letting $p\to 0$, ME-SGD is reduced to SWA. Our analysis provides an understanding of the generalization ability of SWA.

4.4 Weakly-Convex Cases

Our main result can be extended to weakly-convex cases. Let $l > 0$. A function is said to be $-l$-weakly convex if $f(x) + l\|x\|^2/2$ is convex. In this case, we require $p > l$ such that $M(u; S)$ is strongly convex. Firstly, we extend Lemma 4.1, 4.1, and 4.2 to weakly convex cases, which are Lemma 4.2, 4.3, and 4.4. Based on the Lemma, we can derive the stability-based generalization bounds for both exact (Thm. 4.5) and inexact (Thm. 4.6) approaches.

Theorem 4.5. *Assume that h is a weakly-convex, L-Lipschitz function. Suppose we run GD on the smoothed surrogate adversarial loss $M(u; S)$ defined in Eq. (4.3.2) with diminishing stepsize $\alpha \leq (p-l)/(2p^2 - pl)t$ for T steps. Then, the generalization gap satisfies*

$$\mathcal{E}_{gen} \leq \mathcal{O}\left(\frac{2L^2T}{(2p-l)n}\right). \tag{4.4.1}$$

In this case, the bound also does not contain a non-vanishing term. The proof based on the error bound (Lemma 4.3) and the decomposition in the proof of Thm. 4.2.

Theorem 4.6 (Generalization bound of ME-SGD)**.** *Assume that h is $-l$-weakly convex and L-Lipschitz. In Alg. 1, if we choose inner stepsize $c_s^t \leq 1/(p-l)s$, number of steps in inner loop $N = n^2$, outer stepsize $\alpha_t \leq (p-l)/(2p^2 - pl)t$ for T, the generalization gap satisfies*

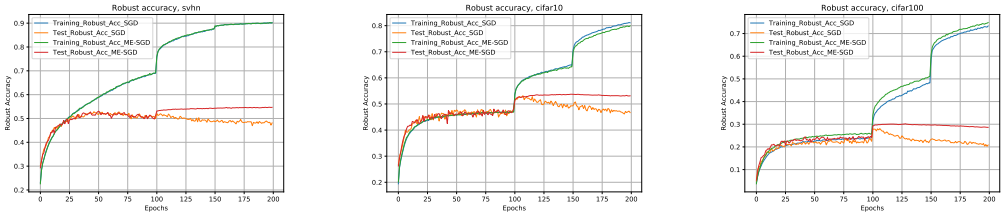
$$\mathcal{E}_{gen} \leq \mathcal{O}\left(\frac{C_3T}{(2p-l)n}\right). \tag{4.4.2}$$

where $C_3 = L(4L + 2pD_W)$.

Remark: Notice that existing uniform stability analysis of SGD on non-smooth losses requires convexity assumption. One of the benefit of ME-SGD is that the analysis can be extended to weakly-convex cases. The generalization bounds do not contain the error floor.

4.5 Experiments

Figure 4.1: Comparison of Generalization Gap induced by SGD and ME-SGD for the toy example.



(a) (b) (c)

Figure 4.2: Robust test accuracy of adversarial training using SGD and ME-SGD on SVHN, CIFAR-10, and CFAR-100.

In this section, we consider experiments on three problems with non-smooth losses. We first consider a toy example with L_1 loss that perfectly matches the assumptions. Then we consider two adversarial robustness problems with adversarial loss and TRADES loss. Our experiments verify the theoretical results and show that employing the Moreau envelopes tools can reduce the generalization error.

Training setting of ME-SGD. The learning rate of SGD depends on the following problems. We set the learning rate for ME-SGD (for the choice of c_s^t in Alg. 1) the

same as that of SGD for comparison. Because of the similarity of ℓ_2 regularization term of weight decay and the proximal term in $K(w, u; z)$, we set $p = 5 \times 10^{-4}$, which is a common choice of weight decay. The step size α_t of updating u is set to be 50, then $\tau = 1 - \alpha p = 0.995$.

4.5.1 L_1 loss

Let the L_1 loss be $h(w, z) = \|w - z\|_1$. It is 1-Lipschitz, since

$$|h(w_1, z) - h(w_2, z)| = \quad |\|w_1 - z\|_1 - \|w_2 - z\|_1| \leq \|w_1 - w_2\|_1,$$

for all z. It is non-smooth due to the ℓ_1-norm, and it is convex. Therefore, L_1-loss $\in \mathcal{H}$.

We perform a toy experiment on L_1 loss. Let the true distribution be a 10-dimensions Gaussian distribution, $\mathcal{D} = \mathcal{N}(0, I)$. We sample 10 to 100000 data to train w with SGD and ME-SGD. All the experiments are repeated 5 runs. The results are shown in Fig. 4.1. When the number of samples increases, the generalization gap induced by SGD does not converge to zero. While using ME-SGD, the gap converges to 0 fastly.

4.5.2 Adversarial Loss and TRADES Loss

Adversarial Loss. Let the loss function

$$h(w; z) = \max_{\|x - x'\| \leq \epsilon} \ell(f_w(x'), y),$$

where ϵ is the perturbation intensity. Here $f_w(\cdot)$ is a neural network parameterized by w, and $z = (x, y)$ is the input-label pair. If the neural networks are defined in a compact domain, *i.e.,* $\|x\| \leq B, \forall x \in \mathcal{X}$, the loss function $h(w; z)$ is L-Lipschitz. It is shown that adversarial loss is L-Lipschitz given the standard loss is L-Lipschitz, and it is non-smooth even if the standard loss is smooth (Xiao et al., 2022).

TRADES Loss. Let the TRADES loss be

$$h(w;z) = \ell(f_w(x'), y) + \beta \max_{\|x-x'\|\leq\epsilon} \ell(f_w(x'), f_w(x)),$$

where β is a hyperparameter (Zhang et al., 2019). Similar to the adversarial loss, the inner maximization problem induces the non-smoothness of the TRADES loss.

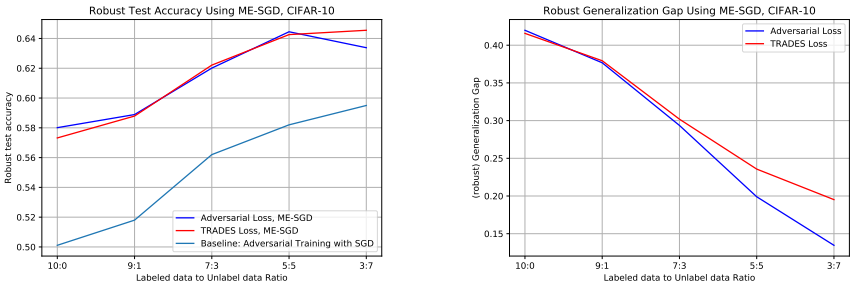
In adversarial robustness settings, the generalization gap we introduced is applied to the adversarially robust generalization gap, which is the robust test error - robust training error. It is shown that adversarially-trained models suffer from severe overfitting issues (Rice et al., 2020). From the perspective of uniform stability, it might be due to the generalization error floor induced by SGD. In the following experiments, we will show that ME-SGD can mitigate the overfitting issue.

To have a first glance at how ME-SGD mitigates robust overfitting, we consider the experiments on a lightweight model, PreActResNet-18, on SVHN, CIFAR-10, and CIFAR-100 to plot the training procedure. For the attack algorithms, we use ℓ_∞-PGD-10 (Madry et al., 2017), $\epsilon = 8/255$. The attack step size is set to be $\epsilon/4$. We use piece-wise learning rates, which are equal to $0.1, 0.01, 0.001$ for epochs 1 to 100, 101 to 150, and 151 to 200, respectively.

The training procedures on SVHN and CIFAR-10/100 are provided in Fig. 4.2. For SGD, the robust test accuracy starts to decrease at around the 100^{th} epoch, which is called robust overfitting. Using ME-SGD, the robust overfitting issue is much milder. These experiments verify the generalization bounds of ME-SGD.

Sample Complexity. Secondly, we study the sample complexity provided in Thm. 4.4. We use Wide-ResNet-28×10 with Swish activation function for better test accuracy instead of ResNet-18. The training setting mainly follows the work of (Gowal et al., 2020). The total number of epochs is 400. Other training settings are similar to the experiments on ResNet-18. CIFAR-10 only contains 50K training samples. We adopt the additional pseudo-label dataset introduced in (Carmon et al., 2019) to study the sample complexity. Increasing the percentage of pseudo-label data is an approximation

of increasing the training data.



(a) (b)

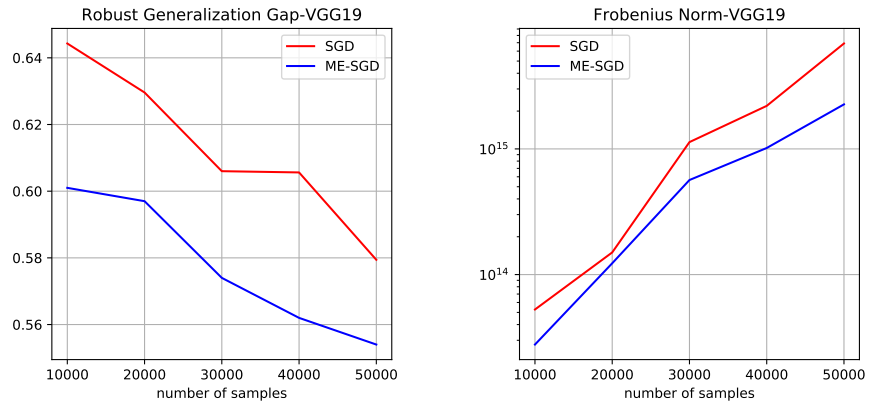
Figure 4.3: Robust test accuracy and generalization gap in the experiments of training CIFAR-10 using ME-SGD.

Robust Generalization Gap. In Fig. 4.3, we show the robust test accuracy (a) and adversarial generalization gap (b). The results are consistent with the theorem that ME-SGD reduces a term in the generalization bounds.

Weight Norms. It is shown in Chapter 2 that robust generalization gap is positively related to the product of weight norms. In ME-SGD, the regularization term $\|w - u\|^2$ regularize the weight w to its moving average u, which is a milder regularization comparing with weight decay. In Fig. 4.4, we can see that the weight norms induced by ME-SGD are smaller, which leads to smaller robust generalization gap.

Table 4.4: Robust test accuracy on TRADES loss. $\epsilon = 8/255$. Model: WideResNet-28×10 with Swish activation function. Training data: Labeled to unlabeled data ratio: 3:7.

Adv Loss	Algorithm	Clean	AutoAttack
CIFAR-10	SGD	90.93±0.25%	58.41±0.25%
	ME-SGD	91.51±0.20%	59.14±0.18%
TRADES	Algorithm	Clean	AutoAttack
CIFAR-10	SGD	88.36%	59.45%
	ME-SGD	85.33±0.13%	62.41±0.11%
CIFAR-100	SGD	59.38%	26.07%
	ME-SGD	59.25±0.22%	28.54±0.19%



(a) (b)

Figure 4.4: Robust generalization gaps and the product of weight norms in the experiments of training CIFAR-10 using SGD and ME-SGD.

In Table 4.4, we provide the robust test performance of ME. The baseline performance on CIFAR-10 is reported in (Gowal et al., 2020). We can see that the performance of ME-SGD is comparable in the same settings used in (Gowal et al., 2020). We adopt the AutoAttack (Croce & Hein, 2020), which is a collection of four attacks in default settings, to evaluate the performance of robust test errors. Notice that the state-of-the-art performance of adversarial robustness is obtained using large models (*e.g.,* WideResNet-106 $\times$ 16) and DDPM-generated data (Rebuffi et al., 2021). More resources are needed to run large models.

4.6 Conclusion

In this chapter, we study a question: can we eliminate the generalization error floor in the non-smooth loss minimization problem? Using tools from Moreau envelopes, we consider a smoothed version of SGD. We prove it has the same convergence guarantee as SGD and eliminates the generalization error floor. ME-SGD attains near-optimal uniform stability bound in non-smooth settings. Our work can lead to a better theoretical understanding of generalization in machine learning. We show that ME-SGD is

effective in practice. It especially improves robust test accuracy in adversarial training problems. We hope ME-SGD inspires more algorithms in deep learning.

4.7 Proof of Theorems

4.7.1 Proof of Lemma 4.1

We first state the Lemma in weakly-convex cases.

Lemma 4.2. *Assume that h is $-l$-weakly convex. Let $p > l$. Then, $M(u; S)$ satisfies*

1. $\min_u M(u; S)$ *has the same global solution set as* $\min_w R_S(w)$.
2. *The gradient of* $M(u; S)$ *is* $\nabla_u M(u; S) = p(u - w(u; S))$.
3. $M(u; S)$ *is* $pl/(p - l)$*-weakly convex in* u.
4. $M(u; S)$ *is* $(2p^2 - pl)/(p - l)$*-gradient Lipschitz continuous.*
5. $M(u; S)$ *has bounded gradient norm* L.

Lemma 4.1 hold by letting $l = 0$. To simplify the notation, we use $M(u)$ as a short hand notation of $M(u; S)$. Similar to $h(u)$, $K(u)$, and $w(u)$.

1. Let $w^* \in \arg\min R_S(w)$. We have

$$R_S(w^*) = K(w^*, u = w^*, S) \geq K(w(u), u = w^*, S) \geq R_S(w(u = w^*)).$$

Then, the equality holds. Therefore, $w = u = w^*$ is the optimal solution of both $\min_w R_S(w)$ and $\min_u M(u; S)$.

2. Since $K(w, u)$ is a $(p - l)$-strongly convex function, $w(u)$ is unique. Then

$$M(u) = h(w(u)) + \frac{p}{2}\|w(u) - u\|^2.$$

By taking the derivative of $M(u)$ with respect to u, we have

$$\nabla_u M(u) = \left[\frac{\partial w(u)}{\partial u}\right]^T \cdot \nabla_{w(u)} h(w(u)) + \left[\frac{\partial w(u)}{\partial u} - I\right]^T \cdot p(w(u) - u) \quad (4.7.1)$$

$$= \left[\frac{\partial w(u)}{\partial u}\right]^T \cdot (\nabla_{w(u)} h(w(u)) + p(w(u) - u)) + p(u - w(u)). \quad (4.7.2)$$

Since $w(u)$ is the optimal solution of $K(w, u)$, we have

$$\nabla_{w(u)} K(w(u), u) = \nabla_{w(u)} h(w(u)) + p(w(u) - u) = 0. \quad (4.7.3)$$

Therefore, the first term in 4.7.2 is equal to zero. We have $\nabla_u M(u) = p(u - w(u))$.

3. In Eq. (4.7.3), take the derivatives with respect to u on both sides, we have

$$\left[\frac{\partial w(u)}{\partial u}\right]^T \nabla_w^2 h(w) + p(\left[\frac{\partial w(u)}{\partial u}\right]^T - I) = 0. \quad (4.7.4)$$

Organizing the terms, we have

$$\left[\frac{\partial w(u)}{\partial u}\right]^T (\nabla_w^2 h(w) + pI) = pI. \quad (4.7.5)$$

Since $h(w)$ is $-l$-weakly convex, $\nabla_w^2 h(w) + pI$ is positive definite. Then,

$$\left[\frac{\partial w(u)}{\partial u}\right]^T \prec \frac{p}{p-l} I. \quad (4.7.6)$$

Then,

$$\nabla_u^2 M(u) = [\frac{\partial}{\partial u} p(u - w(u))]^T = p(I - \left[\frac{\partial w(u)}{\partial u}\right]^T) \succ p(1 - \frac{p}{p-l})I. \quad (4.7.7)$$

Therefore, $M(u)$ is a $pl/(p-l)$-weakly convex function.

4. By Eq. (4.7.6), we have

$$\|\nabla M(u_1) - \nabla M(u_2)\| = p\|u_1 - w(u_1) - u_2 - w(u_2)\| \leq p(1 + \frac{p}{p-l})\|u_1 - u_2\|. \quad (4.7.8)$$

Therefore, $M(u;S)$ is $(2p^2-pl)/(p-l)$-gradient Lipschitz continuous.

5. By Eq. (4.7.3),

$$\|\nabla_u M(u)\| = \|p(u-w(u))\| = \|\nabla_w h(w)\| \leq L. \tag{4.7.9}$$

□

4.7.2 Proof of Thm. 4.2

The optimization error is a standard result of runing GD on smooth objective function.

We focus on the proof of generalization bounds. Thm. 4.2 is not obtained from the work of (Hardt et al., 2016). Notice that

$$M(u;S) = \min_{w\in W} \frac{1}{n}\sum_{z\in S} K(w,u;z) \neq \frac{1}{n}\sum_{z\in S} \min_{w\in W} K(w,u;z).$$

$\min_u M(u;S)$ is not a finite sum problem. The analysis in (Hardt et al., 2016) can only be applied to finite sum problems. Thm. 4.2 requires a different proof. In summary, there are two steps:

1. Build the recursion from $\|u_S^t - u_{S'}^t\|$ to $\|u_S^{t+1} - u_{S'}^{t+1}\|$;
2. Unwind the recursion.

The main challenge comes from the first step. To this end, we develop a new error bound and a different decomposition. We first introduce the following error bound.

Lemma 4.3. *In weakly-convex case, for neighbouring S and S', we have*

$$\|w(u;S) - w(u;S')\| \leq 2L/(n(p-\ell)).$$

Proof. By the $(p-l)$-strongly convexity of $K(w,u;S)$, we have

$$\begin{aligned}
& (p-l)\|w(u;S)-w(u;S')\| \\
\leq\ & \|\nabla K(w(u;S),u;S)-\nabla K(w(u;S'),u;S)\| \\
\leq\ & \|\nabla K(w(u;S),u;S)-\nabla K(w(u;S'),u;S')\| \\
& +\frac{1}{n}\|\nabla h(w(u;S'),z_i)\|+\frac{1}{n}\|\nabla h(w(u;S'),z_i')\| \\
=\ & \frac{1}{n}\|\nabla h(w(u;S'),z_i)\|+\frac{1}{n}\|\nabla h(w(u;S'),z_i')\| \\
\leq\ & \frac{2L}{n},
\end{aligned}$$

where the second inequality is due to the definition of $K(w,u;S)$, the third one is due to the first-order optimally condition, and the last inequality is because of the bounded gradient of $h(w;z)$. □

Next, we move to the proof of Thm. 4.2.

Step 1.

$$\begin{aligned}
& \|u_S^{t+1}-u_{S'}^{t+1}\| \\
=\ & \|u_S^t-u_{S'}^t-\alpha_t(\nabla M(u_S^t;S)-\nabla M(u_{S'}^t;S'))\| \\
\leq\ & \|u_S^t-u_{S'}^t-\alpha_t(\nabla M(u_S^t;S)+\nabla M(u_{S'}^t;S))\|+\alpha_t\|\nabla M(u_{S'}^t;S')-\nabla M(u_{S'}^t;S)\| \\
\leq\ & \|u_S^t-u_{S'}^t\|+\alpha^t\|\nabla M(u_{S'}^t;S')-\nabla M(u_{S'}^t;S)\| \\
=\ & \|u_S^t-u_{S'}^t\|+\alpha^t p\|u_{S'}^t-u_{S'}^t-w(u_{S'}^t,S)+w(u_{S'}^t,S')\| \\
\leq\ & \|u_S^t-u_{S'}^t\|+\frac{2L\alpha_t}{n},
\end{aligned}$$

where the second inequality is due to the non-expansive property of convex function , the last inequality is due to Lemma 4.3.

Step 2. Unwinding the recursion, we have

$$\|u_S^T - u_{S'}^T\| \leq \frac{2L\sum_{t=1}^T \alpha_t}{n}.$$

□

4.7.3 Proof of Lemma 4.2

We first extend Lemma 4.2 to weakly-convex case for further deiscussion.

Lemma 4.4. *Given t and u^t, suppose we run SGD on $K(w, u^t, S)$ w.r.t. w with stepsize $c_s^t \leq 1/(p-l)s$ for N steps. w_N^t is approximately the minimizer with an error C_1^2/N, i.e.,*

$$E\|w_N^t - w(u^t)\|^2 \leq \frac{C_1^2}{N},$$

where $C_1 = (L + pD_W)/(p-l)$.

It can be obtained from classical strongly-convex optimization results. Since

$$\|\nabla_w K(w, u; z)\| = \|\nabla_w h(w; z) + p(w-u)\| \leq L + pD_W,$$

$K(w, u; z)$ has bounded gradient $L_K = L + pD_W$. By (Nemirovski et al., 2009), running SGD on $K(w, u; S)$ with stepsize $c_s \leq 1/s(p-l)$ iccurs an optimization error in

$$E\|w_N - w(u)\|^2 \leq \frac{C_1^2}{N},$$

where $C_1 = (L + pD_W)/(p-l)$.

4.7.4 Proof of Thm. 4.3

Proof. Let $A_{t+1} = \frac{1}{2}\|u^{t+1} - u^*\|^2$ and $a_{t+1} = \frac{1}{2}\mathbb{E}\|u^{t+1} - u^*\|^2$.

$$\begin{aligned} A_{t+1} &= \frac{1}{2}\|u^{t+1} - u^*\|^2 \\ &\leq \frac{1}{2}\|u^t - \alpha_t \nabla_u K(w_N^t, u^t; S) - u^*\|^2 \\ &\leq A_t + \frac{1}{2}\alpha_t^2 L_K^2 - \alpha_t \langle \nabla_u K(w_N^t, u^t; S), u^t - u^* \rangle \\ &= A_t + \frac{1}{2}\alpha_t^2 L_K^2 - \alpha_t \langle \nabla_u M(u^t; S), u^t - u^* \rangle \\ &\quad + \alpha_t \langle \nabla_u M(u^t; S) - \nabla_u K(w_N^t, u^t; S), u^t - u^* \rangle. \end{aligned}$$

By taking the expectation on both sides and Rearranging the terms, we have

$$\begin{aligned} & \alpha_t \mathbb{E}[M(u^t) - M(u^*)] \\ \leq\ & a_t - a_{t+1} + \frac{1}{2}\alpha_t^2 L_K^2 + \alpha_t \mathbb{E}\langle \nabla_u M(u^t; S) - \nabla_u K(w_N^t, u^t; S), u^t - u^* \rangle \end{aligned} \tag{4.7.10}$$

Since

$$\begin{aligned} & \mathbb{E}\langle \nabla_u M(u^t; S) - \nabla_u K(w_N^t, u^t; S), u^t - u^* \rangle \\ \leq\ & \|\nabla_u M(u^t; S) - \nabla_u K(w_N^t, u^t; S)\| \mathbb{E}\|u^t - u^*\| \\ \leq\ & \frac{pC_1 D_W}{\sqrt{N}}, \end{aligned}$$

Eq. (4.7.10) becomes

$$\begin{aligned} & \alpha_t \mathbb{E}[M(u^t) - M(u^*)] \\ \leq\ & a_t - a_{t+1} + \frac{1}{2}\alpha_t^2 L_K^2 + \frac{\alpha_t p C_1 D_W}{\sqrt{N}}. \end{aligned}$$

Let $N \geq T$. Take the summation over t. We obtain that

$$\begin{aligned} & \sum_{t=1}^{T} \alpha_t \mathbb{E}[M(u^t) - M(u^*)] \\ \leq \quad & a_0 + \frac{1}{2}\sum_{t=1}^{T} \alpha_t^2 L_K^2 + \frac{\sum_{t=1}^{T} \alpha_t p C_1 D_W}{\sqrt{T}}. \end{aligned}$$

There exists $t \leq T$, such that

$$\mathbb{E}[M(u^t) - M(u^*)] \leq \frac{a_0 + \frac{1}{2}\sum_{t=1}^{T} \alpha_t^2 L_K^2 + \frac{\sum_{t=1}^{T} \alpha_t p C_1 D_W}{\sqrt{T}}}{\sum_{t=1}^{T} \alpha_t}.$$

Considering constant step $\alpha \leq 1/\sqrt{T}$, we have $\alpha \leq 1/T\alpha$ and $\alpha\sqrt{T} \leq 1$. Therefore,

$$\begin{aligned} \mathbb{E}[M(u^t) - M(u^*)] \quad & \leq \quad \frac{2a_0 + T\alpha^2 L_K^2 + 2\alpha\sqrt{T} p C_1 D_W}{2T\alpha} \\ & \leq \quad \frac{\|u^0 - u^*\|^2 + L_K^2 + 2pC_1 D_W}{2T\alpha} \\ & = \quad \frac{C_2}{T\alpha}. \end{aligned}$$

Since $M(u; S)$ and $R_S(w)$ have the same global solutions, we can use both of them to measure the optimization error. Above is the optimization error defined in $M(u; S)$. Below we provide the optimization error defined in $R_S(w)$.

$$\mathbb{E}[R_S(w(u^t)) - R_S(w^*)] \leq \mathbb{E}[M(u^t) - M(u^*)] \leq \frac{C_2}{T\alpha}.$$

Notice that the choices of algorithm output are slightly different. Therefore, we have

$$\mathcal{E}_{opt} \leq \frac{C_2}{T\alpha},$$

where $C_2 = \|u^0 - u^*\|^2/2 + pC_1 D_W + (L + pD_W)^2/2$. □

4.7.5 Proof of Thm. 4.4

Proof. We decompose $\|u_S^{t+1} - u_{S'}^{t+1}\|$ as

$$\begin{aligned}
& \mathbb{E}\|u_S^{t+1} - u_{S'}^{t+1}\| \\
= \; & \mathbb{E}\|u_S^t - \alpha_t \nabla_u K(w_{N,S}^t, u_S^t; S) - u_{S'}^t + \alpha_t \nabla_u K(w_{N,S'}^t, u_{S'}^t; S')\| \\
\leq \; & \mathbb{E}\|u_S^t - \alpha_t \nabla_u M(u_S^t; S) - u_{S'}^t + \alpha_t \nabla_u M(u_{S'}^t; S')\| \\
+ \; & 2\alpha_t \mathbb{E}\|\nabla_u K(w_{N,S}^t, u_S^t; S) - \nabla_u M(u_S^t; S)\| \\
\leq \; & \mathbb{E}\|u_S^t - u_{S'}^t\| + \frac{2L\alpha_t}{n} + 2\alpha_t p \mathbb{E}\|w_N^t - w(u^t)\| \\
\leq \; & \mathbb{E}\|u_S^t - u_{S'}^t\| + \frac{2L\alpha_t}{n} + 2\alpha_t p \frac{C_1}{\sqrt{N}}.
\end{aligned}$$

Let $N \geq n^2$. Unwind the recursion and let u^T be the output of the algorithm. We have

$$\begin{aligned}
\mathcal{E}_{gen} &\leq L\mathbb{E}\|u_S^T - u_{S'}^T\| \\
&\leq \frac{L(2L + 2C_1 p)\sum_{t=1}^T \alpha_t}{n} \\
&= \frac{C_3 \sum_{t=1}^T \alpha_t}{n}.
\end{aligned}$$

If we choose $w(u^T)$ to be the algorithm output, we have

$$\begin{aligned}
\mathcal{E}_{gen} &\leq L\mathbb{E}\|w(u_S^T; S) - w(u_{S'}^T; S')\| \\
&= L\mathbb{E}\|u_S^T - \frac{1}{p}\nabla M(u_S^T, S) - u_{S'}^T - \frac{1}{p}\nabla M(u_{S'}^T; S'))\| \\
&\leq L\mathbb{E}\|u_S^T - u_{S'}^T\| + \frac{2L^2}{np} \\
&\leq \frac{L(2L + 2C_1 p)\sum_{t=1}^T \alpha_t}{n} + \frac{2L^2}{np} \\
&= \mathcal{O}\left(\frac{\sum_{t=1}^T \alpha_t}{n}\right). \qquad (4.7.11)
\end{aligned}$$

where the first equality is due to $\nabla M(u; S) = p(u - w(u))$, the second inequality is due to the non-expansive propertiy of $M(u; S)$. □

4.7.6 Proof of Thm. 4.5

The proof contains two steps. The first step is to build the recursion, which is based on the error bound and the decomposition of Thm. 4.2. The second step is to unwind the recursion, which is adopt from the analysis of uniform stability in non-smooth case (Hardt et al., 2016).

Step 1.

$$\begin{aligned}
& \|u_S^{t+1} - u_{S'}^{t+1}\| \\
= \; & \|u_S^t - u_{S'}^t - \alpha_t(\nabla M(u_S^t; S) - \nabla M(u_{S'}^t; S'))\| \\
\leq \; & \|u_S^t - u_{S'}^t - \alpha_t(\nabla M(u_S^t; S) + \nabla M(u_{S'}^t; S))\| + \alpha_t\|\nabla M(u_{S'}^t; S') - \nabla M(u_{S'}^t; S)\| \\
\leq \; & \|u_S^t - u_{S'}^t\| + \alpha_t\|\nabla M(u_S^t; S) - \nabla M(u_{S'}^t; S)\| + \alpha^t\|\nabla M(u_{S'}^t; S') - \nabla M(u_{S'}^t; S)\| \\
\leq \; & (1 + \alpha_t\beta)\|u_S^t - u_{S'}^t\| + \alpha^t\|\nabla M(u_{S'}^t; S') - \nabla M(u_{S'}^t; S)\|, \qquad (4.7.12)
\end{aligned}$$

where the first and second inequalities are due to triangular inequality. The last inequality is due to the gradient Lipschitz of $M(u; S)$ and $\beta = (2p^2 - pl)/(p - l)$. Then,

$$\begin{aligned}
& \alpha^t\|\nabla M(u_{S'}^t; S') - \nabla M(u_{S'}^t; S)\| \\
= \; & \alpha^t p\|u_{S'}^t - u_{S'}^t - w(u_{S'}^t, S) + w(u_{S'}^t, S')\| \\
\leq \; & \frac{2Lp\alpha_t}{(p-l)n}, \qquad (4.7.13)
\end{aligned}$$

where the first inequality is due to the form of $\nabla M(u; S)$, the last equality is due to Lemma 4.3. Combining Eq. (4.7.12) and (4.7.13), we have

$$\begin{aligned}
& \|u_S^{t+1} - u_{S'}^{t+1}\| \\
\leq \; & (1 + \alpha_t\beta)\|u_S^t - u_{S'}^t\| + \frac{2Lp\alpha_t}{(p-l)n}. \qquad (4.7.14)
\end{aligned}$$

Step 2. Let S and S' be two samples of size n differing in only a single example. Consider two trajectories $w_1^1, \ldots, w_1^T$ and $w_2^1, \ldots, w_2^T$ induced by running SGD on sample

S and S', respectively. Let $\delta_t = \|w_1^t - w_2^t\|$. Let $t_0 \in \{0, 1, \ldots, n\}$, be the iteration that $\delta_{t_0} = 0$, but SGD picks two different samples form S and S' in iteration $t_0 + 1$, then

$$\mathcal{E}_{gen} \leq \frac{t_0}{n} B + L \,\mathbb{E}\left[\delta_T \mid \delta_{t_0} = 0\right] . \tag{4.7.15}$$

Let $\Delta_t = \mathbb{E}\left[\delta_t \mid \delta_{t_0} = 0\right]$, and $\alpha_t \leq c/(\beta t)$. Then,

$$\begin{aligned}
\Delta_{t+1} &\leq (1 + \alpha_t \beta)\Delta_t + \left(\frac{2Lp}{(p-l)n}\right)\alpha_t \\
&= \left(1 + \frac{c\beta}{t}\right)\Delta_t + \left(\frac{2Lp}{(p-l)n}\right)\frac{c}{t} \\
&\leq \exp\left(\frac{c\beta}{t}\right)\Delta_t + \left(\frac{2Lp}{(p-l)n}\right)\frac{c}{t} .
\end{aligned}$$

Here we used the fact that $1 + x \leq \exp(x)$ for all x.

Using the fact that $\Delta_{t_0} = 0$, we can unwind this recurrence relation from T down to $t_0 + 1$. This gives

$$\begin{aligned}
\Delta_T &\leq \sum_{t=t_0+1}^{T} \left\{ \prod_{k=t+1}^{T} \exp\left(\tfrac{\beta c}{k}\right) \right\} \left(\frac{2Lp}{(p-l)n}\right)\frac{c}{t} \\
&= \sum_{t=t_0+1}^{T} \exp\left(\beta c \sum_{k=t+1}^{T} \tfrac{1}{k}\right) \left(\frac{2Lp}{(p-l)n}\right)\frac{c}{t} \\
&\leq \sum_{t=t_0+1}^{T} \exp\left(\beta c \log(\tfrac{T}{t})\right) \left(\frac{2Lp}{(p-l)n}\right)\frac{c}{t} \\
&= \left(\frac{2Lp}{(p-l)n}\right) c T^{\beta c} \sum_{t=t_0+1}^{T} t^{-\beta c - 1} \\
&\leq \left(\frac{2Lp}{(p-l)n}\right)\frac{1}{\beta c} c \left(\frac{T}{t_0}\right)^{\beta c} \\
&\leq \frac{2Lp}{\beta(p-l)n}\left(\frac{T}{t_0}\right)^{\beta c} ,
\end{aligned}$$

Plugging this bound, we get

$$\mathcal{E}_{gen} \leq \frac{Bt_0}{n} + \frac{2L^2 p}{\beta(p-l)n}\left(\frac{T}{t_0}\right)^{\beta c}.$$

Let $c = 1/\beta$, let $t_0 = 1$, then

$$\mathcal{E}_{gen} \leq \frac{B + 2L^2 T}{(2p-l)n} \leq \mathcal{O}\left(\frac{2L^2 T}{(2p-l)n}\right).$$

□

4.7.7 Proof of Thm. 4.6

We decompose $\|u_S^{t+1} - u_{S'}^{t+1}\|$ as

$$\begin{aligned}
& \mathbb{E}\|u_S^{t+1} - u_{S'}^{t+1}\| \\
= \;& \mathbb{E}\|u_S^t - \alpha_t \nabla_u K(w_{N,S}^t, u_S^t; S) - u_{S'}^t + \alpha_t \nabla_u K(w_{N,S'}^t, u_{S'}^t; S')\| \\
\leq \;& \mathbb{E}\|u_S^t - \alpha_t \nabla_u M(u_S^t; S) - u_{S'}^t + \alpha_t \nabla_u M(u_{S'}^t; S')\| \\
+ \;& 2\alpha_t \mathbb{E}\|\nabla_u K(w_{N,S}^t, u_S^t; S) - \nabla_u M(u_S^t; S)\| \\
\leq \;& \mathbb{E}\|u_S^t - u_{S'}^t\| + \frac{2Lp\alpha_t}{(p-l)n} + 2\alpha_t p \mathbb{E}\|w_N^t - w(u^t)\| \\
\leq \;& \mathbb{E}\|u_S^t - u_{S'}^t\| + \frac{2Lp\alpha_t}{(p-l)n} + 2\alpha_t p \frac{C_1}{\sqrt{N}}.
\end{aligned}$$

Let $N \geq n^2$. Since $C_1 = (L + pD_W)/(p-l)$, we have

$$\begin{aligned}
& \mathbb{E}\|u_S^{t+1} - u_{S'}^{t+1}\| \\
\leq \;& \mathbb{E}\|u_S^t - u_{S'}^t\| + \frac{2Lp\alpha_t}{(p-l)n} + 2\alpha_t p \frac{C_1}{\sqrt{N}} \\
\leq \;& \mathbb{E}\|u_S^t - u_{S'}^t\| + \frac{(4L + 2pD_w)p\alpha_t}{(p-l)n} \\
\leq \;& \mathbb{E}\|u_S^t - u_{S'}^t\| + \frac{(4L + 2pD_w)p\alpha_t}{(p-l)n}.
\end{aligned} \tag{4.7.16}$$

By replacing L by $(4L + 2pD_w)$ in the second step of the proof of Thm. 4.5, we have

$$\mathcal{E}_{gen} \leq \frac{B + 2L(4L + 2pD_w)T}{(2p - l)n} \leq \mathcal{O}\left(\frac{C_3 T}{(2p - l)n}\right).$$

□

□ **End of chapter.**

Chapter 5

On-manifold Adversarial Examples

Summary

One of the hypotheses of the existence of the adversarial examples is the off-manifold assumption: adversarial examples lie off the data manifold. However, recent researches showed that on-manifold adversarial examples also exist. In this chapter, we revisit the off-manifold assumption and study a question: at what level is the poor adversarial robustness of neural networks due to on-manifold adversarial examples? Since the true data manifold is unknown in practice, we consider two approximated on-manifold adversarial examples on both real and synthesis datasets. On real datasets, we show that on-manifold adversarial examples have greater attack rates than off-manifold adversarial examples on both standard-trained and adversarially-trained models. On synthetic datasets, theoretically, we prove that on-manifold adversarial examples are powerful, yet adversarial training focuses on off-manifold directions and ignores the on-manifold adversarial examples. Furthermore, we provide analysis to show that the properties derived theoretically can also be observed in practice. Our analysis suggests that on-manifold adversarial examples are important. We should pay more attention to on-manifold adversarial examples to train robust models.

5.1 Introduction

One of the hypotheses of the existence of adversarial examples is the off-manifold assumption (Szegedy et al., 2013): In recent years, deep neural networks (DNNs) (Krizhevsky et al., 2012; Hochreiter & Schmidhuber, 1997) have become popular and successful in many machine learning tasks. They have been used in different problems with great success. But DNNs are shown to be vulnerable to adversarial examples (Szegedy et al., 2013; Goodfellow et al., 2014b). A well-trained model can be easily attacked by adding small perturbations to the images. One of the hypotheses of the existence of adversarial examples is the off-manifold assumption (Szegedy et al., 2013):

Clean data lies in a low-dimensional manifold. Even though the adversarial examples are close to the clean data, they lie off the underlying data manifold.

Based on the assumption, DNNs might only fit the data in the manifold and perform badly on adversarial examples out of the manifold. Many researches support this point of view. Pixeldefends (Song et al., 2017) leveraged a generative model to show that adversarial examples lie in a low probability region of the data distribution. The work of (Gilmer et al., 2018; Khoury & Hadfield-Menell, 2018) studied the geometry of adversarial examples. They showed that adversarial examples are related to the high dimension of the data manifold and they are conducted in the directions off the data manifold. The work of (Ma et al., 2018) used Local Intrinsic Dimensionality (LID) to characterize the adversarial region and argues that the adversarial subspaces are of low probability, and lie off (but are close to) the data submanifold.

One of the most effective approaches to improve the adversarial robustness of DNNs is to augment adversarial examples to the training set, i.e., adversarial training. However, the performance of adversarially-trained models are still far from satisfactory. The off-manifold assumption provided a possible explanation for this phenomenon: DNNs

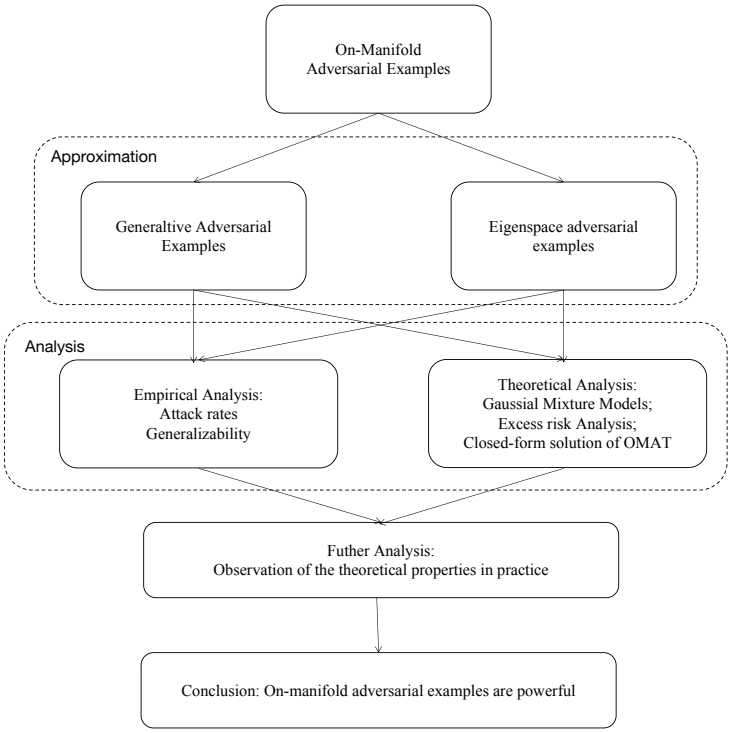


Figure 5.1: Frame diagram of the idea of the chapter.

work well on a low dimensional manifold but might not work well on a high dimensional manifold.

In recent years, researchers found that on-manifold adversarial examples also exist. They can fool the target models (Lin et al., 2020) and boost clean generalization. Therefore, the off-manifold assumption may not be a perfect hypothesis explaining the existence of adversarial examples. It motivates us to revisit the off-manifold assumption. Specifically, we study the following question:

At what level is the poor adversarial robustness of neural networks due to on-manifold adversarial examples?

The main difficulty in studying this question is that the true data manifold is unknown in practice. To have a closer look at on-manifold adversarial examples, we use two approaches to approximate the on-manifold adversarial examples. The first one is generative adversarial examples (Gen-AE). Generative models, such as generative

Table 5.1: Summary of the analysis of our chapter. In Sec. 5.3, we show that (approximated) on-manifold adversarial examples (Gen-AE, Eigen-AE) have higher attack rates than off-manifold adversarial examples. In Sec. 5.4, we provide a theoretical analysis of on-manifold attacks on GMMs, where the true data manifold is known. In Sec. 5.5, we provide further analysis to show the similarity of these four cases.

	Real datasets	Synthetic datasets (GMMs, known manifold)
Gen-AE	Attack rates (Table 5.2)	Excess risk: Thm. 5.1 Adversarial distribution: Thm. 5.3
Eigen-AE	Attack rates (Fig. 5.2)	Excess risk: Thm. 5.2 Adversarial distribution: Thm: 5.4

adversarial networks (GAN) (Goodfellow et al., 2014a) and variational autoencoder (VAE) (Kingma & Welling, 2013), are used to craft adversarial examples. Since the generative model is an approximation of the data manifold, the crafted data approximately lies in the data manifold. We perform perturbation in the latent space of the generative model. Since the first approach relies on the quality of the generative models, we consider the second approach: using the eigenspace of the training dataset to approximate the data manifold, which we call eigenspace adversarial examples (Eigen-AE). In this method, the crafted adversarial examples are closer to the original samples. For more details, see Section 5.3.

To start with, we provide experiments of on-manifold attacks on both stan-dard-trained and adversarially-trained models on common datasets. The experiments show that on-manifold attacks are powerful, with higher attack rates than off-manifold adversarial examples. It helps justify that the off-manifold assumption might not be a perfect hypothesis of the existence of adversarial examples.

The experiments motivate us to study on-manifold adversarial examples from a theoretical perspective. Since the true data manifold is unknown in practice, we provide a theoretical study on synthetic datasets using Gaussian mixture models (GMMs). In this case, the true data manifold is given. We study the *excess risk* (Thm. 5.1 and Thm. 5.2) and *adversarial distribution shift* (Thm. 5.3 and Thm. 5.4) of these two types of on-manifold adversarial examples. Our main technical contribution is providing closed-form solutions to the min-max problems of on-manifold adversarial

training. Our theoretical results show that on-manifold adversarial examples incur a large excess risk and foul the target models. Compared to regular adversarial attacks, we show how adversarial training focuses on off-manifold directions and ignores the on-manifold adversarial examples.

Finally, we provide a comprehensive analysis to connect the four settings (two approximate on-manifold attacks on both real and synthetic datasets). We show the similarity of these four cases: 1) the attacks directions of Gen-AE and Eigen-AE are similar on common datasets, and 2) the theoretical properties derived using GMMs (Thm. 5.1 to Thm. 5.4) can also be observed in practice.

Based on our study, we find that on-manifold adversarial examples are important for adversarial robustness. We emphasize that we should pay more attention to on-manifold adversarial examples for training robust models.

5.2 On-manifold Adversarial Attacks

Adversarial Attacks. Given a classifier f_θ and a sample data (x, y). The goal of *regular adversarial attacks* is to find an adversarial example x' to foul the classifier f_θ. A widely studied norm-based adversarial attack is to solve the optimization problem $\max_{\|x-x'\|\leq\varepsilon} \ell(f_\theta(x'), y)$, where $\ell(\cdot,\cdot)$ is the loss function and ε is the perturbation intensity. In this chapter, we focus on on-manifold adversarial attacks. We must introduce additional constraints to restrict x' in the data manifold and preserve the label y. In practice, the true data manifold is unknown. Assuming that the true data manifold is a push-forward function $p(z) : \mathcal{Z} \to \mathcal{X}$, where $\mathcal{Z}$ is a low dimensional Euclidean space and $\mathcal{X}$ is the support of the data distribution. One way to approximate $p(z)$ is to use a generative model $G(z)$ such that $G(z) \approx p(z)$. The second way is to consider the Taylor expansion of $p(z)$ and use the first-order term to approximate $p(z)$. They correspond to the following two approximated on-manifold adversarial examples.

Generative Adversarial Examples. One method to approximate the data manifold is to use generative models, such as GAN and VAE. Let $G : \mathcal{Z} \to \mathcal{X}$ be a generative

model and $I : \mathcal{X} \to \mathcal{Z}$ be the inverse mapping of $G(z)$. Generative adversarial attack can be formulated as the following problem

$$\max_{\|z'-I(x)\| \leq \varepsilon} \ell(f_\theta(G(z')), y). \tag{5.2.1}$$

Then, $x' = G(z')$ is an (approximate) on-manifold adversarial example. In this method, $\|x - x'\|$ can be large. To preserve the label y, we use the conditional generative models (e.g. C-GAN (Mirza & Osindero, 2014) and C-VAE (Sohn et al., 2015), i.e. the generator $G_y(z)$ and inverse mapping $I_y(x)$ are conditioned on the label y. In the experiments, we use two widely used gradient-based attack algorithms, *fast gradient sign method* (FGSM) (Goodfellow et al., 2014b) and *projected gradient descend* (PGD) (Madry et al., 2017) in the latent space $\mathcal{Z}$, which we call GFGSM and GPGD, respectively.

Eigenspace Adversarial Examples. The on-manifold adversarial examples found by the above method are not norm-bounded. To study norm-based on-manifold adversarial examples, we use the eigenspace to approximate the data manifold. Consider the following problem

$$\begin{aligned} \max_{x'} \quad & \ell(f_\theta(x'), y) \\ s.t. \quad & A_y^\perp (x - x') = 0, \ \|x - x'\| \leq \varepsilon \end{aligned} \tag{5.2.2}$$

where the rows of A_y are the eigenvectors, which corresponds to the top eigenvalues of the co-variance matrix of the training data with label y. Then, x' is restricted in the eigenspace. A baseline algorithm for standard adversarial attack is PGD attack (Madry et al., 2017). To make a fair comparison, we should use the same algorithm to solve the attack problem. Notice that the inner maximization problem in Eq. (5.2.2) cannot be solved by PGD in general. Because the projection step is an optimization problem. We consider ℓ_2 adversarial attacks in this case. Then, Eq. (5.2.2) is equivalent to

$$\max_{\|\Delta z\|_2 \leq \varepsilon,} \quad \ell(f(x + \Delta z^T A_y), y). \tag{5.2.3}$$

We can use PGD to find eigenspace adversarial examples. In this case, the perturbation constraint is a subset of the constraint of regular adversarial attacks. For comparison, we consider the case that the rows of A_y is the eigenvectors corresponds to the bottom eigenvalues. We call this type of on-manifold and off-manifold adversarial examples as top eigenvectors and bottom eigenvectors subspace adversarial examples.

Target Models. In this chapter, we aim to study the performance of on-manifold adversarial attacks on both standard-trained model and adversarially-trained model. For standard training, only the original data is used in training. For adversarial training, adversarial examples are augmented to the training dataset. For ablation studies, we also consider on-manifold adversarial training, *i.e.,* on-manifold adversarial examples crafted by the above mentioned algorithms are augmented to the training dataset.

5.3 Performance of On-manifold Adversarial Attacks

In this section, we provide experiments in different settings to study the performance of on-manifold adversarial attacks.

Table 5.2: Test accuracy of different defense algorithms (PGD-AT, GPGD-AT, and joint-PGD-AT) against different attacks (regular attacks (FGSM, PGD) and generative attacks (GFGSM, GPGD) on MNIST, CIFAR-10, and ImageNet.

MNIST	clean data	FGSM-Attack	PGD-Attack	GFGSM-Attack	GPGD-Attack
Std training	**98.82%**	47.38%	3.92%	42.37%	10.74%
PGD-AT	98.73%	96.50%	95.51%	52.17%	15.53%
GPGD-AT	98.63%	20.63%	2.11%	**99.66%**	**96.78%**
joint-PGD-AT	98.45%	**97.31%**	**95.70%**	99.27%	96.03%
CIFAR-10	clean data	FGSM-Attack	PGD-Attack	GFGSM-Attack	GPGD-Attack
Std training	**91.80%**	15.08%	5.39%	7.07%	3.41%
PGD-AT	80.72%	**56.42%**	**50.18%**	14.27%	8.51%
GPGD-AT	78.93%	10.64%	3.21%	40.18%	26.66%
joint-PGD-AT	79.21%	50.19%	49.77%	**42.87%**	**28.54%**
ImageNet	clean data	FGSM-Attack	PGD-Attack	GFGSM-Attack	GPGD-Attack
Std training	74.72%	2.59%	0.00%	/	0.26%
PGD-AT	73.31%	48.02%	**38.88%**	/	7.23%
GPGD-AT	78.10%	21.68%	0.03%	/	**27.53%**
joint-PGD-AT	**77.96%**	**49.12%**	37.86%	/	20.53%

5.3.1 Experiments of Generative Adversarial Attacks

To study the performance of generative adversarial attacks, we compare the performance of different attacks (FGSM, PGD, GFGSM, and GPGD) versus different models (standard training, PGD-adversarial training, and GPGD-adversarial training).

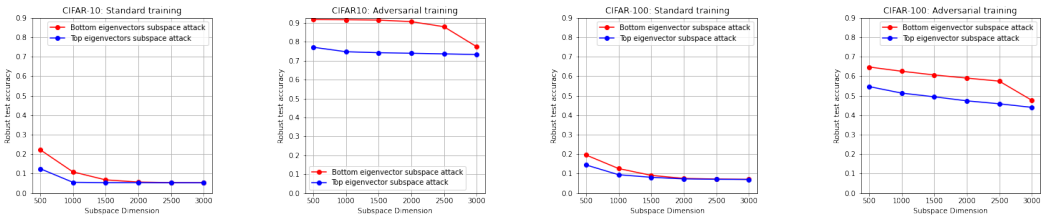
Experiments Setup. In this section we report our experimental results on training LeNet on MNIST (LeCun et al., 1998), ResNet18 (He et al., 2016) on CIFAR-10 (Krizhevsky et al., 2009), and ResNet50 on ImageNet (Deng et al., 2009). On MNIST, we use $\varepsilon = 0.3$ and 40 steps PGD for adversarial training and use $\varepsilon = 1$ and 40 steps PGD for generative adversarial training. On CIFAR-10 and CIFAR-100, we use $\varepsilon = 8/255$, PGD-10 for adversarial training, and $\varepsilon = 0.1$, PGD-10 for generative adversarial training. On ImageNet, we adopt the settings in (Lin et al., 2020) and use $\varepsilon = 4/255$, PGD-5 for adversarial training, and $\varepsilon = 0.02$, PGD-5 for generative adversarial training. The choice of ϵ in the latent space is based on the quality of the generative examples. In the outer minimization, we use the SGD optimizer with momentum 0.9 and weight decay $5 \cdot 10^{-4}$.

Generative AT Cannot Defend Regular Attack, and Vice Versa. In Table 5.2, the test accuracy of GPGD-AT vs PGD-attack is 3.21%, which means that on-manifold adversarial training cannot defend a regular norm-based attack. Similarly, the test accuracy of PGD-AT vs GPGD-Attack is 15.53%, a adversarially-trained model preforms badly on on-manifold attacks. The results on the experiments on CIFAR-10 and ImageNet are similar. PGD-AT is not able to defend GPGD-Attack, and vice versa. In the experiments on CIFAR-10 and ImageNet, we can see that generative attacks have greater attack rates than regular PGD-attacks.

Joint-PGD Adversarial Training. Based on the previous discussion, we may wonder whether we can improve the model robustness by augmenting both on-manifold and regular PGD-adversarial examples to the training set. On MNIST, the jointly-

trained model achieves 96.03% and 95.70% test accuracy against GPGD-Attack and PGD-Attack, which are similar to the test accuracy of the single-trained models. We can see similar results on CIFAR-10 and ImageNet. Since generative adversarial examples are not closed to the clean data because of the large Lipschitz of the generative models. We study the performance of norm-bounded on-manifold adversarial examples in the next subsection.

5.3.2 Eigenspace Adversarial Examples



(a) (b) (c) (d)

Figure 5.2: Robust test error of standard training and adversarial training against eigenspace attack on CIFAR-10 and CIFAR-100. The x-axis is the dimension of the restricted subspace. The performance against top eigenvectors subspace attacks is shown in the blue lines. The performance against bottom eigenvectors subspace attacks is shown in red lines. (a) Standard-trained model on CIFAR-10. (b) Adversarially-trained model on CIFAR-10. (c) Standard-trained model on CIFAR-100. (d) Standard-trained model on CIFAR-100.

Experiment Setups. In Fig. 5.2, we show the results of the experiments of standard training and ℓ_2-adversarial training against ℓ_2-eigenspace adversarial attacks on CIFAR-10 (Krizhevsky et al., 2009) and CIFAR-100. For adversarial training, we use 20 steps PGD-attack with $\varepsilon = 0.5$ for the inner maximization problem. For the eigenspace attack, we restrict the subspace in different dimensions. The results of top eigenvectors subspace attacks restricted in the subspace spanned by the first 500, 1000, 1500, 2500, and 3000 eigenvectors are shown in the blue lines. Correspondingly, the results of bottom eigenvectors subspace attacks restricted in the subspace spanned by the last eigenvectors are shown in the red lines.

Target Model. The target model in Fig. 5.2 is Wide-ResNet-28. In the experiments on CIFAR-10, for the standard-trained model, the clean accuracy is 92.14%. For the adversarial training model, the clean accuracy is 95.71%, and the robust accuracy against ℓ_2-PGD-20 is 73.22%. On CIFAR-100. for standard training, the clean accuracy is 80.34%. For adversarial training, the clean accuracy is 67.18%, and the robust accuracy against PGD-10 is 43.92%.

On-manifold Adversarial Examples Have Greater Attack Rates on Standard-Trained Models. In Fig. 5.2 (a), if we restrict the attack directions in the subspace spanned by the first or last 500 eigenvectors, the robust accuracy are 12% and 22%. As we increase the dimension of the subspace, the test accuracy will converge to the robust accuracy of standard PGD-attack, 5.27%. If the dimension of subspace increases to 3072, the subspace attacks are equivalent to standard PGD-attacks since the dimension of $\mathcal{X}$ is $3 \times 3 \times 32 = 3072$ on CIFAR-10. On-manifold adversarial examples have greater attack rates than off-manifold adversarial examples against the standard training model. The results in Fig. 5.2 (c) are similar.

On-manifold Adversarial Examples Have Greater Attack Rates on Adversarially-Trained Model. Now, we turn to Fig. 5.2 (b). If we restrict the attack directions in the first or last 1000 eigenvectors subspace, the robust accuracy is 58.74% and 66.43% against bottom and top eigenvectors subspace attacks, respectively. Similar for other numbers of dimensions. Similarly for the experiments on CIFAR-100 in Fig. 5.2 (d). We can see that adversarial training can work well on off-manifold attacks, but the performance on on-manifold attacks is not good enough.

Eigenspace Adversarial Training. In this setting, the perturbation constraint is a subset of the constraint of regular adversarial training. The performance of this algorithm is worse than that of regular adversarial training.

Adversarial Training Ignores On-manifold Adversarial Examples. In these experiments, all the on-manifold adversarial examples are norm-bounded. Adversarial training still performs badly on this kind of attack. In Theorem 5.2 in the next section, we will show that it is because off-manifold adversarial examples have larger losses than on-manifold adversarial examples in the norm constraint. Then, norm-based adversarial training algorithms are not able to find the on-manifold adversarial examples in the small norm ball. Therefore, adversarial training cannot fit the on-manifold adversarial examples well.

Take-away Message. In short, Sec. 5.3 shows that (approximate) on-manifold adversarial examples are powerful. It has higher attack rates than off-manifold adversarial examples on both standard-trained and adversarially-trained models. Widely use adversarially-trained models performs badly against (approximate) on-manifold attacks.

5.4 Theoretical Analysis

In this section, we study the excess risk and the adversarial distribution shift of on-manifold adversarial training. We study the binary classification setting proposed by (Ilyas et al., 2019). In this setting, the true data manifold is known. We can derive the optimal closed-form solution of on-manifold adversarial attacks and adversarial training and provide insights to understand on-manifold adversarial examples.

5.4.1 Theoretical Model Setup

Gaussian Mixture Model. Assume that data points (x, y) are sampled according to $y \sim \{-1, 1\}$ uniformly and $x \sim \mathcal{N}(y\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$, where $\boldsymbol{\mu}_*$ and $\boldsymbol{\Sigma}_*$ denote the true mean and covariance matrix of the data distribution. For the data in the class $y = -1$, we replace x by $-x$, then we can view the whole dataset as sampled from $\mathcal{D} = \mathcal{N}(\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$.

Classifier. The goal of standard training is to learn the parameters $\boldsymbol{\Theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$ such that

$$\boldsymbol{\Theta} = \arg\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathcal{L}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \arg\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathbb{E}_{x\sim\mathcal{D}}[\ell(x; \boldsymbol{\mu}, \boldsymbol{\Sigma})], \tag{5.4.1}$$

where $\ell(\cdot)$ represents the negative log-likelihood function. The goal of adversarial training is to find

$$\boldsymbol{\Theta_r} = \arg\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathcal{L}_r(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \arg\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathbb{E}_{x\sim\mathcal{D}}[\max_{\|x-x'\|\leq\varepsilon} \ell(x'; \boldsymbol{\mu}, \boldsymbol{\Sigma})]. \tag{5.4.2}$$

We use $\mathcal{L}$ and $\mathcal{L}_r$ to denote the standard loss and adversarial loss. After training, we classify a new data point x to the class $\text{sgn}(\boldsymbol{\mu}^T\boldsymbol{\Sigma}^{-1}x)$.

Generative Model. In our theoretical study, we use a linear generative model, that is, probabilistic principle components analysis (P-PCA) (Tipping & Bishop, 1999). P-PCA can be viewed as linear VAE (Dai et al., 2017).

Given dataset $\{x_i\}_{i=1}^n \subset \mathbb{R}^d$, let $\boldsymbol{\mu}$ and $\boldsymbol{S}$ be the sample mean and sample covariance matrix. The eigenvalue decomposition of $\boldsymbol{S}$ is $\boldsymbol{S} = \mathbf{u}\boldsymbol{\Lambda}\mathbf{u}^T$, then using the first q eigenvactors, we can project the data to a low dimensional space. P-PCA is to assume that the data are generated by

$$x = \boldsymbol{W}z + \boldsymbol{\mu} + \boldsymbol{\epsilon} \quad \text{where} \quad z \sim \mathcal{N}(0, I)\ ,\ \boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 I),$$

$z \in \mathbb{R}^q$ and $\boldsymbol{W} \in \mathbb{R}^{d\times q}$. Then we have $x \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{W}\boldsymbol{W}^T + \sigma^2 I)$, $x|z \sim \mathcal{N}(\boldsymbol{W}z + \boldsymbol{\mu}, \sigma^2 I)$ and $z|x \sim \mathcal{N}(\boldsymbol{P}^{-1}\boldsymbol{W}^T(x - \boldsymbol{\mu}), \sigma^2\boldsymbol{P}^{-1})$ where $\boldsymbol{P} = \boldsymbol{W}^T\boldsymbol{W} + \sigma^2 I$. The maximum likelihood estimator of $\boldsymbol{W}$ and σ^2 are

$$\boldsymbol{W}_{\text{ML}} = \boldsymbol{U}_q(\boldsymbol{\Lambda}_q - \sigma^2_{\text{ML}} I)^{1/2} \text{ and } \sigma^2_{\text{ML}} = \frac{1}{d-q}\sum_{i=q+1}^{d} \lambda_i,$$

where $\boldsymbol{U_q}$ is the matrix of the first q columns of $\mathbf{u}$, $\boldsymbol{\Lambda_q}$ is the matrix of the first q eigenvalues of $\boldsymbol{\Lambda}$. In the following study, we assume that n is large enough such that we can learn the true $\boldsymbol{\mu}_*$ and $\boldsymbol{\Sigma}_*$. Thus we have $\boldsymbol{S} = \boldsymbol{\Sigma}_*$, $\boldsymbol{U}_q = \boldsymbol{U}_{q*}$, $\boldsymbol{\Lambda}_q = \boldsymbol{\Lambda}_{q*}$ for the

generative model.

Generative Adversarial Examples. To perturb the data in the latent space, data will go through the encode-decode process $x \to z \to \Delta z + z \to x'$. Specifically, we sample $x \sim \mathcal{D}$, then we encode $z = \arg\max q(z|x) = \boldsymbol{P}^{-1}\boldsymbol{W}^T(x - \boldsymbol{\mu}_*)$, add a perturbation Δz, and finally, we decode $x_{adv} = \arg\max p(x|z + \Delta z) = \boldsymbol{W}(z + \Delta z) + \boldsymbol{\mu}_*$. The following lemma shows that the generative adversarial examples can be rewritten in a new form.

Lemma 5.1 (Generative Adversarial Examples)**.** *Using generative model P-PCA, the adversarial examples can be rewritten as*

$$x_{adv} = x' + \boldsymbol{W}\Delta z \quad \textit{and} \quad x' \sim \mathcal{D}' = \mathcal{N}(\boldsymbol{\mu}_*, \boldsymbol{U}_*\boldsymbol{\Lambda}'\boldsymbol{U}_*^T),$$
$$\textit{with } \boldsymbol{\Lambda}' = \begin{bmatrix} (\boldsymbol{\Lambda_q} - \sigma^2 I)^2\boldsymbol{\Lambda_q}^{-1} & 0 \\ 0 & 0 \end{bmatrix}.$$

If the data lie in a q dimensional subspace, i.e. the covariance matrix $\boldsymbol{\Sigma}_*$ is rank q, we have $\boldsymbol{\Lambda}' = \boldsymbol{\Lambda}_*$. Then $\mathcal{D}' = \mathcal{D}$.

Remark. We consider other sampling strategies using generative models in 5.7. We show that the adversarial examples have the same form as in Lemma 5.1 with different $\boldsymbol{\Lambda}'$.

The adversarial expected risk of *generative adversarial training* can be rewritten as the following minimax problem

$$\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathcal{L}_{gat}(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \mathcal{D}') = \min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathbb{E}_{x' \sim \mathcal{D}'} \max_{\|\Delta z\| \le \varepsilon} \ell(x' + \boldsymbol{W}\Delta z, \boldsymbol{\mu}, \boldsymbol{\Sigma}). \tag{5.4.3}$$

Eigenspace Adversarial Examples. If the perturbation is restricted in the eigenspace spanned by U_q. the adversarial examples can be weritten as $x_{adv} = x + \boldsymbol{U}_q\Delta z$.

The adversarial expected risk of *eigenspace adversarial training* can be rewritten as

the following minimax problem

$$\min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathcal{L}_{eat}(\boldsymbol{\mu},\boldsymbol{\Sigma};\mathcal{D}) = \min_{\boldsymbol{\mu},\boldsymbol{\Sigma}} \mathbb{E}_{x\sim\mathcal{D}} \max_{\|\Delta z\|\leq\varepsilon} \ell(x + \boldsymbol{U}_q \Delta z, \boldsymbol{\mu},\boldsymbol{\Sigma}). \tag{5.4.4}$$

When $q = d$, $\boldsymbol{U}_q$ span the whole space. Eq. (5.4.4) reduces to the regular adversarial training.

5.4.2 Excess Risk Analysis

We consider the excess risk incurred by the optimal perturbation of on-manifold adversarial examples, i.e. $\mathcal{L}_{gat} - \mathcal{L}$ and $\mathcal{L}_{eat} - \mathcal{L}$ given the true $\boldsymbol{\Theta}_*$.

We consider the Lagrange penalty form of the inner maximization problem in Eq. (5.4.3), i.e., $\max \ell(x' + \boldsymbol{W}\Delta z, \boldsymbol{\mu}, \boldsymbol{\sigma}) - L\|\Delta z\|^2/2$, where L is the Lagrange multiplier. Similarly, we consider the Lagrange penalty form of the inner maximization problem in Eq. (5.4.4).

Theorem 5.1 (Excess risk of generative adversarial examples)**.** *Let $\mathcal{L}_{gat}$ and $\mathcal{L}$ be the loss in equations (5.4.3) and (5.4.1) respectively), given the non-robustly learned $\boldsymbol{\Theta}_* = (\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$, the excess risk is*

$$\begin{aligned}
&\mathcal{L}_{gat}(\boldsymbol{\Theta}_*, \mathcal{D}') - \mathcal{L}(\boldsymbol{\Theta}_*, \mathcal{D}) \\
=&\frac{1}{2}\sum_{i=1}^{q}\Big[(1 + \frac{\lambda_i - \sigma^2}{(L-1)\lambda_i + \sigma^2})^2 - 1\Big]\frac{\lambda_i'}{\lambda_i} + \frac{1}{2}\log\Big[\frac{\prod_{i=1}^{d}\lambda_i'}{\prod_{i=1}^{d}\lambda_i}\Big] + \frac{1}{2}\Big(\sum_{i=1}^{d}\frac{\lambda_i'}{\lambda_i} - d\Big).
\end{aligned}$$

If the data lie in a low dimensional manifold, rank$(\boldsymbol{\Sigma}_*) = q$, the excess risk is

$$\mathcal{L}_{gat}(\boldsymbol{\Theta}_*, \mathcal{D}') - \mathcal{L}(\boldsymbol{\Theta}_*, \mathcal{D}) = \Theta(qL^{-2}).$$

The optimal perturbation in the latent space will incur an excess risk in $\Theta(qL^{-2})$. The adversarial vulnerability depends on the dimension q and the Lagrange multiplier L. L is negatively related to the perturbation intensity ε. Then, we analyze the excess risk of eigenspace adversarial examples. Since the perturbation thresholds, ε, are on

different scales, the corresponding Lagrange multipliers L are different. We use L_2 in the following Theorem.

Theorem 5.2 (Excess risk of eigenspace adversarial examples)**.** *Let $\mathcal{L}_{eat}$ and $\mathcal{L}$ be the loss in equations (5.4.4) and (5.4.1) respectively, given the non-robustly learned $\boldsymbol{\Theta}_* = (\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$, denote λ_q be the q^{th} eigenvalue of $\boldsymbol{\Sigma}_*$, the excess risk satisfies*

$$\mathcal{L}_{eat}(\boldsymbol{\Theta}_*, \mathcal{D}) - \mathcal{L}(\boldsymbol{\Theta}_*, \mathcal{D}) \leq \mathcal{O}(q(\lambda_q L_2)^{-2}) \text{ and}$$
$$\mathcal{L}_{eat}(\boldsymbol{\Theta}_*, \mathcal{D}) - \mathcal{L}(\boldsymbol{\Theta}_*, \mathcal{D}) \geq \Omega((\lambda_q L_2)^{-2}).$$

Similar to the case of generative attacks, the adversarial vulnerability depends on the dimension q and the Lagrange multiplier $L2$.

Excess Risk of (Regular) Adversarial Examples. When $q = d$ and the data lie in a low dimensional manifold, i.e. $\lambda_{min} = 0$, the excess risk of regular adversarial examples equals to $+\infty$. Regular adversarial attacks focus on the directions corresponding to zero eigenvalues, i.e. the off-manifold directions. However, on-manifold adversarial examples will also incur large excess risks in $\Theta(qL^{-2})$ and $\Omega((\lambda_q L_2)^{-2})$. On-manifold adversarial examples are also powerful but regular adversarial attacks tend to ignore the on-manifold adversarial examples.

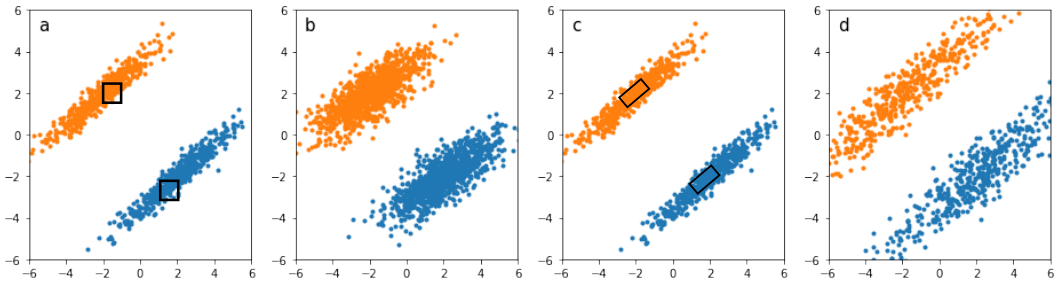


Figure 5.3: Demonstration and numerical simulation of theoretical analysis. Let the ellipses be the Gaussian data, and let the black block be the perturbation constraint with the original example in the center. (a) The constraint of eigenspace adversarial attacks; (b) Adversarial distribution shift of eigenspace adversarial training; (c) The constraint of generative adversarial attacks; (d) Adversarial distribution shifts of generative adversarial training.

5.4.3 Adversarial Distribution Shifts

In this subsection we study the optimal solution of optimization problem in Eq. (5.4.3) and (5.4.4). Since they are not standard min-max problems, we consider a modified problem of (5.4.3):

$$\min_{\boldsymbol{\mu},\boldsymbol{\Lambda}} \max_{\mathbb{E}_{x'}\|\Delta z\|=\varepsilon} \mathbb{E}_{x'\sim\mathcal{D}'}\ell(x' + \boldsymbol{W}\Delta z, \boldsymbol{\mu}, \boldsymbol{\Sigma}). \tag{5.4.5}$$

Similarly, we switch the order of expectation and maximization in problem (5.4.4). The following two theorems are our main results. They give the optimal solutions of generative adversarial training and eigenspace adversarial training.

Theorem 5.3 (Optimal solution of generative adversarial training)**.** *The optimal solution of the modified problem in Eq. (5.4.5) is*

$$\boldsymbol{\mu}_{eat} = \boldsymbol{\mu}_* \quad \textit{and} \quad \boldsymbol{\Sigma}_{eat} = \boldsymbol{U}_*\boldsymbol{\Lambda}^{gat}\boldsymbol{U}_*^T, \quad \textit{where}$$

$$\lambda_i^{gat} = \frac{1}{4}\Big[2\lambda_i' + \frac{4(\lambda_i - \sigma^2)}{L} + 2\lambda_i'\sqrt{1 + \frac{4(\lambda_i - \sigma^2)}{\lambda_i' L}}\Big],$$

$$\textit{for } i = 1 \le q, \quad \textit{and } \lambda_i^{gat} = \lambda_i' \quad \textit{for} \quad i > q.$$

We assume that the data lies in a q-dimensional manifold again. Then we have $\lambda_i^{ls}/\lambda_i = 1/2 + 1/L + \sqrt{1/4 + 1/L} \ge 1$ for $i \le q$ and $\lambda_i^{ls}/\lambda_i = 0$ for $i > q$. Generative adversarial training amplifies the largest q eigenvalues of the covariance matrix of the (adversarial) data distribution.

Theorem 5.4 (Optimal solution of eigenspace adversarial training)**.** *The optimal solution of the modified problem of eigenspace adversarial training*[1] *is*

$$\boldsymbol{\mu}_{eat} = \boldsymbol{\mu}_* \quad \textit{and} \quad \boldsymbol{\Sigma}_{eat} = \boldsymbol{U}_*\boldsymbol{\Lambda}^{eat}\boldsymbol{U}_*^T, \quad \textit{where}$$

[1]Remark: Thm. 5.2 and Thm. 5.4 provide the excess risk and adversarial distribution shift of regular adversarial training by setting $q = d$.

$$\lambda_i^{eat} = \frac{1}{4}\Big[2\lambda_i + \frac{4}{L_2} + 2\lambda_i\sqrt{1+\frac{4}{\lambda_i L_2}}\Big],$$
$$\textit{for } i = 1 \le q, \quad \textit{and } \lambda_i^{eat} = \lambda_i \quad \textit{for} \quad i > q.$$

Considering the ratio

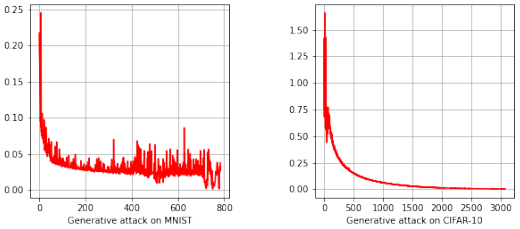
$$\frac{\lambda_i^{eat}}{\lambda_i} = \frac{1}{2} + \frac{1}{L_2\lambda_i} + \sqrt{\frac{1}{L_2\lambda_i} + \frac{1}{4}},$$

the ratio $\lambda_i^{eat}/\lambda_i$ is larger for smaller true eigenvalue λ_i. Therefore, eigenspace adversarial training amplifies the small eigenvalues of the eigenspace.

Optimal Solution of Regular Adversarial Training. When $q = d$, regular adversarial training amplifies the small eigenvalues of the whole space.

Demonstration of the Case $q = 2$. We give a numerical simulation in Fig. 5.3. The black blocks are the perturbation constraint with the original example in the center. Given the original Gaussian data (plots in (a)), we iteratively solve the max and min problem in Eq. (5.4.4), and the resulting data distribution is plotted in (b). We can see that the radius of the short axis of the ellipse is amplified. In Fig. 5.3 (d), we iteratively solve the max and min problem in Eq. (5.4.3), and we can see that the radius of the two axes of the ellipse is amplified.

Take-away Message. In the analysis of GMMs, we prove that regular adversarial attacks and defense only focus on off-manifold directions due to the magnitude of eigenvalues. However, on-manifold adversarial examples also incur large excess risk and lead to different adversarial distribution shift. This is why (regular) adversarial training performs badly against on-manifold attacks.



(a) (b)

Figure 5.4: (a) The average of the generative attack directions on MNIST. (b) The average of the generative attack directions on CIFAR-10.

5.5 Further Analysis of On-manifold Attacks

We provide further analysis of the theoretical properties on real dataset. We will see that the properties derived in Sec. 5.4 can also be observed in practice.

On-manifold Attack Directions. Given the training dataset $\mathcal{D}$, we conduct SVD on this dataset and let the eigenvectors span a basis of the original space. In Fig. 5.4 (a) and (b), we plot the absolute value of the average of all the generative adversarial examples in this space. We can see that generative attack focuses on the directions of top eigenvectors. Gen-AE and Eigen-AE have similar attack directions on real datasets.

Adversarial Distribution Shifts. Let $f_{at}(x)$ and $f_{gat}(x)$ be the adversarially trained and generative adversarially trained neural networks respectively. Given the clean dataset $\mathcal{D}$, let the robust dataset be

$$\mathcal{D}_{at} = \{x_{adv} | x_{adv} = \arg\max \ell(f_{at}(x), y), (x, y) \in \mathcal{D}\}$$

and $\mathcal{D}_{gat} = \{x_{adv} | x_{adv} = \arg\max \ell(f_{gat}(G(z)), y), (x, y) \in \mathcal{D}\}$. We perform SVD on $\mathcal{D}$, $\mathcal{D}_{at}$, and $\mathcal{D}_{gat}$. We plot the eigenvalues in Fig. 5.5. The first row is the experiments on MNIST, and the second row is the experiments on CIFAR-10. On MNIST, the number of dimensions is $1 \times 28 \times 28 = 784$. On CIFAR-10, the numbers of dimension is $3 \times 32 \times 32 = 3072$.

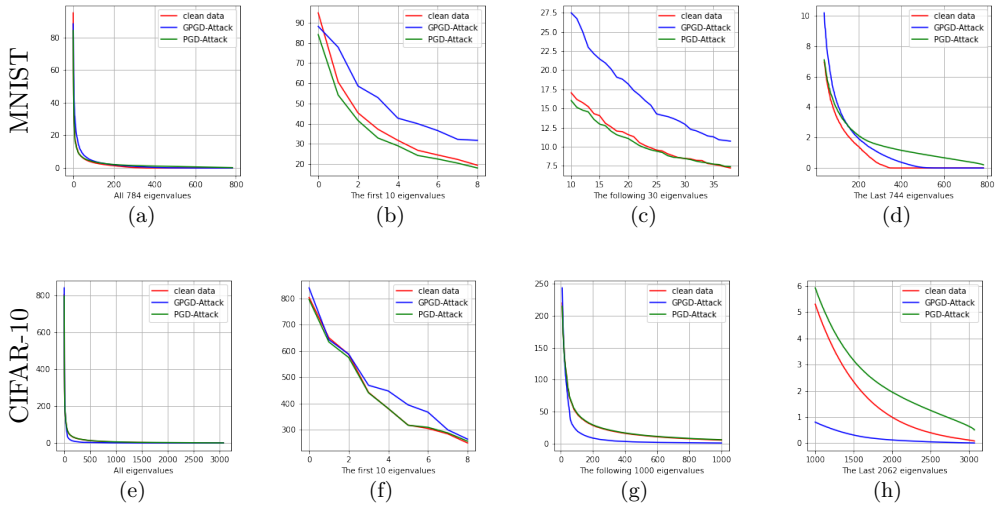


Figure 5.5: Adversarial distribution shifts on MNIST and CIFAR-10. The first row is the experiment on MNIST. The second row is the experiment on CIFAR-10. (a) and (e) plot all the eigenvalues of the dataset. (b) and (f): generative adversarial training (blue line) will amplify the large eigenvalue of the dataset, while adversarial training will not. (d) and (h) Adversarial training (green line) will amplify the small eigenvalues of the dataset, while generative adversarial training will not.

Generative Adversarial Training Amplifies the Top Eigenvalues. In Fig. 5.5 (b) and (f), we plot the top 10 eigenvalues. we can see that the eigenvalues of $\mathcal{D}_{gat}$ (blue line) is larger than that of $\mathcal{D}$ (red line). It means that generative adversarial training will amplify the large eigenvalues, which is consistent with Theorem 5.3. Regular adversarial training does not amplify the top eigenvalues.

Adversarial Training Amplifies the Bottom Eigenvalues. In Fig. 5.5 (d) and (h), we can see that the bottom eigenvalues of $\mathcal{D}_{at}$ (green line) is larger than that of $\mathcal{D}$ (red line), which is consistent with Theorem 5.4, while the bottom eigenvalues of $\mathcal{D}_{gat}$ is not larger than that of $\mathcal{D}$. Therefore, Adversarial training amplify the bottom eigenvalues.

Discussion on Adversarial Distribution Shifts. In general, it is hard to characterize the adversarial distribution shifts in detail since analyzing the minimax problem

of adversarial training is challenging. In these experiments, we show that the eigenvalues, which are second-order statistics of the dataset or the variance of the data distribution in the corresponding directions, are good indicators of the adversarial distribution shifts. Adversarial training tries to fit the adversarial examples in directions of small variance, while generative adversarial training tries to fit the adversarial examples in directions of large variance. Therefore, adversarial training ignores on-manifold adversarial examples, which are also important for adversarial robustness.

5.6 Conclusion

In this chapter, We show that on-manifold adversarial examples are also powerful, but adversarial training focuses on off-manifold directions and ignores the on-manifold adversarial examples. On-manifold adversarial examples are also important for adversarial training. We think that our analysis can inspire more theoretical research on adversarial robustness. For example, a well-designed norm-bounded on-manifold adversarial attacks algorithm may improve the performance of adversarial training.

5.7 Proof of the Theorems

5.7.1 Proof of Lemma 5.1

We first introduce the general form of Lemma 5.1. To perturb the data in the latent space, data will go through the encode-decode process $x \to z \to \Delta z + z \to x'$. Based on the probabilistic model, we may choose z with the highest probability or just sample it from the distribution we learned. Hence, we could have different strategies.

Strategy 1: Sample $x \sim \mathcal{D}$, encode $z = \arg\max q(z|x) = \boldsymbol{P}^{-1}\boldsymbol{W}^T(x - \boldsymbol{\mu}_*)$, add a perturbation Δz, and finally, decode $x_{adv} = \arg\max p(x|z + \Delta z) = \boldsymbol{W}(z + \Delta z) + \boldsymbol{\mu}_*$.

Strategy 2: Sample $x \sim \mathcal{D}$, then sample $z \sim q(z|x)$, add a perturbation Δz, and finally, sample $x_{adv} \sim p(x|z + \Delta z)$.

Strategy 3: Sample $z \sim \mathcal{N}(0, I)$, add a perturbation Δz, and then sample $x_{adv} \sim$

$p(x|z+\Delta z)$. In this strategy, x_{adv} can be viewed as the adversarial example of $x = \arg\max_x q(z|x)$.

Lemma 5.2 below is a general form of the Lemma 5.1, where we only consider strategy 1 ($j = 1$). Similar to the proof of Thm. 5.1 to 5.4.

Lemma 5.2 (generative adversarial examples)**.** *Using the 3 strategies defined above, the generative adversarial examples can be unified as*

$$x_{adv} = x' + \boldsymbol{W}\Delta z \quad \textit{and} \quad x' \sim \mathcal{D}'_j = \mathcal{N}(\boldsymbol{\mu}_*, \boldsymbol{U}_*\boldsymbol{\Lambda}^{(j)}\boldsymbol{U}_*^T), \quad j = 1, 2, 3,$$

where

$$\boldsymbol{\Lambda}^{(1)} = \begin{bmatrix} (\boldsymbol{\Lambda_q} - \sigma^2 I)^2 \boldsymbol{\Lambda_q}^{-1} & 0 \\ 0 & 0 \end{bmatrix} \quad \boldsymbol{\Lambda}^{(3)} = \begin{bmatrix} \boldsymbol{\Lambda_q} & 0 \\ 0 & \sigma^2 I \end{bmatrix}$$

$$\boldsymbol{\Lambda}^{(2)} = \begin{bmatrix} (\boldsymbol{\Lambda_q} - \sigma^2 I)^2 \boldsymbol{\Lambda_q}^{-1} + (\boldsymbol{\Lambda_q} - \sigma^2 I)\boldsymbol{\Lambda_q}^{-1}\sigma^2 + \sigma^2 I & 0 \\ 0 & \sigma^2 I \end{bmatrix}.$$

If the data lie in a q dimensional subspace, i.e. the covariance matrix $\boldsymbol{\Sigma}_*$ is rank q, we have $\boldsymbol{\Lambda}^{(1)} = \boldsymbol{\Lambda}^{(2)} = \boldsymbol{\Lambda}^{(3)} = \boldsymbol{\Lambda}_*$. Then $\mathcal{D}' = \mathcal{D}$.

Proof:

Using P-PCA generative model, $x \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{W}\boldsymbol{W}^T + \sigma^2 I)$, $x|z \sim \mathcal{N}(\boldsymbol{W}z + \boldsymbol{\mu}, \sigma^2 I)$ and $z|x \sim \mathcal{N}(\boldsymbol{P}^{-1}\boldsymbol{W}^T(x - \boldsymbol{\mu}), \sigma^2\boldsymbol{P}^{-1})$ where $\boldsymbol{P} = \boldsymbol{W}^T\boldsymbol{W} + \sigma^2 I$. The maximum likelihood estimator of $\boldsymbol{W}$ and σ^2 are

$$\boldsymbol{W}_{\text{ML}} = \boldsymbol{U}_q(\boldsymbol{\Lambda}_q - \sigma^2 I)^{1/2} \quad \text{and} \quad \sigma^2_{\text{ML}} = \frac{1}{d-q}\sum_{i=q+1}^{d} \lambda_i.$$

Strategy 1 Sample $x \sim \mathcal{D}$, encode $z = \arg\max q(z|x) = \boldsymbol{P}^{-1}\boldsymbol{W}^T(x - \boldsymbol{\mu}_*)$, add a perturbation Δz, and finally, decode $x_{adv} = \arg\max p(x|z + \Delta z) = \boldsymbol{W}(z + \Delta z) + \boldsymbol{\mu}_*$.

Then

$$\begin{aligned} x_{adv} =& \boldsymbol{W}(\mathbf{p}\boldsymbol{W}^T(x-\boldsymbol{\mu}_*)+\Delta z)+\boldsymbol{\mu}_* \\ =& \boldsymbol{W}\mathbf{p}\boldsymbol{W}^T(x-\boldsymbol{\mu}_*)+\boldsymbol{\mu}_*+\boldsymbol{W}\Delta z \\ =& x'+\boldsymbol{W}\Delta z. \end{aligned}$$

Since $x \sim (\boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$, we have $x - \boldsymbol{\mu}_* \sim (0, \boldsymbol{\Sigma}_*)$, Then

$$x' \sim \mathcal{N}(\boldsymbol{\mu}_*, \boldsymbol{W}\mathbf{p}\boldsymbol{W}^T\boldsymbol{\Sigma}_*(\boldsymbol{W}\mathbf{p}\boldsymbol{W}^T)^T),$$

With

$$\begin{aligned} & \boldsymbol{W}\mathbf{p}\boldsymbol{W}^T\boldsymbol{\Sigma}_*(\boldsymbol{W}\mathbf{p}\boldsymbol{W}^T)^T \\ =& \boldsymbol{U}_* \begin{bmatrix} \boldsymbol{\Lambda}_{\boldsymbol{q}}-\sigma^2 & 0 \\ 0 & 0 \end{bmatrix}^{1/2} \begin{bmatrix} \boldsymbol{\Lambda}_{\boldsymbol{q}} & 0 \\ 0 & \sigma^2 I \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{\Lambda}_{\boldsymbol{q}}-\sigma^2 & 0 \\ 0 & 0 \end{bmatrix}^{1/2} \boldsymbol{\Lambda}_* \\ & \begin{bmatrix} \boldsymbol{\Lambda}_{\boldsymbol{q}}-\sigma^2 & 0 \\ 0 & 0 \end{bmatrix}^{1/2} \begin{bmatrix} \boldsymbol{\Lambda}_{\boldsymbol{q}} & 0 \\ 0 & \sigma^2 I \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{\Lambda}_{\boldsymbol{q}}-\sigma^2 & 0 \\ 0 & 0 \end{bmatrix}^{1/2} \boldsymbol{U}_*^T \\ =& \boldsymbol{U}_* \begin{bmatrix} (\boldsymbol{\Lambda}_{\boldsymbol{q}}-\sigma^2 I)^2\boldsymbol{\Lambda}_{\boldsymbol{q}}^{-1} & 0 \\ 0 & 0 \end{bmatrix} \boldsymbol{U}_*^T \\ =& \boldsymbol{U}_*\boldsymbol{\Lambda}^{(j)}\boldsymbol{U}_*^T, \quad j=1. \end{aligned}$$

Strategy 2 Sample $x \sim \mathcal{D}$, then sample $z \sim q(z|x)$, add a perturbation Δz, and finally, sample $x_{adv} \sim p(x|z+\Delta z)$. Then

$$z \sim \mathcal{N}(0, \mathbf{p}\boldsymbol{W}^T\boldsymbol{\Sigma}_*(\mathbf{p}\boldsymbol{W}^T)^T + \sigma^2\mathbf{p})$$

and

$$x_{adv} \sim \mathcal{N}(\boldsymbol{\mu}_* + \boldsymbol{W}\Delta z, \boldsymbol{W}\mathbf{p}\boldsymbol{W}^T\boldsymbol{\Sigma}_*(\mathbf{p}\boldsymbol{W}^T)^T\boldsymbol{W}^T + \boldsymbol{W}\sigma^2\mathbf{p}\boldsymbol{W}^T + \sigma^2 I),$$

$$x_{adv} = x' + \boldsymbol{W}\Delta z,$$

With

$$\begin{aligned}
&\boldsymbol{W}\mathbf{p}\boldsymbol{W}^T\boldsymbol{\Sigma}_*(\mathbf{p}\boldsymbol{W}^T)^T\boldsymbol{W}^T + \boldsymbol{W}\sigma^2\mathbf{p}\boldsymbol{W}^T + \sigma^2 I \\
=&\boldsymbol{U}_* \begin{bmatrix} (\boldsymbol{\Lambda_q} - \sigma^2 I)^2\boldsymbol{\Lambda_q}^{-1} & 0 \\ 0 & 0 \end{bmatrix} \boldsymbol{U}_*^T + \boldsymbol{U}_* \begin{bmatrix} (\boldsymbol{\Lambda_q} - \sigma^2 I)\boldsymbol{\Lambda_q}^{-1}\sigma^2 & 0 \\ 0 & 0 \end{bmatrix} \boldsymbol{U}_*^T + \sigma^2 I \\
=&\boldsymbol{U}_* \begin{bmatrix} (\boldsymbol{\Lambda_q} - \sigma^2 I)^2\boldsymbol{\Lambda_q}^{-1} + (\boldsymbol{\Lambda_q} - \sigma^2 I)\boldsymbol{\Lambda_q}^{-1}\sigma^2 + \sigma^2 I & 0 \\ 0 & \sigma^2 I \end{bmatrix} \boldsymbol{U}_*^T \\
=&\boldsymbol{U}_*\boldsymbol{\Lambda}^{(j)}\boldsymbol{U}_*^T, \quad j = 2.
\end{aligned}$$

Strategy 3 Sample $z \sim \mathcal{N}(0, I)$, add a perturbation Δz, and then sample $x_{adv} \sim p(x|z + \Delta z)$. In this strategy, x_{adv} can be viewed as the adversarial example of $x = \arg\max_x q(z|x)$.

$$x_{adv} \sim \mathcal{N}(\boldsymbol{\mu}_* + \boldsymbol{W}\Delta z, \boldsymbol{W}\boldsymbol{W}^T + \sigma^2 I),$$

With

$$\begin{aligned}
&\boldsymbol{W}\boldsymbol{W}^T + \sigma^2 I \\
=&\boldsymbol{U}_* \begin{bmatrix} \boldsymbol{\Lambda_q} & 0 \\ 0 & \sigma^2 I \end{bmatrix} \boldsymbol{U}_*^T \\
=&\boldsymbol{U}_*\boldsymbol{\Lambda}^{(j)}\boldsymbol{U}_*^T, \quad j = 3.
\end{aligned}$$

In these 3 strageties, the adversarial examples can be summerized as

$$x_{adv} = x' + \boldsymbol{W}\Delta z \quad \text{and} \quad x' \sim \mathcal{D}'_j, \quad j = 1, 2, 3,$$

where $j = 1, 2, 3$ corresponding to strategy 1,2 and 3.

If the data lie in a low dimensional space, i.e. the covariance matrix $\boldsymbol{\Sigma}_*$ is rank q. Then the maximum likelihood of $\sigma^2_{ML} = \sum_{i=q+1}^{d} \lambda_i/(d-q) = 0$. Then

$$\boldsymbol{\Lambda}^{(1)} = \boldsymbol{\Lambda}^{(2)} = \boldsymbol{\Lambda}^{(3)} = \begin{bmatrix} \boldsymbol{\Lambda_q} & 0 \\ 0 & 0 \end{bmatrix} = \boldsymbol{\Lambda}_*.$$

□

5.7.2 Proof of Thm. 5.1

Before we prove Thm. 5.1, we need to prove the following lemma first.

Lemma 5.3 (optimal perturbation)**.** *Given* $\boldsymbol{\Theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$*, the optimal solution of the inner max problem in equation 5.4.3 is*

$$\Delta z^* = \boldsymbol{W}^T(L\boldsymbol{\Sigma} - \boldsymbol{W}\boldsymbol{W}^T)^{-1}(x' - \boldsymbol{\mu}),$$

where L *is the lagrange multiplier satisfying* $\|\Delta z^*\| = \varepsilon$*.*

Proof: Consider problem

$$\max_{\|\Delta z\| \leq \varepsilon} \ell(x' + \boldsymbol{W}\Delta z, \boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

The Lagrangian function is

$$\begin{aligned}
&\ell(x' + \boldsymbol{W}\Delta z, \boldsymbol{\mu}, \boldsymbol{\Sigma}) - \frac{L}{2}(\|\Delta z\|^2 - \varepsilon^2) \\
=&\frac{d}{2}\log(2\pi) + \frac{1}{2}\log|\boldsymbol{\sigma}| + \frac{1}{2}(x' - \boldsymbol{\mu} + \boldsymbol{W}\Delta z)^T\boldsymbol{\sigma}^{-1}(x' - \boldsymbol{\mu} + \boldsymbol{W}\Delta z) \\
&- \frac{L}{2}(\|\Delta z\|^2 - \varepsilon^2).
\end{aligned}$$

Notice that this quadratic objective function is concave when L is larger than the largest eigenvalue of $\boldsymbol{W}^T\boldsymbol{\sigma}^{-1}\boldsymbol{W}$. Calculate the partial derivative with respect to Δz and set it to be zero, we have

$$\begin{aligned}
&\boldsymbol{W}^T\boldsymbol{\sigma}^{-1}(x' - \boldsymbol{\mu} + \boldsymbol{W}\Delta z^*) - L\Delta z^* = 0 \\
\Leftrightarrow&(L - \boldsymbol{W}^T\boldsymbol{\sigma}^{-1}\boldsymbol{W})\Delta z^* = \boldsymbol{W}^T\boldsymbol{\sigma}^{-1}(x' - \boldsymbol{\mu}) \\
\Leftrightarrow&\Delta z^* = (L - \boldsymbol{W}^T\boldsymbol{\sigma}^{-1}\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{\sigma}^{-1}(x' - \boldsymbol{\mu}) \\
\Leftrightarrow&\Delta z^* = \boldsymbol{W}^T(L\boldsymbol{\sigma} - \boldsymbol{W}\boldsymbol{W}^T)^{-1}(x' - \boldsymbol{\mu}).
\end{aligned}$$

The last equation comes from the Woodbury matrix inversion Lemma. We can obtain L by solving the equation $\|\Delta z^*\| = \varepsilon$. □

Now we move to the proof of Thm. 5.1. To derive the expression of excess risk, we decompose it into two parts

$$\begin{aligned}&\mathcal{L}_{gat}(\boldsymbol{\Theta}_*;\mathcal{D}'_j)-\mathcal{L}(\boldsymbol{\Theta}_*;\mathcal{D})\\=&\underbrace{\mathcal{L}_{gat}(\boldsymbol{\Theta}_*;\mathcal{D}'_j)-\mathcal{L}(\boldsymbol{\Theta}_*;\mathcal{D}'_j)}_{perturbation}+\underbrace{\mathcal{L}(\boldsymbol{\Theta}_*;\mathcal{D}'_j)-\mathcal{L}(\boldsymbol{\Theta}_*;\mathcal{D})}_{change\ of\ distribution}.\end{aligned}\tag{5.7.1}$$

Since

$$x'\sim\mathcal{D}_j=\mathcal{N}(\boldsymbol{\mu}_*,\boldsymbol{\Sigma_j})=\mathcal{N}(\boldsymbol{\mu}_*,\boldsymbol{U}_*\boldsymbol{\Lambda}^{(j)}\boldsymbol{U}_*^T).$$

We denote v as

$$v=x'-\boldsymbol{\mu}_*\sim\mathcal{N}(0,\boldsymbol{U}_*\boldsymbol{\Lambda}^{(j)}\boldsymbol{U}_*^T).$$

Besides, we have

$$\boldsymbol{W}\boldsymbol{W}^T=\boldsymbol{U_q}(\boldsymbol{\Lambda_q}-\sigma^2I)\boldsymbol{U_q}^T=\boldsymbol{U}_*\begin{bmatrix}\boldsymbol{\Lambda_q}-\sigma^2I & 0\\ 0 & 0\end{bmatrix}\boldsymbol{U}_*^T.$$

Then, the excess risk caused by perturbation is

$$\begin{aligned}&2(\mathcal{L}_{gat}(\boldsymbol{\Theta}_*,\mathcal{D}'_j)-\mathcal{L}(\boldsymbol{\Theta}_*,\mathcal{D}'_j))\\=&\mathbb{E}(v+\boldsymbol{W}\boldsymbol{W}^T(L\boldsymbol{\Sigma}_*-\boldsymbol{W}\boldsymbol{W}^T)^{-1}v)^T\boldsymbol{\Sigma}_*^{-1}(v+\boldsymbol{W}\boldsymbol{W}^T(L\boldsymbol{\Sigma}_*-\boldsymbol{W}\boldsymbol{W}^T)^{-1}v)\\&-\mathbb{E}v^T\boldsymbol{\Sigma}_*^{-1}v\\=&Tr\big[(I+\boldsymbol{W}\boldsymbol{W}^T(L\boldsymbol{\Sigma}_*-\boldsymbol{W}\boldsymbol{W}^T)^{-1})^T\boldsymbol{\Sigma}_*^{-1}(I+\boldsymbol{W}\boldsymbol{W}^T(L\boldsymbol{\Sigma}_*-\boldsymbol{W}\boldsymbol{W}^T)^{-1})\mathbb{E}vv^T\big]\\&-Tr\big[\boldsymbol{\Sigma}_*^{-1}\mathbb{E}vv^T\big]\\=&Tr\big[\boldsymbol{U}_*\begin{bmatrix}[I+(\boldsymbol{\Lambda_q}-\sigma^2I)((L-1)\boldsymbol{\Lambda_q}+\sigma^2I)^{-1}]^2 & 0\\ 0 & I\end{bmatrix}\boldsymbol{\Lambda}_*^{-1}\boldsymbol{\Lambda}^{(j)}\boldsymbol{U}_*^T\big]-Tr\big[\boldsymbol{\Lambda}_*^{-1}\boldsymbol{\Lambda}^{(j)}\big]\\=&Tr\big[\begin{bmatrix}[I+(\boldsymbol{\Lambda_q}-\sigma^2I)((L-1)\boldsymbol{\Lambda_q}+\sigma^2I)^{-1}]^2 & 0\\ 0 & I\end{bmatrix}\boldsymbol{\Lambda}_*^{-1}\boldsymbol{\Lambda}^{(j)}\big]-Tr\big[\boldsymbol{\Lambda}_*^{-1}\boldsymbol{\Lambda}^{(j)}\big]\\=&\sum_{i=1}^{q}\Big[(1+\frac{\lambda_i-\sigma^2}{(L-1)\lambda_i+\sigma^2})^2-1\Big]\frac{\lambda_i^{(j)}}{\lambda_i},\quad j=1,2,3.\end{aligned}$$

The excess risk caused by changed of distribution is

$$
\begin{aligned}
&2(\mathcal{L}(\boldsymbol{\Theta}_*, \mathcal{D}'_j) - \mathcal{L}(\boldsymbol{\Theta}_*, \mathcal{D})) \\
=&\log|\boldsymbol{\Sigma_j}| - \log|\boldsymbol{\Sigma}_*| + \mathbb{E}_{x'}(x' - \boldsymbol{\mu}_*)^T \boldsymbol{\Sigma}_*^{-1}(x' - \boldsymbol{\mu}_*) \\
&- \mathbb{E}_x(x - \boldsymbol{\mu}_*)^T \boldsymbol{\Sigma}_*^{-1}(x - \boldsymbol{\mu}_*) \\
=&\log|\boldsymbol{\Sigma_j}| - \log|\boldsymbol{\Sigma}_*| + Tr(\boldsymbol{\Sigma}_*^{-1}\mathbb{E}_{x'}(x' - \boldsymbol{\mu}_*)(x' - \boldsymbol{\mu}_*)^T) \\
&- Tr(\boldsymbol{\Sigma}_*^{-1}\mathbb{E}_x(x - \boldsymbol{\mu}_*)(x - \boldsymbol{\mu}_*)^T) \\
=&\log\Big[\frac{\prod_{i=1}^d \lambda_i^{(j)}}{\prod_{i=1}^d \lambda_i}\Big] + Tr(\boldsymbol{\Lambda}_*^{-1}\boldsymbol{\Lambda}^{(j)}) - Tr(\boldsymbol{\Lambda}_*^{-1}\boldsymbol{\Lambda}_*) \\
=&\log\Big[\frac{\prod_{i=1}^d \lambda_i^{(j)}}{\prod_{i=1}^d \lambda_i}\Big] + \big(\sum_{i=1}^d \frac{\lambda_i^{(j)}}{\lambda_i} - d\big).
\end{aligned}
$$

If we further assume that the data lie in a q dimension manifold, by Lemma 5.2, we have $\sigma^2 = 0$. $\lambda_i^{(j)} = \lambda_i$ and $\mathcal{D}'_j = \mathcal{D}$. Hence the excess risk caused by changed of distribution

$$\mathcal{L}(\boldsymbol{\Theta}_*, \mathcal{D}'_j) - \mathcal{L}(\boldsymbol{\Theta}_*, \mathcal{D}) = 0.$$

The excess risk caused by perturbation is

$$
\begin{aligned}
&2(\mathcal{L}_{gat}(\boldsymbol{\Theta}_*, \mathcal{D}'_j) - \mathcal{L}(\boldsymbol{\Theta}_*, \mathcal{D}'_j)) \\
=&\sum_{i=1}^q \Big[(1 + \frac{\lambda_i - \sigma^2}{(L-1)\lambda_i + \sigma^2})^2 - 1\Big]\frac{\lambda_i^{(j)}}{\lambda_i} \\
=&\sum_{i=1}^q \Big[(1 + \frac{1}{(L-1)})^2 - 1\Big] \\
=&\Theta(qL^{-2}).
\end{aligned}
$$

□

5.7.3 Proof of Thm. 5.2

Proof: When $q = d$, the inner maximization problem is

$$\max_{\|\Delta x\| \leq \varepsilon} \ell(x + \Delta x, \boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

The Lagrangian function is

$$\begin{aligned}
&\ell(x + \Delta x, \boldsymbol{\mu}, \boldsymbol{\Sigma}) - \frac{L}{2}(\|\Delta x\|^2 - \varepsilon^2) \\
=&\frac{d}{2}\log(2\pi) + \frac{1}{2}\log|\boldsymbol{\sigma}| + \frac{1}{2}(x - \boldsymbol{\mu} + \Delta x)^T \boldsymbol{\sigma}^{-1}(x - \boldsymbol{\mu} + \Delta x) - \frac{L}{2}(\|\Delta x\|^2 - \varepsilon^2).
\end{aligned}$$

Notice that this quadratic objective function is concave when L is larger than the largest eigenvalue of $\boldsymbol{\sigma}^{-1}$. Calculate the partial derivative with respect to Δx and set it to be zero, we have

$$\begin{aligned}
&\boldsymbol{\sigma}^{-1}(x - \boldsymbol{\mu} + \Delta x^*) - L\Delta x^* = 0 \\
\Leftrightarrow &\Delta x^* = (L\boldsymbol{\sigma} - I)^{-1}(x - \boldsymbol{\mu}).
\end{aligned}$$

The excess risk is

$$\begin{aligned}
&2(\mathcal{L}_r(\boldsymbol{\Theta}_*, \mathcal{D}) - \mathcal{L}(\boldsymbol{\Theta}_*, \mathcal{D})) \\
=&\mathbb{E}(v + (L\boldsymbol{\Sigma}_* - I)^{-1}v)^T \boldsymbol{\Sigma}_*^{-1}(v + (L\boldsymbol{\Sigma}_* - I)^{-1}v) - \mathbb{E}v^T\boldsymbol{\Sigma}_*^{-1}v \\
=&Tr\left[(I + (L\boldsymbol{\Sigma}_* - I)^{-1})^T \boldsymbol{\Sigma}_*^{-1}(I + (L\boldsymbol{\Sigma}_* - I)^{-1})\mathbb{E}vv^T\right] - Tr\left[\boldsymbol{\Sigma}_*^{-1}\mathbb{E}vv^T\right] \\
=&\sum_{i=1}^{d}[(1 + \frac{1}{L\lambda_i - 1})^2 - 1].
\end{aligned}$$

On the one hand,

$$\begin{aligned}
&\sum_{i=1}^{d}[(1 + \frac{1}{L\lambda_i - 1})^2 - 1] \\
\geq&[(1 + \frac{1}{L\lambda_{min} - 1})^2 - 1] \\
\geq&\Omega((L\lambda_{min})^{-2}).
\end{aligned} \tag{5.7.2}$$

On the other hand,

$$\begin{aligned}&\sum_{i=1}^{d}[(1+\frac{1}{L\lambda_i-1})^2-1]\\ \leq& d[(1+\frac{1}{L\lambda_{min}-1})^2-1]\\ \leq& \mathcal{O}(d(L\lambda_{min})^{-2}).\end{aligned} \tag{5.7.3}$$

When $q \leq d$, $\max_{\|\Delta x\|\leq\varepsilon} \ell(x+\boldsymbol{U_q}\Delta x, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \max_{\|\Delta z\|\leq\varepsilon} \ell(z+\Delta z, \boldsymbol{\mu}, \boldsymbol{\Sigma}_q)$, Thm. 5.2 is obtain by replacing λ_{min} with λ_q in Eq. (5.7.2) and (5.7.3).

5.7.4 Proof of Thm. 5.3 and Thm. 5.4

We first state the general form of Thm. 5.3.

Theorem 5.5 (Optimal solution of generative adversarial training)**.** *The optimal solution of the modified problem in Eq. (5.4.5) is*

$$\boldsymbol{\mu}_{gat} = \boldsymbol{\mu}_* \quad \textit{and} \quad \boldsymbol{\Sigma}_{gat} = \boldsymbol{U}_*\boldsymbol{\Lambda}^{gat}\boldsymbol{U}_*^T,$$

where

$$\lambda_i^{gat} = \tfrac{1}{4}\Big[2\lambda_i^{(j)} + \tfrac{4(\lambda_i-\sigma^2)}{L} + 2\lambda_i^{(j)}\sqrt{1+\tfrac{4(\lambda_i-\sigma^2)}{\lambda_i^{(j)}L}}\Big] \textit{ for } i=1\leq q \textit{ and}$$
$$\lambda_i^{gat} = \lambda_i^{(j)} \textit{ for } i > q.$$

$j = 1, 2, 3$ *corresponding to strategies 1,2 and 3.*

By Lemma 5.3, the optimal perturbation Δz^* is a matrix M times $x-\boldsymbol{\mu}$. Consider the problem

$$\min_{\boldsymbol{\mu},\boldsymbol{\Lambda}} \max_{\mathbb{E}_{x'}\|M(x'-\boldsymbol{\mu})\|=\varepsilon} \mathbb{E}_{x'\sim\mathcal{D}_j'}\ell(x'+\boldsymbol{W}M(x'-\boldsymbol{\mu}), \boldsymbol{\mu}, \boldsymbol{\Sigma}), \quad j=1,2,3. \tag{5.7.4}$$

Lemma 5.4 (optimal perturbation of M)**.** *Given* $\boldsymbol{\Theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$ *the optimal solution of*

the inner max problem of 5.7.4 is

$$M^* = \boldsymbol{W}^T(L\boldsymbol{\Sigma} - \boldsymbol{W}\boldsymbol{W}^T)^{-1}.$$

Proof: Consider the problem

$$\max_{\mathbb{E}\|M(x'-\boldsymbol{\mu})\|=\varepsilon} \mathbb{E}\ell(x' + \boldsymbol{W}M(x'-\boldsymbol{\mu}), \boldsymbol{\mu}, \boldsymbol{\sigma})).$$

The lagrangian function is

$$\mathbb{E}\big[\ell(x' + \boldsymbol{W}M(x'-\boldsymbol{\mu}), \boldsymbol{\mu}, \boldsymbol{\Sigma}) - \frac{L}{2}(\|M(x'-\boldsymbol{\mu})\|^2 - \varepsilon^2)\big].$$

Let $x' - \boldsymbol{\mu} = v$, Take the gradient with respect to M and set it to be zero, we have

$$\begin{aligned}
&\frac{\partial}{\partial M}\mathbb{E}\big[\ell(x' + \boldsymbol{W}M(x'-\boldsymbol{\mu}), \boldsymbol{\mu}, \boldsymbol{\Sigma}) - \frac{L}{2}(\|M(x'-\boldsymbol{\mu})\|^2 - \varepsilon^2)\big] \\
=&\nabla_M \mathbb{E}\big[v^T M\boldsymbol{W}^T\boldsymbol{\sigma}^{-1}v + \frac{1}{2}v^T M\boldsymbol{W}^T\boldsymbol{\sigma}^{-1}\boldsymbol{W}Mv - Lv^TMMv/2\big] \\
=&\big[\boldsymbol{W}^T\boldsymbol{\sigma}^{-1} + \boldsymbol{W}^T\boldsymbol{\sigma}^{-1}\boldsymbol{W}M - LM\big]\mathbb{E}[vv^T] \\
=&0.
\end{aligned}$$

Then we have

$$\begin{aligned}
M^* =&(L - \boldsymbol{W}^T\boldsymbol{\sigma}^{-1}\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{\sigma}^{-1} \\
=&\boldsymbol{W}^T(L\boldsymbol{\Sigma} - \boldsymbol{W}\boldsymbol{W}^T)^{-1}.
\end{aligned}$$

The last equality is due to the Woodbury matrix inversion Lemma. □

To solve the problem 5.7.4, we need to introduce Danskin's Theorem.

Theorem 5.6 (Danskin's Theorem)**.** *Suppose $\phi(x, z) : \mathcal{X} \times \mathcal{Z} \to \mathbb{R}$ is a continuous function of two arguments, where $\mathcal{Z} \subset \mathbb{R}^m$ is compact. Define $f(x) = \max_{z\in\mathcal{Z}} \phi(x, z)$. Then, if for every $z \in \mathcal{Z}$, $\phi(x, z)$ is convex and differentiable in x, and $\partial\phi/\partial x$ is continuous:*

The subdifferential of $f(x)$ is given by

$$\partial f(x) = conv\{\frac{\partial \phi(x,z)}{\partial x}, z \in \mathcal{Z}_0(x)\},$$

where $conv(\cdot)$ is the convex hull, and $\mathcal{Z}_0(x)$ is

$$\mathcal{Z}_0(x) = \{\bar{z} : \phi(x,\bar{z}) = \max \phi(x,z)\}.$$

If the outer minimization problem is convex and differentiable, we can use any maximizer for the inner maximization problem to find the saddle point. Now we move to the proof of Theorem 5.3. By Lemma 5.4, we have

$$\begin{aligned} M^* =& \boldsymbol{W}^T(L\boldsymbol{\Sigma} - \boldsymbol{W}\boldsymbol{W}^T)^{-1} \\ =& \begin{bmatrix} \boldsymbol{\Lambda_q} - \sigma^2 & 0 \\ 0 & 0 \end{bmatrix}^{1/2} \Big(L\boldsymbol{\Lambda} - \begin{bmatrix} \boldsymbol{\Lambda_q} - \sigma^2 & 0 \\ 0 & 0 \end{bmatrix} \Big)^{-1} \boldsymbol{U}_*^T. \end{aligned}$$

Which is a diagonal matrix $\boldsymbol{\Lambda_M}$ times $\boldsymbol{U}_*^T$. Let $T = \boldsymbol{\Lambda}^{-1}$, $m = \boldsymbol{\Lambda}^{-1}\boldsymbol{U}_*^T\boldsymbol{\mu}$ and $x'' = \boldsymbol{U}_*^T x'$. The optimization problem becomes

$$\begin{aligned} \min_{m,T} \max_{\boldsymbol{\Lambda_M}} \quad & \mathbb{E}_{x' \sim \mathcal{D}'_j} \ell(x' + \boldsymbol{W}\boldsymbol{\Lambda_M}\boldsymbol{U}_*^T(x' - \boldsymbol{\mu}), m, T) \\ \text{s.t.} \quad & \mathbb{E}_{x'} \|\boldsymbol{\Lambda_M}\boldsymbol{U}_*(x' - \boldsymbol{\mu})\|^2 = \varepsilon^2. \end{aligned} \tag{5.7.5}$$

Obviously, the inner constraint is compact (by Heine-Borel theorem), we only need to prove the convexity of the outer problem to use Danskin's Theorem. For any x' and $\boldsymbol{\Lambda_M}$,

$$\begin{aligned} & \ell(x' + \boldsymbol{W}\boldsymbol{\Lambda}_M^T\boldsymbol{U}_*(x' - \boldsymbol{\mu}), m, T) \\ =& \frac{d}{2}\log(2\pi) + \frac{1}{2}\log|\boldsymbol{\sigma}| + \frac{1}{2}(x' - \boldsymbol{\mu} + \boldsymbol{W}M(x' - \boldsymbol{\mu})^T\boldsymbol{\sigma}^{-1}(x' - \boldsymbol{\mu} + \boldsymbol{W}M(x' - \boldsymbol{\mu}). \end{aligned}$$

Let $u = \boldsymbol{U}_*^T(x' - \boldsymbol{\mu})$, and $A = (I + \begin{bmatrix} \boldsymbol{\Lambda_q} - \sigma^2 & 0 \\ 0 & 0 \end{bmatrix} \boldsymbol{\Lambda_M})^2$, consider the third term, we

have

$$\begin{aligned}&\frac{1}{2}\log|\boldsymbol{\sigma}|+\frac{1}{2}(x'-\boldsymbol{\mu}+\boldsymbol{W}M(x'-\boldsymbol{\mu})^T\boldsymbol{\sigma}^{-1}(x'-\boldsymbol{\mu}+\boldsymbol{W}M(x'-\boldsymbol{\mu})\\=&\frac{1}{2}u^TA^2Tu.\end{aligned}$$

By (Daskalakis et al., 2018), The hessian matrix is

$$H=\mathrm{Cov}_{z\sim\mathcal{N}(T^{-1}m,(AT)^{-1})}\left[\begin{pmatrix}\mathrm{vec}(-\frac{1}{2}Azz^T)\\ z\end{pmatrix},\begin{pmatrix}\mathrm{vec}(-\frac{1}{2}Azz^T)\\ z\end{pmatrix}\right]\succeq 0.$$

Therefore, the outer problem of Eq. (5.7.5) is a convex problem. By Lemma 5.4, a maximizer of the inner problem is

$$\boldsymbol{\Lambda}_{\boldsymbol{M}}^*=\begin{bmatrix}\boldsymbol{\Lambda_q}-\sigma^2 & 0\\ 0 & 0\end{bmatrix}^{1/2}\left(L\boldsymbol{\Lambda}-\begin{bmatrix}\boldsymbol{\Lambda_q}-\sigma^2 & 0\\ 0 & 0\end{bmatrix}\right)^{-1}.$$

Then

$$A=\left[I+\begin{bmatrix}\boldsymbol{\Lambda_q}-\sigma^2 & 0\\ 0 & 0\end{bmatrix}\left(L\boldsymbol{\Lambda}-\begin{bmatrix}\boldsymbol{\Lambda_q}-\sigma^2 & 0\\ 0 & 0\end{bmatrix}\right)^{-1}\right]^2.$$

Then the first order derivative (by (Daskalakis et al., 2018)) is

$$\nabla_{[T,m]^T}\ell=\begin{bmatrix}\frac{1}{2}A\boldsymbol{\Lambda}^{(j)}-\frac{1}{2}T^{-1}\\ AT^{-1}m-A\boldsymbol{U}_*^T\boldsymbol{\mu}_*\end{bmatrix}=0.$$

From the second equation, we directly have $\boldsymbol{\mu}_{gat}=\boldsymbol{\mu}_*$. From the first equation, for $i>q$, we have

$$(1+0)^2\lambda_i^{(j)}=\lambda_i^{gat}.$$

For $i\le q$, we have

$$(1+(\lambda_i-\sigma^2)/(L\lambda_i^{gat}-\lambda_i+\sigma^2))^2\lambda_i^{(j)}=\lambda_i^{gat}.$$

It equivalents to a second order equation of $\sqrt{\lambda_i^{gat}}$

$$\sqrt{\lambda_i^{gat}}^2 - \sqrt{\lambda_i^{(j)}}\sqrt{\lambda_i^{gat}} - \frac{\lambda_i - \sigma^2}{L} = 0.$$

Solving this equation, we obtained

$$\lambda_i^{gat} = \frac{1}{4}\left[2\lambda_i^{(j)} + \frac{4(\lambda_i - \sigma^2)}{L} + 2\lambda_i^{(j)}\sqrt{1 + \frac{4(\lambda_i - \sigma^2)}{\lambda_i^{(j)} L}}\right] \text{ for } i = 1 \leq q \text{ and}$$
$$\lambda_i^{gat} = \lambda_i^{(j)}, \text{ for } i > q.$$

Thm. 5.4 is obtained by replacing W with $\boldsymbol{U}_q$. □

5.8 Addtional Experiments

5.8.1 Training Settings

On MNIST, we use LeNet5 for the classifier and 2 layers MLP (with hidden size 256 and 784) for the encoder and decoder of conditional VAE. For standard training of the classifier, we use 30 epochs, batch size 128, learning rate 10^{-3}, and weight decay 5×10^{-4}. For the CVAE, we use 20 epochs, learning rate 10^{-3}, batch size 64, and latent size 10. For standard adversarial training and, we use $\varepsilon = 0.3$ for FGSM and PGD. in PGD, we use 40 steps for the inner part. Adversarial training start after 10 epochs standard training. For generative adversarial training, we use $\varepsilon = 1$ in the latent space with FGSM and PGD. We use 40 steps PGD for latent space adversarial training. Adversarial training start after 10 epoches standard training. In the attack part, we use the same ε as the adversarial training part.

On CIFAR-10, we use ResNet32 for the classifier and 4 layers CNN for the encoder and decoder of conditional VAE. For standard training of the classifier, we use 200 epochs, batch size 128. The learning rate schedule is 0.1 for the first 100 epochs, 0.01 for the following 50 epochs, and 0.001 for the last 50 epochs. The weight decay is 5×10^{-4}. For the CVAE, we use 200 epochs, learning rate 10^{-3}, batch size 64, and latent size 128.

For standard adversarial training, we use $\varepsilon = 8/255$ for FGSM and PGD. in PGD, we use 10 steps for the inner part. For generative adversarial training, we use $\varepsilon = 0.1$ in the latent space with FGSM and PGD. Since we see that the modeling power of VAE in CIFAR10 is not good enough. For each of the image, the encode variance is very small. When we add a small perturbation to the encode mean value, the output image are blured. Hence we only use a small $\varepsilon = 0.1$. In the attack part, we use $\varepsilon = 8/255$ for norm-based attacks and $\varepsilon = 0.1$ for generative attack on the test set.

CIFAR-100. The training hyperparameters are the same as that on CIFAR-10. In Table 5.3, we provide the experiments results on CIFAR-100. We ca get the same

conclusion as in the MNIST and CIFAR-10. Standard adversarial training cannot defense generative attack, and vice versa. Notice that the modeling power of condition-VAE is not good enough for CIFAR-100. Therefore, we use a more powerful StyleGAN for ImageNet.

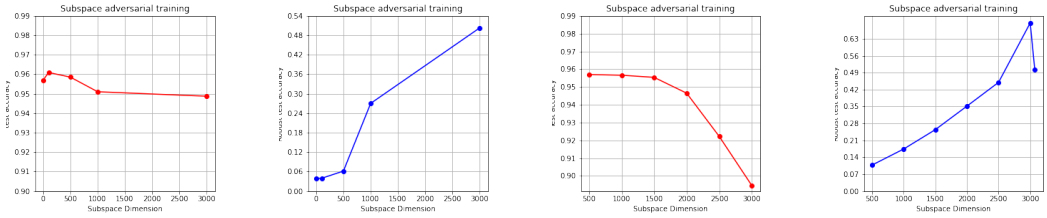
Table 5.3: Test accuracy of different defense algorithms (PGD-AT, GPGD-AT, and joint AT) against different attacks (regular attacks (FGSM, PGD) and generative attack (GFGSM, GPGD)) on CIFAR-100.

CIFAR-100	clean data	FGSM-Attack	PGD-Attack	VAE-FGSM-Attack	VAE-PGD-Attack
Std training	67.26%	9.13%	7.31%	1.67%	0.34%
PGD-AT	50.85%	31.49%	27.27%	6.41%	4.18%
VAE-PGD-AT	45.21%	3.45%	1.21%	12.45%	10.75%
joint-PGD-AT	48.34%	30.81%	25.55%	12.01%	10.12%

ImageNet. We adopt the setting of dual manifold adversarial training (DMAT) (Lin et al., 2020) in our experiments. DMAT is an algorithm using both on-manifold and off-manifold adversarial examples to train a neural network. The off-manifold adversarial examples are ℓ_∞ PGD adversarial examples. The on-manifold adversarial examples are crafted by a StyleGAN (Karras et al., 2019) trained on ImageNet. The difference between the algorithms we used and DMAT is the generative model, CVAE, and StyleGAN. StyleGAN is better on ImageNet because of the modeling power. In Table 5.2, we show the results of DMAT on ImageNet. We can see that, on-manifold adversarial training (GPGD-AT) is not able to defend PGD-Attack, with 0.03% robust test accuracy, and vice versa. Joint-AT can achieve comparable robust test accuracy to the single-trained models. Therefore, the results of DMAT support our analysis. Conversely, our theory gives an analysis of the performance of DMAT.

5.8.2 Eigenspace Adversarial Training

In Figure 5.6 (a) and (b), we give the experimental results of on-manifold adversarial training, i.e. adversarial training restricted in the subspace spanned by the top eigenvectors. The red lines are the standard test accuracy and the blue lines are the robust test accuracy against PGD-10 ℓ_2 attacks. (c) and (d) are the results of off-manifold



(a) (b) (c) (d)

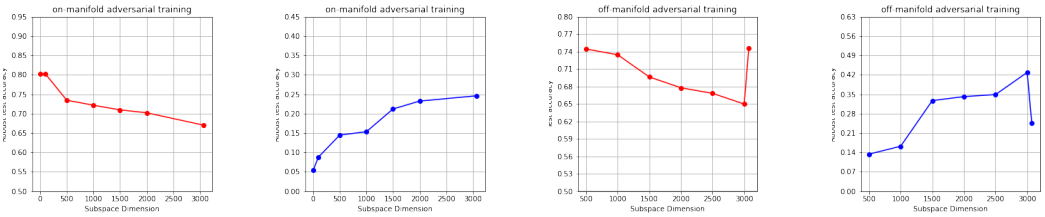
Figure 5.6: Standard and robust test accuracy of subspace adversarial training on CIFAR-10. The x-axis is the dimension of the restricted subspace of adversarial training. The red lines are the standard test accuracy and the blue lines are the robust test accuracy for (full space) PGD-10 ℓ_2 attacks. (a) and (b) are the results of on-manifold adversarial training, i.e. adversarial training restricted in the subspace spanned by the top 500, 1000, $\cdots$, 3000 eigenvectors; (c) and (d) are the results of off-manifold adversarial training.

adversarial training.

Bridging the Generalization-Robustness Trade-off. The range of the x-axis is from 0 to 3072. Notice that if the number of subspace dimension is 0, no perturbation is allowed. It equivalents to the standard training case. If the number of subspace dimension is 3072, the perturbation Δx can be chosen in the whole norm ball $\{\|\cdot\| \leq \varepsilon\}$. It equivalents to the regular adversarial training case. Therefore, subspace adversarial training is a middle situation between standard training and adversarial training.

On-manifold Adversarial Training Improves Generalization. In Figure 5.6 (a), the best test accuracy is 96.17%, achieved in subspace adversarial training restricted to the top 100 eigenvectors. In Figure 5.6 (c), we can see that off-manifold adversarial training cannot obtain a better test accuracy than standard training. It is because on-manifold adversarial examples will cause a small distributional shift, the negative effect of distributional shift is smaller than the positive effect of data augmentation. But for off-manifold adversarial examples, the negative effect of off-manifold adversarial examples is larger.

Eigenspace Adversarial Training Cannot Improves Robustness. In Figure 5.6, we show the experiments of Off-manifold adversarial training. We can see that eigenspace adversarial training cannot improves robustness.



(a) (b) (c) (d)

Figure 5.7: Standard and robust test accuracy of subspace adversarial training on CIFAR-100. The red lines are the standard test accuracy and the blue lines are the robust test accuracy for (full space) PGD-10 ℓ_2 attacks. (a) and (b) are the results of on-manifold adversarial training; (c) and (d) are the results of off-manifold adversarial training.

Eigenspace Adversarial Training on CIFAR-100. As it is shown in Figure 5.7, on-manifold adversarial training (top 100 dimensions) get the same test accuracy as standard training. Off-manifold training (bottom 3000 dimensions) get larger robust accuracy (42.89%) than standard adversarial training (24.60%). The results are similar to those in CIFAR-10.

5.8.3 Ablation Study of Adversarial Distribution Shift

We plot the adversarial distribution shift of all the classes in this section, see Figure 5.8 and 5.9. For all the 10 classes, We can see that adversarial training will amplify the small eigenvalues and generative adversarial training will amplify the large eigenvalues.

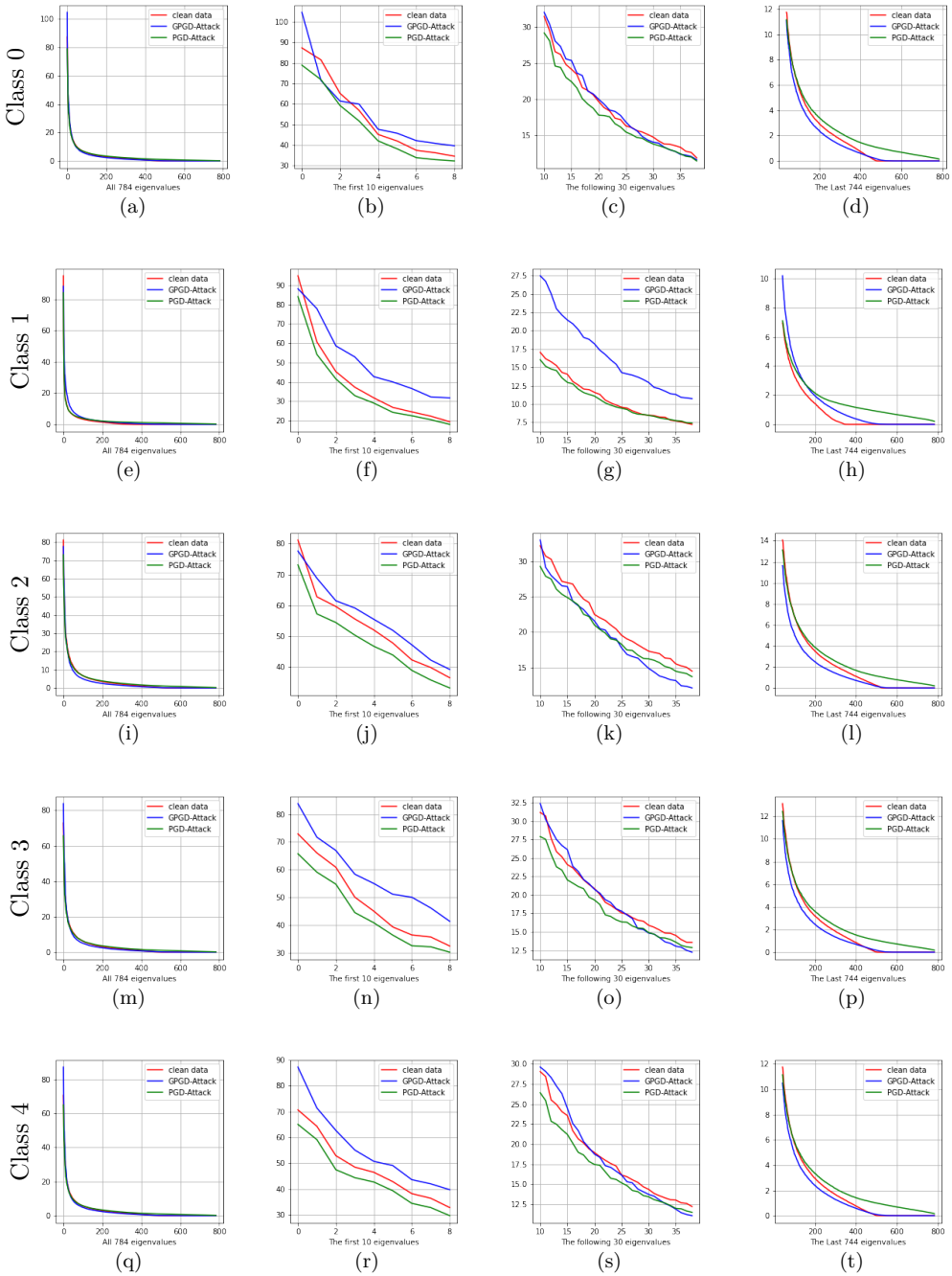


Figure 5.8: Adversarial distributional shift on MNIST for all the 10 classes (0-4).

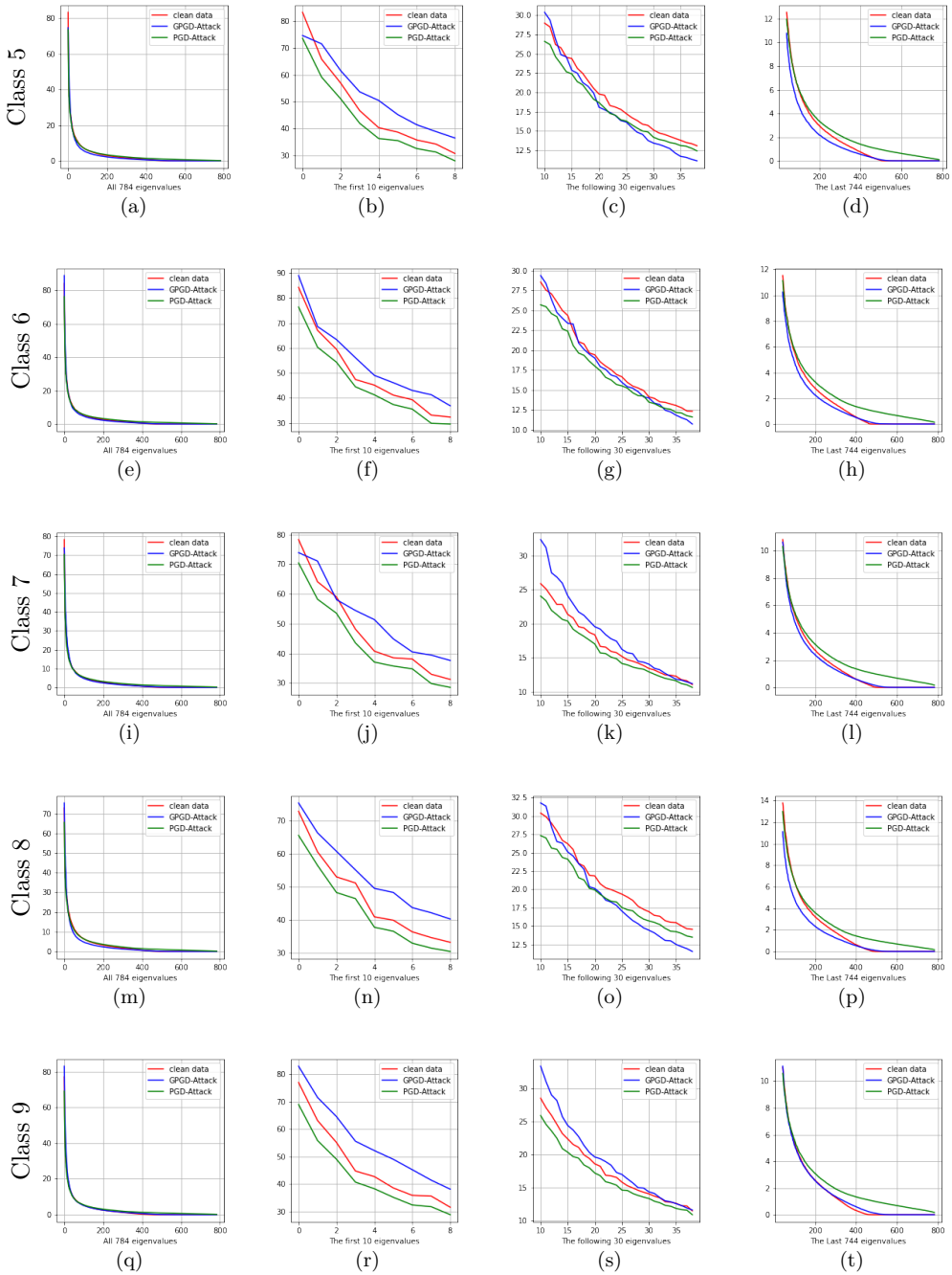


Figure 5.9: Adversarial distributional shift on MNIST for all the 10 classes (5-9).

Chapter 6

Conclusion

In this thesis, we study the poor adversarially robust generalization from the perspective of classical learning theory.

6.1 Summary of the Thesis

Firstly, we study the adversarial Rademacher complexity (ARC) of deep neural networks. We provide the first bound of adversarial Rademacher complexity of deep neural networks, which was an open problem. We provide a novel adversarial perturbation bound such that we are able to calculate the ARC. The bounds of ARC provide an interpretation of poor adversarial generalization. Secondly, we study the robust overfitting issue of adversarial training by using tools from uniform stability. We derive stability-based generalization bounds for stochastic gradient descent (SGD) on the general class of η-approximate smooth functions, which covers the adversarial loss. Our results suggest adversarial training can have nonvanishing generalization errors. Robust test accuracy decreases in ϵ when T is large, with a speed between $\Omega(\epsilon\sqrt{T})$ and $\mathcal{O}(\epsilon T)$. A natural question arises: can we eliminate the generalization error floor in adversarial training? Chapter 4 gives an affirmative answer. We employ a smoothing technique to smooth the adversarial loss function. Based on the smoothed loss function, we design a ME-SGD algorithm to eliminate the generalization error floor. Lastly, we

study the properties of on-manifold adversarial examples. Our analysis suggests that on-manifold adversarial examples are important, and we should pay more attention to on-manifold adversarial examples for training robust models.

Conclusion of the Thesis. Based on our research, we find that the following factors are highly related to the (i.i.d) adversarially robust generalization gap:

1. the product of the weight norms;
2. the non-smoothness of loss function induced by the adversarial attacks.

To overcome these issues, we introduce a variant of SGD called Moreau envelope-SGD. First of all, ME-SGD uses a regularization term that regularizes the weight to its moving average, which controls the weight norms and helps to mitigate the first issue. Secondly, ME-SGD smooths the loss function, which overcomes the second issue.

We find that the following factor is related to the (O.O.D) adversarially robust generalization:

1. on-manifold adversarial examples.

Adversarial training focuses on off-manifold adversarial examples and ignores on-manifold adversarial examples. We show that on-manifold and joint-manifold adversarial training can improve the O.O.D robust generalization of DNNs.

6.2 Open Problem and Future Works

There are some open problems raised in this thesis.

The Gap between the Upper and Lower Bounds for ARC. The gap between the upper and lower bounds is the dependence on the depth-l and the width-h, which can be summarized as the size of the DNNs. Based on the observation in practice and the related work of standard Rademacher complexity, we conjecture that the lower bound is closer to the truth. However, how to close the gap between the upper and lower bounds is an open mathematical problem.

Uniform Stability for Non-smooth Non-convex Functions. In Chapter 4, we have shown that the generalization floor can be eliminated in non-smooth weakly-convex cases. However, the loss functions are non-convex in many real applications. How to derive tight uniform stability-based generalization bound in non-convex cases is an open problem.

Better Approximation of Data Manifold. From the experiments in Chapter 5, we can see that the data manifold approximated by generative models (GAN, VAE) is not good enough. We conjecture that adversarial robustness can be benefited from on-manifold adversarial examples crafted using a better approximation of the data manifold.

Lower Bounds for Robust Representation Error. This thesis mainly focuses on robust generalization, with some results about robust optimization. We can see a trade-off between robust generalization and robust optimization. A natural question is: what is the optimal trade-off given a neural network model? In other words, what is the lower bound for robust representation error, and how to design better neural networks to reduce this lower bound? It requires further investigation.

Also, there are some big problems in the adversarial machine learning research field.

The Gap between Learning Theory and Practical Studies. This thesis focuses on classical learning theory. However, there is a big gap between classical learning theory and modern empirical deep learning research. How to derive more informative bounds and develop a better theory to explain deep learning are still open problems.

Empirical Research. In empirical adversarial machine learning research, the state-of-the-art models are about 65%, which is far from satisfactory. How to achieve the ultimate goal of adversarial machine learning research, which is to train a safe model with acceptable robust accuracy, is an open problem.

Bibliography

Zeyuan Allen-Zhu and Yuanzhi Li. Feature purification: How adversarial training performs robust deep learning. *arXiv preprint arXiv:2005.10190*, 2020.

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pp. 242–252. PMLR, 2019.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

Idan Attias, Aryeh Kontorovich, and Yishay Mansour. Improved generalization bounds for adversarially robust learning. 2021.

Pranjal Awasthi, Natalie Frank, and Mehryar Mohri. Adversarial learning guarantees for linear hypotheses and neural networks. In *International Conference on Machine Learning*, pp. 431–441. PMLR, 2020.

Peter Bartlett, Dylan J Foster, and Matus Telgarsky. Spectrally-normalized margin bounds for neural networks. *arXiv preprint arXiv:1706.08498*, 2017.

Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Raef Bassily, Vitaly Feldman, Cristóbal Guzmán, and Kunal Talwar. Stability of stochastic gradient descent on nonsmooth convex losses. *arXiv preprint arXiv:2006.06914*, 2020.

Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.

Olivier Bousquet and André Elisseeff. Stability and generalization. *The Journal of Machine Learning Research*, 2:499–526, 2002.

Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. In *International Conference on Learning Representations*, 2018.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. IEEE, 2017.

Yair Carmon, Aditi Raghunathan, Ludwig Schmidt, John C Duchi, and Percy S Liang. Unlabeled data improves adversarial robustness. In *Advances in Neural Information Processing Systems*, pp. 11190–11201, 2019.

Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 15–26, 2017.

Yuansi Chen, Chi Jin, and Bin Yu. Stability and convergence trade-off of iterative optimization algorithms. *arXiv preprint arXiv:1804.01619*, 2018.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pp. 1310–1320. PMLR, 2019.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pp. 2206–2216. PMLR, 2020.

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

Bin Dai, Yu Wang, John Aston, Gang Hua, and David Wipf. Hidden talents of the variational autoencoder. *arXiv preprint arXiv:1706.05148*, 2017.

Chen Dan, Yuting Wei, and Pradeep Ravikumar. Sharp statistical guaratees for adversarially robust gaussian classification. In *International Conference on Machine Learning*, pp. 2345–2355. PMLR, 2020.

Constantinos Daskalakis, Themis Gouleakis, Chistos Tzamos, and Manolis Zampetakis. Efficient statistics, in high dimensions, from truncated samples. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 639–649. IEEE, 2018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Guneet S Dhillon, Kamyar Azizzadenesheli, Zachary C Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Anima Anandkumar. Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*, 2018.

Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning*, pp. 1675–1685. PMLR, 2019.

Farzan Farnia and Asuman Ozdaglar. Train simultaneously, generalize better: Stability of gradient-based minimax learners. In *International Conference on Machine Learning*, pp. 3174–3185. PMLR, 2021.

Vitaly Feldman and Jan Vondrak. Generalization bounds for uniformly stable algorithms. *arXiv preprint arXiv:1812.09859*, 2018.

Vitaly Feldman and Jan Vondrak. High probability generalization bounds for uniformly stable algorithms with nearly optimal rate. In *Conference on Learning Theory*, pp. 1270–1279. PMLR, 2019.

Qingyi Gao and Xiao Wang. Theoretical investigation of generalization bounds for adversarial learning of deep neural networks. *Journal of Statistical Theory and Practice*, 15(2):1–28, 2021.

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

Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. In *Conference On Learning Theory*, pp. 297–299. PMLR, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014a.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014b.

Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv preprint arXiv:2010.03593*, 2020.

Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning*, pp. 1225–1234. PMLR, 2016.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Joong-Won Hwang, Youngwan Lee, Sungchan Oh, and Yuseok Bae. Adversarial training with stochastic weight average. In *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 814–818. IEEE, 2021.

Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Prior convictions: Black-box adversarial attacks with bandits and priors. *arXiv preprint arXiv:1807.07978*, 2018.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pp. 125–136, 2019.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Adel Javanmard, Mahdi Soltanolkotabi, and Hamed Hassani. Precise tradeoffs in adversarial training for linear regression. In *Conference on Learning Theory*, pp. 2034–2078. PMLR, 2020.

Vishaal Munusamy Kabilan, Brandon Morris, Hoang-Phuong Nguyen, and Anh Nguyen. Vectordefense: Vectorization as a defense to adversarial examples. In *Soft Computing for Biomedical Applications and Related Topics*, pp. 19–35. Springer, 2018.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4401–4410, 2019.

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

Marc Khoury and Dylan Hadfield-Menell. On the geometry of adversarial examples. *arXiv preprint arXiv:1811.00525*, 2018.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Jernej Kos, Ian Fischer, and Dawn Song. Adversarial examples for generative models. In *2018 IEEE Security and Privacy Workshops (SPW)*, pp. 36–42. IEEE, 2018.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 656–672. IEEE, 2019.

Michel Ledoux and Michel Talagrand. *Probability in Banach Spaces: isoperimetry and processes*. Springer Science & Business Media, 2013.

Yandong Li, Lijun Li, Liqiang Wang, Tong Zhang, and Boqing Gong. Nattack: Learning the distributions of adversarial examples for an improved black-box attack on deep neural networks. *arXiv preprint arXiv:1905.00441*, 2019.

Wei-An Lin, Chun Pong Lau, Alexander Levine, Rama Chellappa, and Soheil Feizi. Dual manifold adversarial robustness: Defense against lp and non-lp adversarial attacks. *arXiv preprint arXiv:2009.02470*, 2020.

Chen Liu, Mathieu Salzmann, Tao Lin, Ryota Tomioka, and Sabine Süsstrunk. On the loss landscape of adversarial training: Identifying challenges and how to overcome them. *arXiv preprint arXiv:2006.08403*, 2020.

Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E Houle, and James Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. *arXiv preprint arXiv:1801.02613*, 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.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. 2018.

Omar Montasser, Steve Hanneke, and Nathan Srebro. Vc classes are adversarially robustly learnable, but only improperly. In *Conference on Learning Theory*, pp. 2512–2530. PMLR, 2019.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.

Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609, 2009.

Arkadij Semenovič Nemirovskij and David Borisovich Yudin. Problem complexity and method efficiency in optimization. 1983.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. Norm-based capacity control in neural networks. In *Conference on Learning Theory*, pp. 1376–1401. PMLR, 2015.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nathan Srebro. Exploring generalization in deep learning. *arXiv preprint arXiv:1706.08947*, 2017a.

Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. *arXiv preprint arXiv:1707.09564*, 2017b.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*, pp. 372–387. IEEE, 2016.

Aaditya Prakash, Nick Moran, Solomon Garber, Antonella DiLillo, and James Storer. Deflecting adversarial attacks with pixel deflection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8571–8580, 2018.

Sylvestre-Alvise Rebuffi, Sven Gowal, Dan A Calian, Florian Stimberg, Olivia Wiles, and Timothy Mann. Fixing data augmentation to improve adversarial robustness. *arXiv preprint arXiv:2103.01946*, 2021.

Leslie Rice, Eric Wong, and Zico Kolter. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning*, pp. 8093–8104. PMLR, 2020.

R Tyrrell Rockafellar. Monotone operators and the proximal point algorithm. *SIAM j. control optim.*, 14(5):877–898, August 1976.

William H Rogers and Terry J Wagner. A finite sample distribution-free performance bound for local discrimination rules. *The Annals of Statistics*, pp. 506–514, 1978.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Aman Sinha, Hongseok Namkoong, and John Duchi. Certifiable distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2, 2017.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, pp. 3483–3491, 2015.

Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon. Constructing unrestricted adversarial examples with generative models. In *Advances in Neural Information Processing Systems*, pp. 8312–8323, 2018.

Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23 (5):828–841, 2019.

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.

Hossein Taheri, Ramtin Pedarsani, and Christos Thrampoulidis. Asymptotic behavior of adversarial training in binary classification. *arXiv preprint arXiv:2010.13275*, 2020.

Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61 (3):611–622, 1999.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. *arXiv preprint arXiv:2002.08347*, 2020.

Vladimir N Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. In *Measures of complexity*, pp. 11–30. Springer, 2015.

Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge University Press, 2019.

Puyu Wang, Yunwen Lei, Yiming Ying, and Ding-Xuan Zhou. Stability and generalization for markov chain stochastic gradient methods. *arXiv preprint arXiv:2209.08005*, 2022.

Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. In *ICML*, volume 1, pp. 2, 2019.

Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.

Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.

Jiancong Xiao, Yanbo Fan, Ruoyu Sun, Jue Wang, and Zhi-Quan Luo. Stability analysis and generalization bounds of adversarial training. In *Thirty-Sixth Conference on Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=78aj7sPX4s-.

Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*, 2017.

Yue Xing, Qifan Song, and Guang Cheng. On the generalization properties of adversarial training. In *International Conference on Artificial Intelligence and Statistics*, pp. 505–513. PMLR, 2021a.

Yue Xing, Qifan Song, and Guang Cheng. On the algorithmic stability of adversarial training. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021b. URL https://openreview.net/forum?id=xz80iPFIjvG.

Yue Xing, Ruizhi Zhang, and Guang Cheng. Adversarially robust estimate and risk analysis in linear regression. In *International Conference on Artificial Intelligence and Statistics*, pp. 514–522. PMLR, 2021c.

Zhenhuan Yang, Yunwen Lei, Siwei Lyu, and Yiming Ying. Stability and differential privacy of stochastic gradient descent for pairwise learning with non-smooth loss. In *International Conference on Artificial Intelligence and Statistics*, pp. 2026–2034. PMLR, 2021.

Dong Yin, Ramchandran Kannan, and Peter Bartlett. Rademacher complexity for ad-

versarially robust generalization. In *International Conference on Machine Learning*, pp. 7085–7094. PMLR, 2019.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pp. 7472–7482. PMLR, 2019.

Jiawei Zhang and Zhi-Quan Luo. A proximal alternating direction method of multiplier for linearly constrained nonconvex minimization. *SIAM Journal on Optimization*, 30 (3):2272–2302, 2020.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. Generating natural adversarial examples. *arXiv preprint arXiv:1710.11342*, 2017.

Copyright Authorization for Thesis/Dissertation

Author: Jiancong Xiao

Date: Feb. 11th 2023

Title: Understanding Adversarially Robust Generalization: A Learning Theory Perspective

I grant to the Chinese University of Hong Kong, Shenzhen to make my Thesis/Dissertation available in the University's Thesis and Dissertation Collection where my Thesis/Dissertation will be preserved both in print and electronic format. I understand that my Thesis/Dissertation will be available to the university members who are permitted to access the University's Thesis and Dissertation Collection, and I give my permission for the University to reproduce, distribute, display, and transmit my Thesis/Dissertation in order to make it available to university members, provided that any and all such acts are only for scholastic and academic purpose and with proper acknowledgement of authorship.

I understand that this permission constitutes a nonexclusive, perpetual, royalty-free license, and that I retain all other rights to the copyright in my Thesis/Dissertation, including the right to use it in other works such as articles and books.

I warrant that I am the sole author and owner of the copyright in my Thesis/Dissertation, and that I have full and sole authority to grant this permission. I also warrant that this Thesis/Dissertation does not infringe or violate any rights of others. I have obtained any third-party rights, if necessary.

Permission Granted by
Jiancong Xiao

Print Name

Jiancong Xiao

Signature