공학박사학위논문

Analyzing Loss Landscape of Deep Learning Models for Better Robustness and Generalization

손실함수 탐색을 통한 딥러닝 모델의 강건성과 일반화 향상

2023 년 2 월

서울대학교 대학원

산업공학과

김 호 기

Analyzing Loss Landscape of Deep Learning Models for Better Robustness and Generalization

손실함수 탐색을 통한 딥러닝 모델의 강건성과 일반화 향상

지도교수 이 재 욱

이 논문을 공학박사 학위논문으로 제출함

2022 년 12 월

서울대학교 대학원

산업공학과

김 호 기

김호기의 공학박사 학위논문을 인준함

2023 년 1 월

위 원 장	조 성 준	(인)
부위원장	이 재 욱	(인)
위 원	박 우 진	(인)
위 원	정 규 환	(인)
위 원	이 우 진	(인)

Abstract

Analyzing Loss Landscape of Deep Learning Models for Better Robustness and Generalization

Hoki Kim
Department of Industrial Engineering
The Graduate School
Seoul National University

Recent advances in deep learning have demonstrated significant performance improvements in various domains, such as computer vision and speech recognition, yielding numerous industrial applications. Compared to other machine learning models, deep learning models have a large number of parameters and this brings near zero training loss that was previously considered impossible. To train these over-parameterized models, we generally minimize the loss on training data, which we call empirical risk minimization (ERM). However, recent studies have demonstrated that these deep learning models trained by ERM may suffer from two major problems: *adversarial vulnerability* and *poor generalization*.

Adversarial vulnerability is an intriguing property of deep learning models that makes them susceptible to adversarial attacks that create malicious examples with slight modifications (Szegedy et al., 2013; Goodfellow et al., 2014). Prior studies have also confirmed that there exist the potential risks of deep learning models in

real-world applications (Papernot et al., 2017; Kurakin et al., 2016). Adversarial attacks entail severe hazards in real-world applications, e.g., causing autonomous vehicle accidents by manipulating decision-making or extracting private information by circumventing voice authorization. Thus, to prevent these malicious cases arisen from the existence of adversarial attacks, many researchers proposed various methods to enhance the robustness of deep learning models against adversarial attacks.

Poor generalization, another issue with current deep learning models, is a large discrepancy between training accuracy and test accuracy. In other words, existing methods can successfully minimize loss on train datasets, but this does not guarantee high performance on test datasets (Ishida et al., 2020; Foret et al., 2020). To achieve an ideal performance over various domains, improving the generalization of neural networks has been a core challenge in deep learning.

In this dissertation, focusing on the fact that both robustness and generalization are heavily related to the loss landscape, we aim to gain a deeper understanding of adversarial robustness and generalization performance of deep learning models by analyzing their loss landscape. First, we investigate the adversarial robustness with respect to its loss landscape. Through analyzing the loss landscape of adversarially trained models, we discover that the distortion of the loss landscape can occur, resulting in poor adversarial robustness. Based on this observation, we extend the loss landscape analysis to adversarial attacks and defenses to improve the adversarial robustness of deep learning models. We further analyze sharpness-aware minimization with its loss landscape and reveal that there exists a convergence instability problem due to its inherent algorithm. Specifically, whether the loss landscape in the parameter space has a saddle point can heavily affect the optimization and its generalization

performance. Given this phenomenon, we investigate the loss landscape with respect to perturbation in the parameter space and improve generalization performance by exploring a wider loss landscape.

Keywords: Deep learning, Robustness, Generalization, Loss landscape

Student Number: 2018-23641

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation of the Dissertation

Recent developments in deep learning have shown considerable enhancement in performance across a range of domains, such as computer vision and speech recognition. Based on these improvements, deep learning models are now actively used in real-world applications such as autonomous vehicles and smart home devices. A core mechanism behind the success of deep learning is minimizing training loss with a number of parameters, which we call empirical risk minimization (ERM). Given an input $\boldsymbol{x}$, model parameters $\boldsymbol{w}$, and a loss function $\ell(\boldsymbol{x}, \boldsymbol{w})$, ERM can be formulated as follows:

$$\min_{\boldsymbol{w}} \ell(\boldsymbol{x}, \boldsymbol{w}). \tag{1.1}$$

This has enabled deep learning models to reach near zero training loss and attain performance levels that were previously thought to be impossible.

However, recent research has revealed that this simple objective for deep learning models might result in two major issues: *adversarial vulnerability* and *poor generalization*. The intriguing property of deep learning models known as adversarial vulnerability corresponds to the phenomenon that deep learning models are vulnerable

to adversarial examples, which are intentionally perturbed to cause misclassification (Szegedy et al., 2013; Goodfellow et al., 2014). This adversarial vulnerability has been observed in various domains, including vision, audio, time series, etc. Moreover, adversarial vulnerability can lead to severe hazards in real-world applications, such as causing autonomous vehicle accidents by manipulating decision-making or extracting private information by circumventing voice authorization. (Papernot et al., 2017; Kurakin et al., 2016).

A method that generates adversarial examples is called an adversarial attack, which aims to find the perturbation $\boldsymbol{\delta}$ that maximizes the loss function for given input $\boldsymbol{x}$ and parameter $\boldsymbol{w}$:

$$\max_{\boldsymbol{\delta}} \ell(\boldsymbol{x} + \boldsymbol{\delta}, \boldsymbol{w}). \tag{1.2}$$

By using techniques, such as back-propagation and genetic algorithms, prior studies have found a large number of adversarial attacks in various fields.

To prevent malicious cases that arise from these adversarial attacks, many researchers aim to enhance the robustness of deep learning models against adversarial attacks. Adversarial robustness can be guaranteed by the following equation:

$$\min_{\boldsymbol{w}} \max_{\boldsymbol{\delta}} \ell(\boldsymbol{x} + \boldsymbol{\delta}, \boldsymbol{w}). \tag{1.3}$$

Since Szegedy et al. (2013) identified the existence of adversarial examples, many defenses have been proposed (Athalye et al., 2018a; Tramer et al., 2020), but the state-of-art performance is still insufficient to prevent models against possible adversarial examples (Madry et al., 2018; Zhang et al., 2019b).

Poor generalization, another issue with current deep learning models, is a large

gap between training accuracy and test accuracy. Under the i.i.d. assumption on data, minimizing training loss generally yields the best solution of $\boldsymbol{w}$ on the true distribution of data, however, this i.i.d assumption is often violated by the limitation of training data and model structure in common practice. Therefore, existing methods can successfully minimize the loss on train datasets, but this does not guarantee high performance on test datasets (Ishida et al., 2020; Foret et al., 2020).

To achieve an ideal generalization performance, improving the generalization of neural networks has been a core challenge in deep learning. Prior studies have suggested diverse techniques such as augmentation Zhang et al. (2017b); Yun et al. (2019) and regularization Srivastava et al. (2014); Ioffe and Szegedy (2015); Barrett and Dherin (2020), but recently researchers have discovered the relationship between the geometric characteristics of the loss surface and the generalization performance Hochreiter and Schmidhuber (1994); Keskar et al. (2017); Li et al. (2018). These studies discuss the geometric properties of loss surfaces and argue that the sharpness, which measures the flatness of a loss landscape, might be a core component of generalization. Following Hochreiter and Schmidhuber (1994); Keskar et al. (2017); Li et al. (2018), extensive theoretical and empirical analyses on the sharpness lead to new sharpness-aware training methods (Izmailov et al., 2018; He et al., 2019a; Chaudhari et al., 2019). Among them, sharpness-aware minimization (SAM) Foret et al. (2020) has shown state-of-the-art performance by reaching flat minima across various tasks and model structures Zhuang et al. (2021); Chen et al. (2022). SAM aims to minimize the worst-case loss over its parameter neighborhood rather than

minimizing the vanilla training loss $\ell(\boldsymbol{w}, \boldsymbol{w})$ as follows:

$$\min_{\boldsymbol{w}} \max_{\boldsymbol{\delta}} \ell(\boldsymbol{x}, \boldsymbol{w} + \boldsymbol{\delta}), \tag{1.4}$$

where $\boldsymbol{\delta}$ is a perturbation in the parameter space.

In this dissertation, we focus on the fact that both adversarial robustness Eq. (1.2) and generalization Eq. (1.4) depends on the loss landscape $\ell(\cdot)$. By analyzing their loss landscape, we aim to gain a deeper understanding of adversarial robustness and generalization performance.

1.2 Aims of the Dissertation

To enhance the adversarial robustness and generalization performance of deep learning models, we conduct four different works in this dissertation. The detailed summaries of this research are presented as follows:

Adversarial Robustness and Loss Landscape (Chapter 2) Adversarial examples are perturbations of inputs designed to deceive deep learning models by adding adversarial noise. Despite being undetectable to humans, these perturbations lead to incorrect classifications. To improve the robustness of deep-learning models against adversarial attacks, a line of work was proposed, with projected gradient descent (PGD) adversarial training (Madry et al., 2018) being one of the most successful approaches. However, this method requires a high computational cost due to multiple forward and back propagation during batch training. To reduce this cost, single-step adversarial training methods have been proposed, such as fast adversarial training (Wong et al., 2020). Although fast adversarial training is both robust and

efficient, it suffers from "catastrophic overfitting," where robustness against PGD suddenly decreases to 0%. In this study, we analyze the differences before and after catastrophic overfitting, and identify the relationship between distortion of the decision boundary and catastrophic overfitting. We find that sometimes a smaller perturbation is sufficient to fool the model, while the model is robust against larger perturbations during single-step adversarial training, which we call "decision boundary distortion."

Geometry-Aware Adversarial Attack and Defense (Chapter 3) Deep neural networks have demonstrated impressive performance in a variety of tasks, from image classification to speech recognition. However, they have been shown to be vulnerable to adversarial examples, which are generated by adding imperceptible noise to the original example. To better understand and improve the adversarial robustness of neural networks, two different works have been conducted in various applications. These works include generating transferable adversarial examples and improving adversarial training. Through these works, we discover that the transferability of audio adversarial examples is related to noise sensitivity, and that there is a negative effect of the smoothness regularize on maximizing the margin.

Generalization and Loss Landscape (Chapter 4) Sharpness-aware minimization is a recently proposed training method that seeks to find flat minima in deep learning, resulting in state-of-the-art performance. We investigate the convergence instability of SAM dynamics near a saddle point, utilizing the qualitative theory of dynamical systems. We theoretically prove that the saddle point can become an attractor under SAM dynamics, and that SAM diffusion is worse than that of vanilla

gradient descent in terms of saddle point escape. Additionally, we show that momentum and batch-size are important to mitigate the convergence instability and achieve high generalization performance.

Sharpness-Aware Minimization with Multi-Ascent (Chapter 5) Recent studies have discussed the geometric properties of loss surfaces in order to understand and improve the generalization performance of deep neural networks. They been argued that the sharpness, which measures the flatness of a minimum, may be a core component of generalization. In this chapter, we analyze the effect of the number of ascent steps on the inner maximization and investigate the difference between models trained with single-step and multi-step ascent. We demonstrate that perturbed weights obtained by different number of ascent steps have their unique gradient information and bring different effects on the perturbed loss surface. A new training method is proposed which utilizes all gradient information during multi-step ascent and is shown to improve generalization performance across various models and datasets.

1.3 Organization of the Dissertation

The remainder of this dissertation is organized as follows. In Chapter 2, by investigating the loss landscape of adversarially trained models, we first discover that distortion of the landscape can lead to poor adversarial robustness. In Chapter 3, we extend the loss landscape analysis to adversarial attacks and defenses to improve the adversarial robustness of deep learning models. In Chapter 4, we analyze sharpness-aware minimization and its loss landscape and reveal that the presence of

a saddle point in the parameter space can affect the optimization and generalization performance. In Chapter 5, we explore a wider loss landscape with respect to perturbation in the parameter space, resulting in improved generalization performance. Finally, we concludes the dissertation along with the contribution and future plan of the research.

Chapter 2

Adversarial Robustness and Loss Landscape

2.1 Chapter Overview

Adversarial examples are perturbed inputs that are designed to deceive machine learning classifiers by adding adversarial noises to the original data. Although such perturbations are sufficiently subtle and undetectable by humans, they result in an incorrect classification. Since deep-learning models were found to be vulnerable to adversarial examples (Szegedy et al., 2013), a line of work was proposed to mitigate the problem and improve robustness of the models. Among the numerous defensive methods, projected gradient descent (PGD) adversarial training (Madry et al., 2018) is one of the most successful approaches for achieving robustness against adversarial attacks. Although PGD adversarial training serves as a strong defensive algorithm, because it relies on a multi-step adversarial attack, a high computational cost is required for multiple forward and back propagation during batch training.

To overcome this issue, other studies (Shafahi et al., 2019; Wong et al., 2020) on reducing the computational burden of adversarial training using single-step adversarial attacks (Goodfellow et al., 2014) have been proposed. Among them, inspired by Shafahi et al. (2019), Wong et al. (2020) suggested fast adversarial training, which is a modified version of fast gradient sign method (FGSM) adversarial training de-

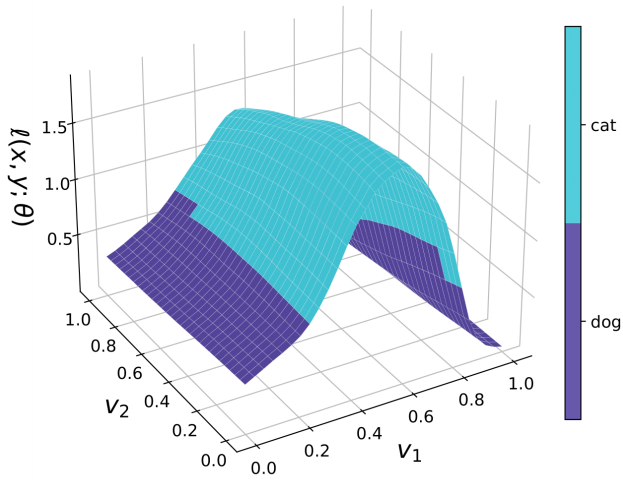


Figure 2.1: Visualization of distorted decision boundary. The origin indicates the original image x, the label of which is "dog". In addition, v_1 is the direction of a single-step adversarial perturbation and v_2 is a random direction. The adversarial image $x + v_1$ is classified as the correct label, although there is distorted interval where $x + k \cdot v_1$ is misclassified even when k is less than 1. Due to this decision boundary distortion, single-step adversarial training becomes vulnerable to multi-step adversarial attacks.

signed to be as effective as PGD adversarial training.

Fast adversarial training has demonstrated both robustness and efficiency; however, it suffers from the problem of "catastrophic overfitting," which is a phenomenon that robustness against PGD suddenly decreases to 0%, whereas robustness against FGSM rapidly increases. Wong et al. (2020) first discovered this issue and suggested the use of early stopping to prevent it. Later, it was found that catastrophic overfitting also occurs in different single-step adversarial training methods such as free adversarial training (Andriushchenko and Flammarion, 2020).

In this regard, few attempts have been made to discover the underlying reason for catastrophic overfitting and methods proposed to prevent this failure (An-

driushchenko and Flammarion, 2020; Vivek and Babu, 2020; Li et al., 2020a). However, these approaches were computationally inefficient or did not provide a fundamental reason for the problem.

In this study, we first analyze the differences before and after catastrophic overfitting. We then identify the relationship between distortion of the decision boundary and catastrophic overfitting. Unlike the previous notion in which a larger perturbation implies a stronger attack, we discover that sometimes a smaller perturbation is sufficient to fool the model, whereas the model is robust against larger perturbations during single-step adversarial training. We call this phenomenon "decision boundary distortion."

Figure 2.1 shows an example of decision boundary distortion by visualizing the loss surface. The model is robust to perturbations when the magnitude of the attack is equal to the maximum perturbation ϵ, but not to other smaller perturbations. When decision boundary distortion occurs, the model becomes more robust against a single-step adversarial attack but reveals fatal weaknesses to multi-step adversarial attacks and leads to catastrophic overfitting.

Through extensive experiments, we empirically discovered the relationship between single-step adversarial training and decision boundary distortion, and found that the problem of single-step adversarial training is a fixed magnitude of the perturbation, not the direction of the attack. Based on this observation, we present a simple algorithm that determines the appropriate magnitude of the perturbation for each image and prevents catastrophic overfitting. [1]

[1] This work was accepted to AAAI 2021 (Kim et al., 2021a).

2.2 Preliminaries

2.2.1 Adversarial Robustness

There are two major movements for building a robust model: provable defenses and adversarial training.

A considerable number of studies related to provable defenses of deep-learning models have been published. Provable defenses attempt to provide provable guarantees for robust performance, such as linear relaxations (Wong and Kolter, 2018; Zhang et al., 2019a), interval bound propagation (Gowal et al., 2018; Lee et al., 2020), and randomized smoothing (Cohen et al., 2019; Salman et al., 2019). However, provable defenses are computationally inefficient and show unsatisfied performance compared to adversarial training.

Adversarial training is an approach that augments adversarial examples generated by adversarial attacks (Goodfellow et al., 2014; Madry et al., 2018; Tramèr et al., 2017a). Because this approach is simple and achieves high empirical robustness for various attacks, it has been widely used and developed along with other deep learning methods such as mix-up (Zhang et al., 2017b; Lamb et al., 2019; Pang et al., 2019b) and unsupervised training (Alayrac et al., 2019; Najafi et al., 2019; Carmon et al., 2019).

In this study, we focus on adversarial training. Given an example $(x, y) \sim \mathcal{D}$, let $\ell(x, y; \theta) = \ell(f_\theta(x), y)$ denote the loss function of a deep learning model f with parameters θ. Then, adversarial training with a maximum perturbation ϵ can be formalized as follows:

$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}}\big[\max_{\delta\in\mathcal{B}(x,\epsilon)} \ell(x+\delta, y; \theta)\big] \tag{2.1}$$

A perturbation δ is in $\mathcal{B}(x, \epsilon)$ that denotes the ϵ-ball around an example x with a specific distance measure. The most used distance measures are L_0, L_2, and L_∞. In this study, we use L_∞ for such a measure.

However, the above optimization is considered as NP-hard because it contains a non-convex min-max problem. Thus, instead of the inner maximization problem, adversarial attacks are used to find the perturbation δ.

Fast gradient sign method (FGSM) (Goodfellow et al., 2014) is the simplest adversarial attack, which uses a sign of a gradient to find an adversarial image x'. Because FGSM requires only one gradient, it is considered the least expensive adversarial attack (Goodfellow et al., 2014; Madry et al., 2018).

$$x' = x + \epsilon \cdot \mathrm{sgn}(\nabla_x \ell(x, y; \theta)) \tag{2.2}$$

Projected gradient descent (PGD) (Madry et al., 2018) uses multiple gradients to generate more powerful adversarial examples. With a step size α, PGD can be formalized as follows:

$$x^{t+1} = \Pi_{\mathcal{B}(x,\epsilon)}(x^t + \alpha \cdot \mathrm{sgn}(\nabla_x \ell(x, y; \theta))) \tag{2.3}$$

where $\Pi_{\mathcal{B}(x,\epsilon)}$ refers the projection to the ϵ-ball $\mathcal{B}(x, \epsilon)$. Here, x^t is an adversarial example after t-steps. A large number of steps allows us to explore more areas in $\mathcal{B}(x, \epsilon)$. Note that PGDn corresponds to PGD with n steps (or iterations). For instance, PGD7 indicates that the number of PGD steps is 7.

2.2.2 Single-step and Multi-step Adversarial Attack

Single-step adversarial training was previously believed to be a non-robust method because it produces nearly 0% accuracy against PGD (Madry et al., 2018). Moreover,

the model trained using FGSM has been confirmed to have typical characteristics, such as gradient masking, which indicates that a single-step gradient is insufficient to find a decent adversarial examples (Tramèr et al., 2017a). For the above reasons, a number of studies have been conducted on multi-step adversarial attacks.

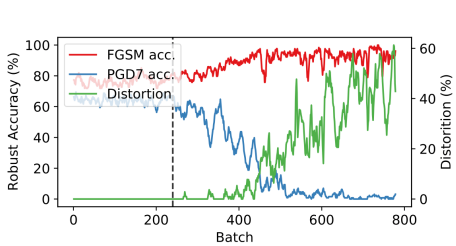
Contrary to this perception, however, free adversarial training (Shafahi et al., 2019) has achieved a remarkable performance with a single-step gradient using redundant batches and accumulative perturbations. Following Shafahi et al. (2019), Wong et al. (2020) proposed fast adversarial training using FGSM with a uniform random initialization. Fast adversarial training shows an almost equivalent performance to those of PGD (Madry et al., 2018) and free adversarial training (Shafahi et al., 2019).

$$\begin{aligned} \eta &= \text{Uniform}(-\epsilon, \epsilon) \\ \delta &= \eta + \alpha \cdot \text{sgn}(\nabla_\eta \ell(x + \eta, y; \theta)) \\ x' &= x + \delta \end{aligned} \tag{2.4}$$

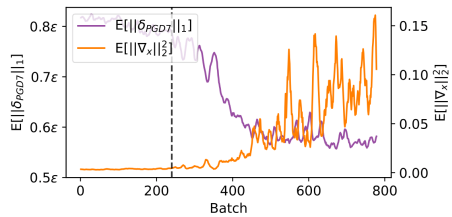
2.2.3 Catastrophic Overfitting

Although fast adversarial training performs well in a short time, a previously undiscovered phenomenon has been identified. That is, after a few epochs with single-step adversarial training, robustness of the model against PGD decreases sharply. This phenomenon is called catastrophic overfitting. Fast adversarial training (Wong et al., 2020) uses early stopping to temporally avoid catastrophic overfitting by tracking robustness accuracy against PGD on the training batches.

To apply early stopping, robustness against PGD must be continuously confirmed. Furthermore, standard accuracy does not yield the maximum potential (An-



(a) Robust accuracy and distortion



(b) Mean of absolute value of PGD7 perturbations and L_2 norm of the gradients of the images

Figure 2.2: (CIFAR10) Analysis of catastrophic overfitting. Plot (a) shows robust accuracy of fast adversarial training against FGSM (red) and PGD7 (blue). Distortion (green) denotes the ratio of images in distorted interval in Equation (2.5). Plot (b) shows the mean of absolute value of PGD7 perturbation $\mathbb{E}[||\delta_{PGD7}||_1]$ (purple) and the L_2 norm of the gradients of the images $\mathbb{E}[||\nabla_x||_2]$ (orange). Dashed black lines correspond to the 240th batch, which is the start point of catastrophic overfitting in both plots.

driushchenko and Flammarion, 2020). To resolve these shortcomings and gain a deeper understanding of catastrophic overfitting, a line of work has been proposed. Vivek and Babu (2020) identified that catastrophic overfitting arises with early overfitting to FGSM. To prevent this type of overfitting, the authors introduced dropout scheduling and demonstrated stable adversarial training for up to 100 epochs. In addition, Li et al. (2020a) trained a model with FGSM at first and then changed it into PGD when there was a large decrease in the PGD accuracy. Andriushchenko and Flammarion (2020) found that an abnormal behavior of a single filter leads to a nonlinear model with single-layer convolutional networks. Based on this observation, they proposed a regularization method, GradAlign, which maximizes $\cos(\nabla_x \ell(x, y; \theta), \nabla_x \ell(x + \eta, y; \theta))$ and prevents catastrophic overfitting by inducing a gradient alignment.

However, even with an increased understanding of catastrophic overfitting and

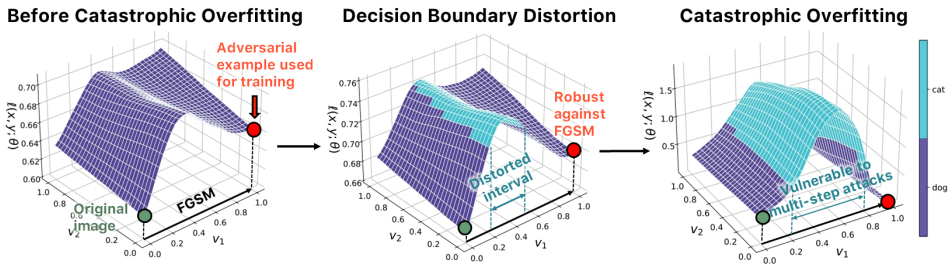


Figure 2.3: Process of normal decision boundary turns into distorted decision boundary. (Left) The loss surface before catastrophic overfitting with a FGSM adversarial direction v_1 and a random direction v_2. The red point denotes an adversarial example $x+v_1$ generated from the original image x, the label of which is "dog." (Middle) The changed loss surface after learning adversarial example $x + v_1$. Here, v_1 is the same vector as that on the left. Distorted interval begins to occur for the first time. (Right) As training continues, distorted decision boundary grows uncontrollably such that robustness against multi-step adversarial attacks decreases.

methods for its prevention, a key question remains unanswered:

What characteristic of single-step adversarial attacks is the cause of catastrophic overfitting?

In this chapter, we discuss the cause of catastrophic overfitting in the context of single-step adversarial training. We then propose a new simple method to facilitate stable single-step adversarial training, wherein longer training can produce a higher standard accuracy with sufficient adversarial robustness.

2.3 Methodology

2.3.1 Revisiting Catastrophic Overfitting

First, to analyze catastrophic overfitting, we start by recording robust accuracy of fast adversarial training on CIFAR-10 (Krizhevsky et al., 2009). The maximum perturbation ϵ is fixed to 8/255. We use FGSM and PGD7 to verify robust accuracy

with the same settings $\epsilon = 8/255$ and a step size $\alpha = 2/255$.

Figure 2.2 shows statistics on the training batch when catastrophic overfitting occurs (71st out of 200 epochs). In plot (a), after 240 batches, robustness against PGD7 begins to decrease rapidly; conversely, robustness against FGSM increases. Plot (b) shows the mean of the absolute value of PGD7 perturbation $\mathbb{E}[||\delta_{PGD7}||_1]$ and squared L_2 norm of the gradient of the images $\mathbb{E}[||\nabla_x||_2]$ of each batch. After catastrophic overfitting, there is a trend of decreasing mean perturbation. This is consistent with the phenomenon in which the perturbations of the catastrophic overfitted model are located away from the maximum perturbation, unlike the model that is stopped early (Wong et al., 2020). Concurrently, a significant increase in the squared L_2 norm of the gradient is also observed. The highest point indicates a large difference, approximately 35 times greater than that before catastrophic overfitting.

These two observations, a low magnitude of perturbations and a high gradient norm, make us wonder what would the loss surface looks like. Figure 2.3 illustrates the progress of adversarial training in which catastrophic overfitting occurs. The loss surface of the perturbed example is shown, where the green spot denotes the original images and the red spot denotes the adversarial example used for adversarial training in the batch. The v_1 axis indicates the direction of FGSM, whereas the v_2 axis is a random direction. The true label of the original sample is "dog." Hence, the purple area indicates where the perturbed sample is correctly classified, whereas the blue area indicates a misclassified area.

On the left side of Figure 2.3, we can easily observe that the model is robust against FGSM. However, after training the batch, an interval vulnerable to a smaller perturbation than the maximum perturbation ϵ appears, whereas the model is still

robust against FGSM. This distorted interval implies that the adversarial example with a larger perturbation is weaker than that with a smaller perturbation, which is contrary to the conventional belief that a larger magnitude of perturbation induces a stronger attack. As a result, the model with distorted interval is vulnerable to multi-step adversarial attacks that can search the vulnerable region further inside $\mathcal{B}(x, \epsilon)$. As the training continues, the area of distorted interval increases as shown in the figure on the right. It is now easier to see that the model is now perfectly overfitted for FGSM, yet loses its robustness to the smaller perturbations. We call this phenomenon "decision boundary distortion."

The evidence of decision boundary distortion is also shown in Figure 2.2 (b). When robustness against PGD7 sharply decreases to 0%, the mean of the absolute value of PGD7 perturbation $\mathbb{E}[||\delta_{PGD7}||_1]$ decreases. It indicates that, when catastrophic overfitting arises, a smaller perturbation is enough to fool the model than the maximum perturbation ϵ, which implies that distorted interval exists. In addition, during the process of having distorted decision boundary, as shown in the figure on the right, the loss surface inevitably becomes highly curved, which matches the observation of increasing the L_2 norm of the gradients of the images $\mathbb{E}[||\nabla_x||_2]$. This is also consistent with previous research (Andriushchenko and Flammarion, 2020). Andriushchenko and Flammarion (2020) argued that $\nabla_x \ell(x, y; \theta)$ and $\nabla_x \ell(x + \eta, y; \theta)$ tend to be perpendicular in catastrophic overfitted models where η is drawn from a uniform distribution $U(-\epsilon, \epsilon)$. Considering that a highly curved loss surface implies $(\nabla_x \ell(x, y; \theta))^T (\nabla_x \ell(x + \eta, y; \theta)) \approx 0$ in high dimensions, the reason why GradAlign (Andriushchenko and Flammarion, 2020) can avoid catastrophic overfitting might be because the gradient alignment leads the model to learn a linear loss surface

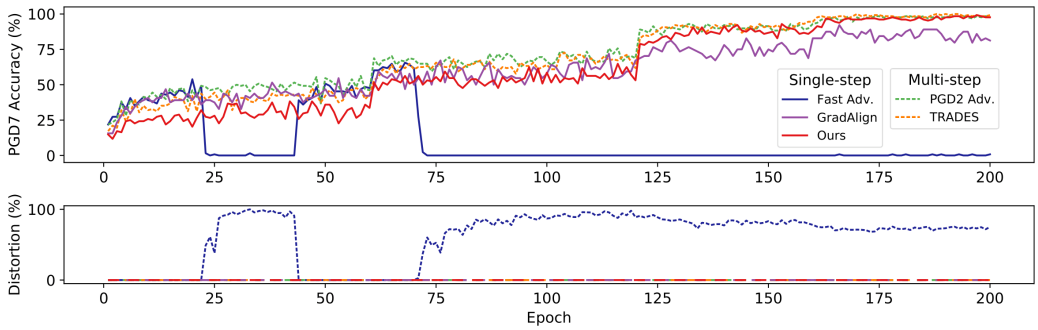


Figure 2.4: (CIFAR10) Robust accuracy and distortion on the training batch for each epoch. Two multi-step adversarial attacks show zero distortion during the entire training time and reach nearly 100% PGD7 accuracy. By contrast, fast adversarial training shows high distortion and eventually collapses after the 71st epoch. The proposed method successfully avoids such problems and achieves a high PGD7 accuracy similiar to multi-step adversarial training (Best viewed in color).

which reduces the chance of having distorted decision boundary.

We next numerically measured the degree of decision boundary distortion. To do so, we first define a new measure distortion d. Given a deep learning model f and a loss function ℓ, distortion d can be formalized as follows:

$$\begin{aligned} \mathbf{S}_D &= \{x | \exists k \in (0,1) \text{ s.t. } f(x + k \cdot \epsilon \cdot \text{sgn}(\nabla_x \ell)) \neq y\} \\ \mathbf{S}_N &= \{x | f(x) = y, f(x + \epsilon \cdot \text{sgn}(\nabla_x \ell)) = y\} \\ d &= \frac{|\mathbf{S}_D \cap \mathbf{S}_N|}{|\mathbf{S}_N|} \end{aligned} \tag{2.5}$$

where (x, y) is an example drawn from dataset $\mathcal{D}$. However, because the loss function of the model is not known explicitly, we use a number of samples to estimate distortion d. In all experiments, we tested 100 samples in the adversarial direction $\delta = \epsilon \cdot \text{sgn}(\nabla_x \ell)$ for each example. Indeed, we can see that distortion increases in Figure 2.2 (a) when catastrophic overfitting arises.

To verify that decision boundary distortion is generally related to catastrophic

overfitting, we demonstrate how distortion and robustness against PGD7 change during training. We conducted an experiment on five different models: fast adversarial training (Fast Adv.) (Wong et al., 2020), PGD2 (PGD2 Adv.) (Madry et al., 2018), TRADES (Zhang et al., 2019b), GradAlign (Andriushchenko and Flammarion, 2020), and the proposed method (Ours). All models were tested on $\epsilon = 8/255$. The step size α is set to $\alpha = 1.25\epsilon$, $\alpha = 1/2\epsilon$, and $\alpha = 1/4\epsilon$ for fast adversarial training, PGD2 adversarial training, and TRADES, respectively. We also conducted same experiment on PGD adversarial training with different number of steps; however, because these show similar results to PGD2 adversarial training, we only included PGD2. TRADES is trained with seven steps.

As the key observation in Figure 2.4, the point where decision boundary distortion begins in fast adversarial training (22nd epoch) is identical to the point where robustness against PGD7 sharply decreases; that is, catastrophic overfitting occurs. Then, when decision boundary distortion disappears (45th to 72nd epoch), the model immediately recovers robust accuracy. After the 72nd epoch, the model once again suffers a catastrophic overfitting and never regains its robustness with high distortion. Hence, we conclude that there is a close connection between decision boundary distortion and the vulnerability of the model against multi-step adversarial attacks.

2.3.2 Stable Single-Step Adversarial Training

Based on previous results, we assume that distorted decision boundary might be the reason for catastrophic overfitting. Here, we stress that the major cause of distorted decision boundary is that single-step adversarial training uses a point with a fixed distance ϵ from the original image x as an adversarial image x' instead of an optimal solution of the inner maximum in Equation (2.1). Under this linearity assumption,

the most powerful adversarial perturbation δ would be the same as $\epsilon \cdot \text{sgn}(\nabla_x \ell)$ where ϵ is the maximum perturbation, and the following formula should be satisfied.

$$\begin{aligned} \ell(x+\delta) - \ell(x) &= (\nabla_x \ell)^T \delta \\ &= (\nabla_x \ell)^T \epsilon \cdot \text{sgn}(\nabla_x \ell) \\ &= \epsilon ||\nabla_x \ell||_1 \end{aligned} \tag{2.6}$$

However, as confirmed in the previous scetion, decision boundary distortion with a highly curved loss surface has been observed during the training phase, which indicates that ϵ is no longer the strongest adversarial step size in the direction of δ. Thus, the linear approximation of the inner maximization is not satisfied when distorted decision boundary arises.

To resolve this issue, we suggest a simple fix to prevent catastrophic overfitting by forcing the model to verify the inner interval of the adversarial direction. In this case, the appropriate magnitude of the perturbation should be taken into consideration instead of using ϵ:

$$\begin{aligned} \delta &= \epsilon \cdot \text{sgn}(\nabla_x \ell) \\ &\arg \max_{k \in [0,1]} \ell(x + k \cdot \delta, y; \theta) \end{aligned} \tag{2.7}$$

Here, we introduce k, which denotes the scaling parameter for the original adversarial direction $\text{sgn}(\nabla_x \ell)$. In contrast to previous single-step adversarial training which uses a fixed size of $k = 1$, an appropriate scaling parameter k^* helps the model to train

Algorithm 1: Stable single-step adversarial training

Parameter: B mini-batches, a perturbation size ϵ, a step size α, and c check points for a network f_θ

for $i = 1, ..., B$ **do**

$\eta = \text{Uniform}(-\epsilon, \epsilon)$

$\hat{y}_{i,0} = f_\theta(x_i + \eta)$

$\delta = \eta + \alpha \cdot \nabla_\eta \ell(\hat{y}_{i,0}, y_i)$

for $j = 1, ..., c$ **do**

$\hat{y}_{i,j} = f_\theta(x_i + j \cdot \delta / c))$

end

$x'_i = x_i + \min(\{k | \hat{y}_{i,k} \neq y_i\} \cup \{1\}) \cdot \delta / c$

$\theta = \theta - \nabla_\theta \ell(f_\theta(x'_i), y_i)$

end

stronger adversarial examples as follows:

$$\begin{aligned} &\delta = \epsilon \cdot \text{sgn}(\nabla_x \ell) \\ &k^* = \min_{k \in [0,1]} \{k | y \neq f(x + k \cdot \delta; \theta)\} \\ &\min_\theta \mathbb{E}_{(x,y) \sim \mathcal{D}} [\ell(x + k^* \cdot \delta, y; \theta)] \end{aligned} \tag{2.8}$$

In this way, regardless of the linearity assumption, we can train the model with stronger adversarial examples that induce an incorrect classification in the adversarial direction. Simultaneously, we can also detect distorted decision boundary by inspecting the inside of distorted interval, as shown in Figure 2.3.

However, because we do not know the explicit loss function of the model, forward propagation is the only approach for checking the adversarial images in the single-step adversarial attack direction. Hence, we propose the following simple method. First, we calculate the single-step adversarial direction δ. Next, we choose multiple checkpoints $(x + \frac{1}{c}\delta, ..., x + \frac{c-1}{c}\delta, x + \delta)$. Here, c denotes the number of checkpoints except for the clean image x, which is tested in advance during the single-step

adversarial attack process. We then feed all checkpoints to the model and verify that the predicted label $\hat{y}_j$ matches the correct label y for all checkpoints $x + \frac{j}{c}\delta$ where $j \in \{1, ..., c\}$. Among the incorrect images and the clean image x, the smallest j is selected; if all checkpoint are correctly classified, the adversarial image $x' = x + \delta$ is used. Algorithm 1 shows a summary of the proposed method.

Suppose the model has L layers with n neurons. Then, the time complexity of forward propagation is $O(Ln^2)$. Considering that backward propagation has the same time complexity, the generation of one adversarial example requires $O(2Ln^2)$ in total. Thus, with c checkpoints, the proposed method consumes $O((c + 4)Ln^2)$ because it requires one adversarial direction $O(2Ln^2)$, forward propagation for c checkpoints $O(cLn^2)$, and one optimization step $O(2Ln^2)$. Compared to PGD2 adversarial training, which demands $O(6Ln^2)$, the proposed method requires more time when $c > 2$. However, the proposed method does not require additional memory for computing the gradients of the checkpoints because we do not need to track a history of variables for backward propagation; hence, larger validation batch sizes can be considered. Indeed, the empirical results indicate that the proposed method consumes less time than PGD2 adversarial training under $c \leq 4$.

Figure 2.4 shows that the proposed method successfully avoids catastrophic overfitting despite using a single-step adversarial attack. Furthermore, the proposed model not only achieves nearly 100% robustness against PGD7, which fast adversarial training cannot accomplish, but also possesses zero distortion until the end of the training. This is the opposite of the common understanding that single-step adversarial training methods cannot perfectly defend the model against multi-step adversarial attacks.

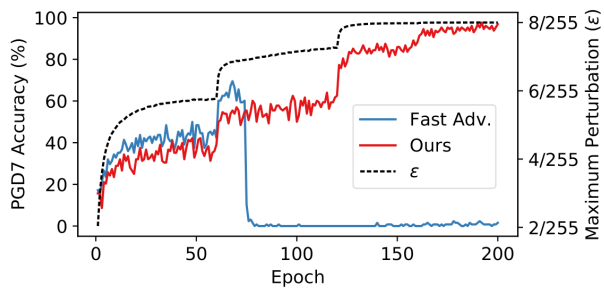


Figure 2.5: (CIFAR10) Comparison of PGD7 accuracy on the training batch between fast adversarial training with ϵ-scheduling and the proposed method. The dashed line indicates the average maximum perturbation $\mathbb{E}\left[||\delta||_\infty\right]$ calculated from the proposed method for each epoch and is used as the maximum perturbation of fast adversarial training.

The proposed model learns the image with the smallest perturbation among the incorrect adversarial images. In other words, during the initial states, the model outputs incorrect predictions for almost every image such that $\min(\{k|\hat{y}_{i,k} \neq y_i\} \cup \{1\}) = 0$ in Algorithm 1. As additional batches are trained, the average maximum perturbation $\mathbb{E}[||\delta||_\infty]$ increases, as in Figure 2.5, where $\delta = x' - x$ and x' is selected by the proposed method. Thus, the proposed method may appear to simply be a variation of ϵ-scheduling. In order to point out the difference, fast adversarial training with ϵ-scheduling is also considered. For each epoch, we use the average maximum perturbation $\mathbb{E}[||\delta||_\infty]$ calculated from the proposed method as the maximum perturbation ϵ. The result is summarized in Figure 2.5.

Notably, ϵ-scheduling cannot help fast adversarial training avoid catastrophic overfitting. The main difference between ϵ-scheduling and the proposed method is that, whereas ϵ-scheduling uniformly applies the same magnitude of the perturbation for every image, the proposed method gradually increases the magnitude of the perturbation appropriately by considering the loss surface of each image. Therefore,

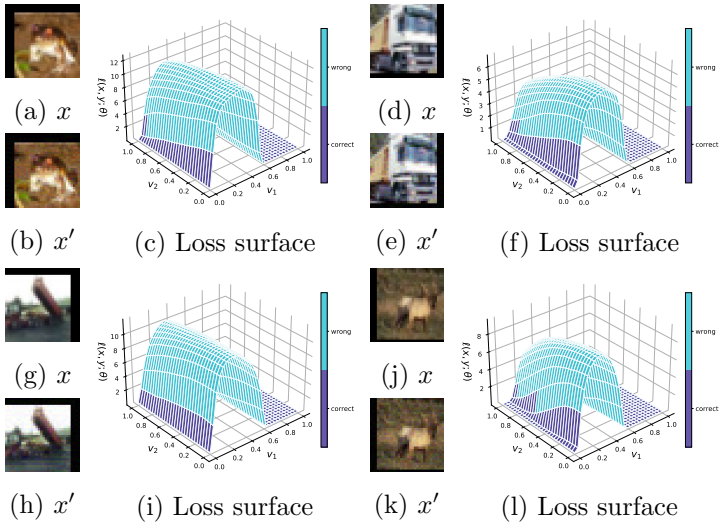
in contrast to ϵ-scheduling, the proposed method successfully prevents catastrophic overfitting, despite the same size of the average perturbation used during the training process.

2.4 Experiments

2.4.1 Experimental Setup

In this section, we conduct a set of experiments on CIFAR10 (Krizhevsky et al., 2009) and Tiny ImageNet (Le and Yang, 2015) , using PreAct ResNet-18 (He et al., 2016a). Input normalization and data augmentation including 4-pixel padding, random crop and horizontal flip are applied. We use SGD with a learning rate of 0.01, momentum of 0.9 and weight decay of 5e-4. To check whether catastrophic overfitting occurs, we set the total epoch to 200. The learning rate decays with a factor of 0.2 at 60, 120, and 160 epochs. All experiments were conducted on a single NVIDIA TITAN V over three different random seeds. Our implementation in PyTorch (Paszke et al., 2019) with Torchattacks (Kim, 2020) is available at https://github.com/Harry24k/catastrophic-overfitting.

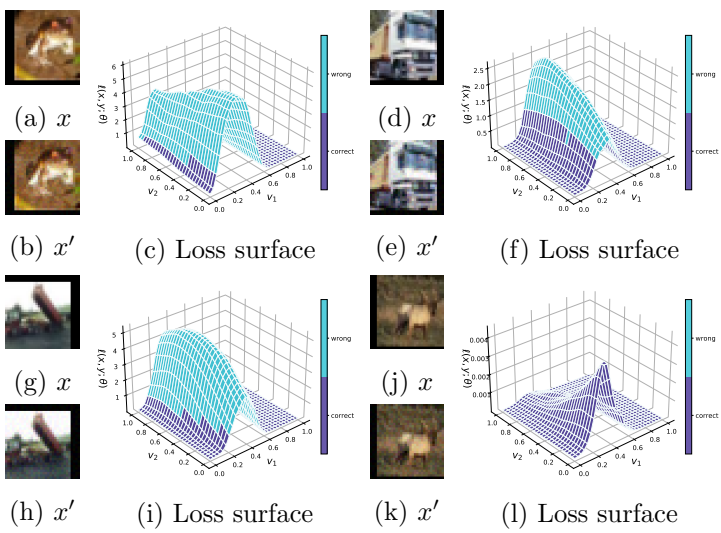
During the training session, the maximum perturbation ϵ was set to $8/255$. For PGD adversarial training, we use a step size of $\alpha = \max(2/255, \epsilon/n)$, where n is the number of steps. TRADES uses $\alpha = 2/255$ and seven steps for generating adversarial images. Following Wong et al. (2020), we use $\alpha = 1.25\epsilon$ for fast adversarial training and the proposed method. The regularization parameter β for the gradient alignment of GradAlign is set to 0.2 as suggested by Andriushchenko and Flammarion (2020).



(a) x (b) x' (c) Loss surface (d) x (e) x' (f) Loss surface

(g) x (h) x' (i) Loss surface (j) x (k) x' (l) Loss surface

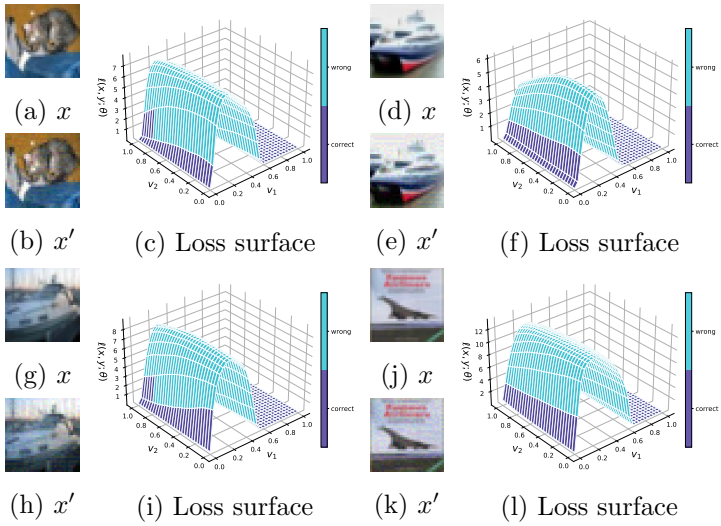
Figure 2.6: (CIFAR10) Direction of FGSM adversarial perturbation v_1 and random direction v_2.



(a) x (b) x' (c) Loss surface (d) x (e) x' (f) Loss surface

(g) x (h) x' (i) Loss surface (j) x (k) x' (l) Loss surface

Figure 2.7: (CIFAR10) Direction of fast adversarial perturbation v_1 and random direction v_2.



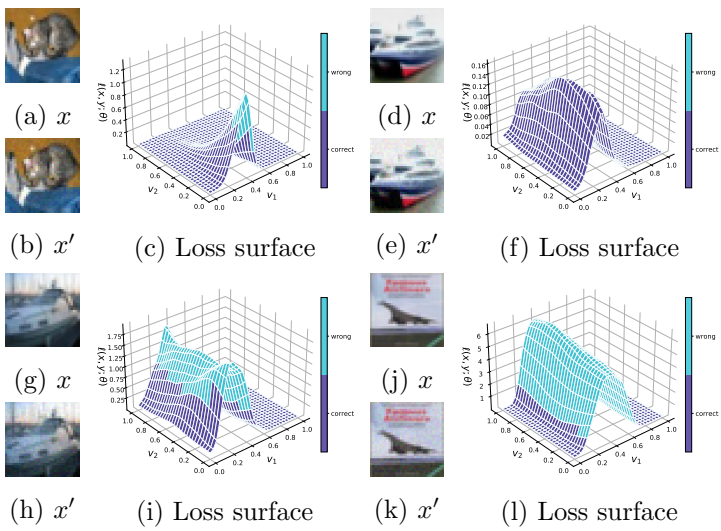
(a) x (c) Loss surface (d) x (f) Loss surface

(b) x' (e) x'

(g) x (i) Loss surface (j) x (l) Loss surface

(h) x' (k) x'

Figure 2.8: (CIFAR10) Direction of FGSM adversarial perturbation v_1 and random direction v_2.



(a) x (c) Loss surface (d) x (f) Loss surface

(b) x' (e) x'

(g) x (i) Loss surface (j) x (l) Loss surface

(h) x' (k) x'

Figure 2.9: (CIFAR10) Direction of fast adversarial perturbation v_1 and random direction v_2.

2.4.2 Visualizing Decision Boundary Distortion

CIFAR10

In this section, we aim to provide more figures to describe the relationship between catastrophic overfitting and decision boundary distortion. First, we plot various shapes of distorted decision boundary. We note that it is not difficult to observe distorted decision boundary of the catastrophic overfitted model.

Figure 2.6 shows distorted decision boundary of the fast adversarial trained model on the first four images in the training set of CIFAR10. Although we applied random crop and padding, the loss surface has distorted interval in the direction of their FGSM adversarial perturbation.

In Figure 2.7, we plot distorted decision boundary in the direction of perturbation obtained by the attack used in fast adversarial training. Similarly, even though the attack used in fast adversarial training uses a uniform random initialization $U(-\epsilon, \epsilon)$ before adding a sign of gradient, decision boundary distortion is observed for most of the cases. This indicates that single-step adversarial training cannot escape from decision boundary distortion by using a random initialization.

Similar results were found for the test set of CIFAR10 as shown in Figures 2.8 and 2.9. Each figure shows the loss surface of the fast adversarial trained model for the first four images in the test set.

Tiny ImageNet

To push further, we conducted the same experiment on Tiny ImageNet to observe the relationship between catastrophic overfitting and distorted decision boundary. As shown in Figure 2.10, PGD2 adversarial training shows zero distortion during the

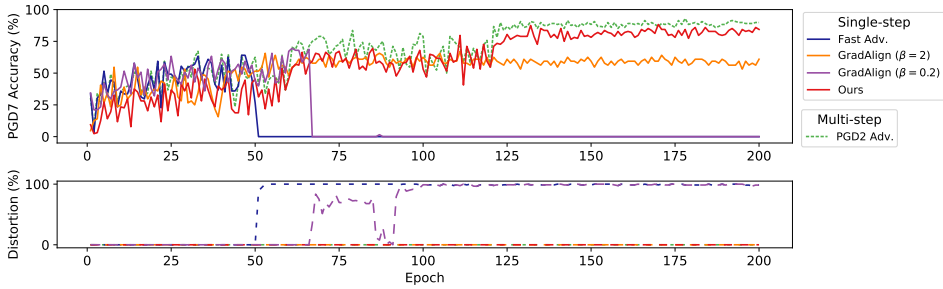
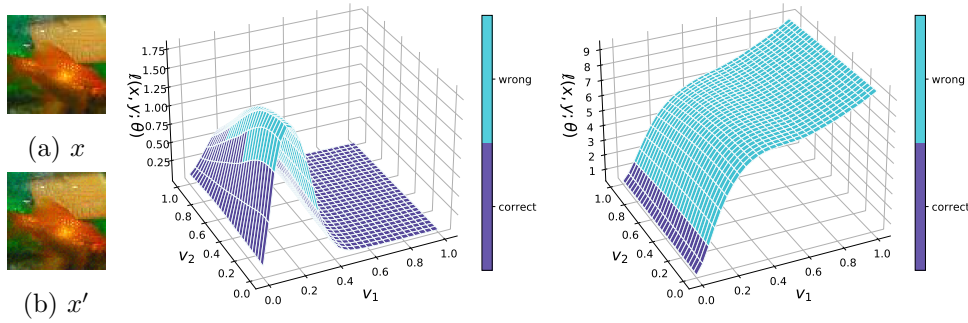


Figure 2.10: (Tiny ImageNet) Robust accuracy and distortion on the training set for each epoch.



(a) x

(b) x'

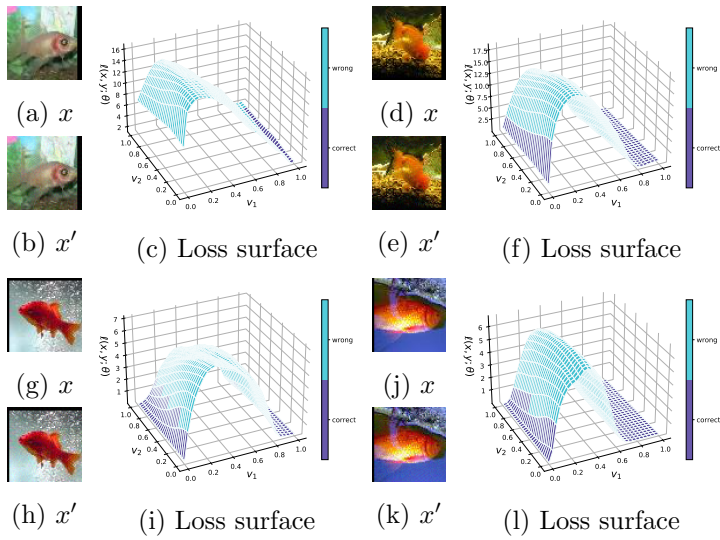
(c) Loss surface (67th epoch)

(d) Loss surface (89th epoch)

Figure 2.11: (Tiny ImageNet) Direction of FGSM adversarial perturbation v_1 and random direction v_2.

training time and achieves nearly 100% accuracy against PGD7. Among the single-step adversarial training methods, fast adversarial training shows 0% of robustness against PGD7 with high distortion. The proposed method avoids catastrophic overfitting and achieves a high PGD7 accuracy, similar to multi-step adversarial training.

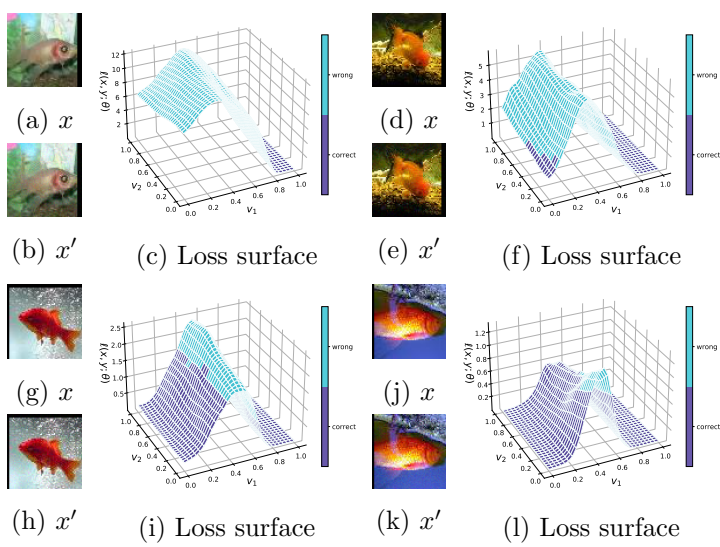
Noteworthy is the fact that GradAlign shows unusual behavior for different regularization parameter β for the gradient alignment. First, contrary to CIFAR10, catastrophic overfitting occurs in GradAlign with $\beta = 0.2$ at the 67th epoch on Tiny ImageNet. At 89th epoch, distortion of GradAlign decreases to 0% even when the



(a) x (b) x' (c) Loss surface (d) x (e) x' (f) Loss surface

(g) x (h) x' (i) Loss surface (j) x (k) x' (l) Loss surface

Figure 2.12: (Tiny ImageNet) Direction of FGSM adversarial perturbation v_1 and random direction v_2.



(a) x (b) x' (c) Loss surface (d) x (e) x' (f) Loss surface

(g) x (h) x' (i) Loss surface (j) x (k) x' (l) Loss surface

Figure 2.13: (Tiny ImageNet) Direction of fast adversarial perturbation v_1 and random direction v_2.

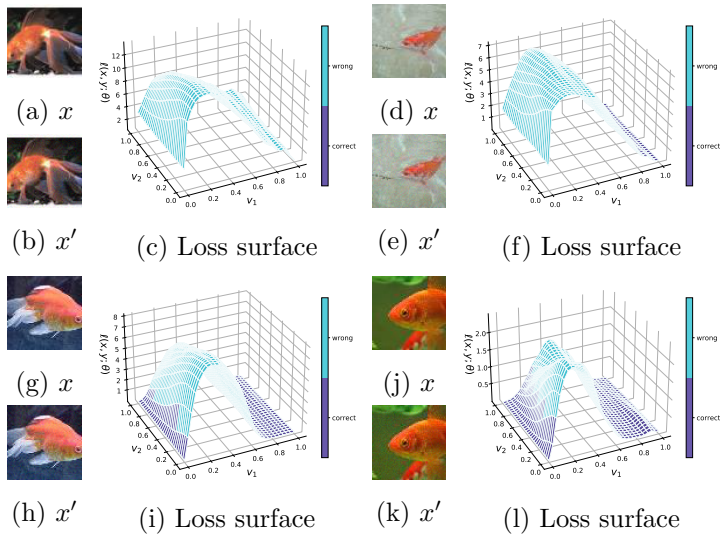


Figure 2.14: (Tiny ImageNet) Direction of FGSM adversarial perturbation v_1 and random direction v_2.

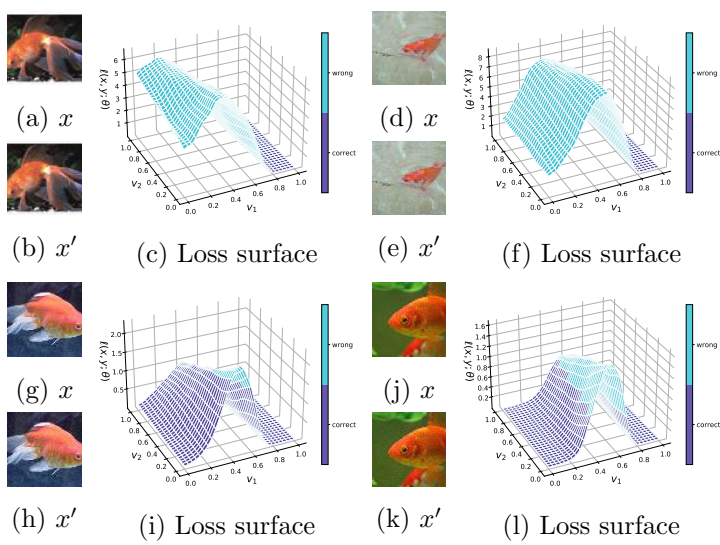


Figure 2.15: (Tiny ImageNet) Direction of fast adversarial perturbation v_1 and random direction v_2.

accuracy against PGD7 is nearly zero. To investigate this result, we plot the loss surface of GradAlign for each epoch. As in Figure 2.11, GradAlign with $\beta = 0.2$ shows distorted decision boundary at the 67th epoch. However, at the 89th epoch, although gradient alignment works properly after catastrophic overfitting, the model cannot escape from the local minima and thus PGD7 accuracy is still zero.

Considering that the successful value of β was different for different maximum perturbation ϵ as Andriushchenko and Flammarion (2020) noted, we also conducted the same experiment with different β because Tiny ImageNet is a more complicated dataset than CIFAR10. We additionally use $\beta = 2$ and 0.02. When we use $\beta = 2$, GradAlign shows zero-distortion until the end of the training. However, robustness against PGD7 on the training batch cannot achieve near 100% unlike PGD adversarial training and the proposed method. We think this is an interesting phenomenon that might help the community interested to continue further research in better schemes for single-step adversarial training. Note that the result of GradAlign with $\beta = 0.02$ was similar to $\beta = 0.2$.

Finally, we plot distorted decision boundary of the fast adversarial trained model on Tiny ImageNet. Figure 2.12 and Figure 2.13 show the results on the training set and Figure 2.14 and Figure 2.15 on the test set. Interestingly, the model occasionally outputs correct answers for the adversarial image but not for the original image, for example, Figure 2.13 (c). This can be considered additional evidence that the model is overfitted to only single-step adversarial images.

2.4.3 Distortion and Nonlinearity of the Loss Function

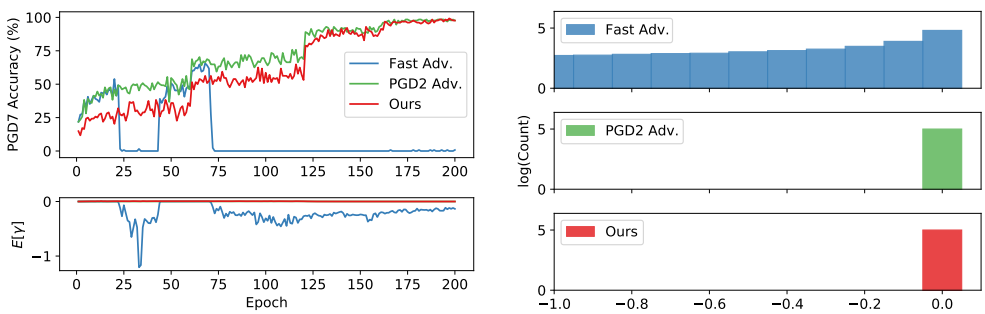
Previously, we mentioned that single-step adversarial training leads to a highly curved loss surface, and single-step adversarial attacks such as FGSM cannot gen-

erate strong adversarial examples because the following linearity assumption is not satisfied.

$$\ell(x+\delta) - \ell(x) = \epsilon||\nabla_x \ell||_1 \tag{2.9}$$

Here, to detect decision boundary distortion from a different perspective of nonlinearity, we propose a new measure γ for the nonlinearity of the loss function based on Equation (2.9):

$$\gamma = \{\ell(x+\delta) - \ell(x)\} - \epsilon||\nabla_x \ell||_1 \tag{2.10}$$



(a) Change in $\mathbb{E}[\gamma]$ (b) Distribution of γ

Figure 2.16: (CIFAR10) Nonlinearity of the loss function. In plot (a), the top figure shows robust accuracy against PGD7 on the entire training images. The bottom figure shows the average of the nonlinearity of the loss function over entire training images. Plot (b) shows the distribution of γ for each model at the last epoch.

Figure 2.16 demonstrates the different characteristics of γ for each method. Plot (a) shows the mean value of γ for all 50,000 CIFAR10 training images. When catastrophic overfitting occurs, $\mathbb{E}[\gamma]$ becomes negative. The negative $\mathbb{E}[\gamma]$ of fast adversarial training indicates that the loss function becomes nonlinear, which is one of the main characteristics of distorted decision boundary.

	Method	Standard	FGSM	PGD50	Black-box	AA	Time (h)
Multi-step	PGD2 Adv.	**86.6±0.8**	49.7±2.6	36.0±2.3	**85.6±0.8**	34.8±2.1	4.5
	PGD4 Adv.	86.0±0.8	49.6±3.0	36.7±2.9	85.3±0.8	35.4±2.6	6.7
	PGD7 Adv.	84.4±0.2	**51.5±0.1**	**40.5±0.1**	83.8±0.2	**39.4±0.2**	11.1
	TRADES	85.3±0.4	50.7±1.6	39.3±1.9	84.4±0.4	38.6±2.0	15.1
Single-step	Fast Adv.	84.5±4.3	**95.1±6.8**	0.1±0.1	80.8±8.7	0.0±0.0	3.2
	GradAlign	83.9±0.2	44.3±0.0	31.7±0.2	83.3±0.3	30.9±0.2	13.6
	Ours ($c = 2$)	86.8±0.3	48.3±0.5	32.5±0.2	85.9±0.1	30.9±0.2	3.5
	Ours ($c = 3$)	87.7±0.8	50.5±2.4	**33.9±2.3**	86.7±0.9	**32.3±2.2**	3.9
	Ours ($c = 4$)	**87.8±0.9**	50.5±2.3	33.7±2.4	**87.0±0.8**	32.2±2.4	4.4

Table 2.1: Standard and robust accuracy (%) and training time (hour) on CIFAR10.

Method	Standard	FGSM	PGD50	Time (h)
PGD2 Adv.	46.3±1.2	14.7±2.7	**10.3±2.7**	27.7
Fast Adv.	26.2±0.7	**49.0±5.7**	0.0±0.0	19.6
Ours ($c = 3$)	**49.6±1.5**	12.5±0.1	7.8±0.1	25.7

Table 2.2: Standard accuracy and robustness (%) and training time (h) on Tiny ImageNet.

Similarly, in plot (b), the distribution of γ of fast adversarial training shows a totally different behavior from that of PGD2 adversarial training and the proposed method. Compared to the proposed method and PGD2 adversarial training with few negative values, there are noticeably many negative values in fast adversarial training. These results indicate the loss function ℓ of catastrophic overfitted model follows $\ell(x + \delta) - \ell(x) \leq \epsilon||\nabla_x \ell||_1$, which implies highly nonlinear loss surface.

2.4.4 Adversarial Robustness

We then evaluate robustness on the test set. FGSM and PGD50 with 10 random restarts are used for evaluating robustness of the models. Furthermore, to estimate accurate robustness and detect gradient obfuscation (Athalye et al., 2018a), we also consider PGD50 adversarial images generated from Wide-ResNet 40-10 (Zagoruyko and Komodakis, 2016a) trained on clean images (Black-box), and AutoAttack (AA)

which is one of the latest strong adversarial attacks proposed by Croce and Hein (2020).

Tables 2.1 and 2.2 summarize the results. From Table 2.1, we can see that multi-step adversarial training methods yield more robust models, but generally requires a longer computational time. In particular, TRADES requires over 15 hours, which is 5-times slower than the proposed method. Among the single-step adversarial training methods, fast adversarial training is computationally efficient, however, because catastrophic overfitting has occurred, it shows 0% accuracy against PGD50 and AA.

Interestingly, we observe that fast adversarial training achieves a higher accuracy for FGSM adversarial images than clean images in both datasets, which does not appear in other methods. The accuracy when applying FGSM on only correctly classified images is 84.4% on CIFAR10, whereas all other numbers remain almost unchanged when we use attacks on correctly classified clean images.

The proposed method, by contrast, shows the best standard accuracy and robustness against PGD50, Black-box, and AA with a shorter time. GradAlign also provides sufficient robustness; however, it takes 3-times longer than the proposed method. As shown in Table 2.2, similar results are observed on Tiny ImageNet. We include the results of the main competitors, PGD2 adversarial training, fast adversarial training, and the proposed method with $c = 3$ which shows the best performance on CIFAR10. Here again, the proposed method shows high standard accuracy and adversarial robustness close to that of PGD2 adversarial training.

2.5 Chapter Summary

In this study, we empirically showed that catastrophic overfitting is closely related to decision boundary distortion by analyzing their loss surface and robustness during training. Decision boundary distortion provides a reliable understanding of the phenomenon in which a catastrophic overfitted model becomes vulnerable to multi-step adversarial attacks, while achieving a high robustness on the single-step adversarial attacks. Based on these observations, we suggested a new simple method that determines the appropriate magnitude of the perturbation for each image. Further, we evaluated robustness of the proposed method against various adversarial attacks and showed sufficient robustness using single-step adversarial training without the occurrence of any catastrophic overfitting.

Chapter 3

Geometry-Aware Adversarial Attack and Defense

3.1 Chapter Overview

Deep neural networks have shown promising performance in a wide range of tasks, from image classification to speech recognition. Despite their success, neural networks have been shown to be vulnerable to *adversarial examples* Szegedy et al. (2013); Goodfellow et al. (2014) generated by adding imperceptible noise to the original example Biggio and Roli (2018). However, both adversarial attacks and defenses are not perfectly understood, and still have unexplored areas. To understand the adversarial vulnerability and improve the adversarial robustness of neural networks, we conduct two different works in various applications as follows:

1. *Generating Transferable Adversarial Examples:* In this study, we provide a thorough evaluation of the transferability of audio adversarial examples between various models under both silent and noisy conditions. We discover that the transferability of audio adversarial examples is related to noise sensitivity. Based on these observations, We propose a novel attack that generates highly transferable audio adversarial examples by intentionally adding noise during the gradient ascent process. [1]

[1] This work was accepted to Pattern Recognition (Kim et al., 2022b).

2. *Improving Adversarial Training:* We investigate the characteristics of the regularizers of the most popular adversarial training methods, AT and TRADES, and find that there exists a negative effect of the smoothness regularizer on maximizing the margin. From the analyses, we propose a novel method to mitigate the negative effect and provide stable performance by bridging the gap between clean and adversarial examples. [2]

3.2 Preliminaries

3.2.1 Adversarial Attack

White-box adversarial attack

Adversarial attacks generate imperceptible noise to make models incorrect (Szegedy et al., 2013). A classifier f attempts to classify an image $\boldsymbol{x}$ as a correct label y or a one-hot encoded vector $\boldsymbol{y}$. Adversarial attacks aim to find an adversarial perturbation $\boldsymbol{\delta}$ that maximizes a loss function ℓ with a maximum perturbation ϵ:

$$\max_{\boldsymbol{\delta} \in \mathcal{B}(0,\epsilon)} \ell(f(\boldsymbol{x}+\boldsymbol{\delta}), \boldsymbol{y}) \tag{3.1}$$

where $\mathcal{B}(\boldsymbol{x}, \epsilon)$ is the ϵ-ball around $\boldsymbol{x}$.

In white-box settings, an attacker can access the information of the model and generate adversarial examples by solving optimization problems or using the gradient of the input. Goodfellow et al. (2014) first proposed an effective gradient-based attack called fast gradient sign method (FGSM).

$$\boldsymbol{x}' = \boldsymbol{x} + \epsilon \cdot \mathrm{sgn}(\nabla_{\boldsymbol{x}} \ell(f(\boldsymbol{x}), \boldsymbol{y})) \tag{3.2}$$

where $\mathrm{sgn}(\cdot)$ represents the sign of each element. Madry et al. (2018) proposed an

[2] This work was preprinted in arXiv (Kim et al., 2021b).

improved attack called projected gradient descent (PGD). With a step size α, PGD can be formalized as follows:

$$\boldsymbol{x}^{t+1} = \Pi_{\mathcal{B}(\boldsymbol{x},\epsilon)}(\boldsymbol{x}^t + \alpha \cdot \mathrm{sgn}(\nabla_{\boldsymbol{x}} \ell(f(\boldsymbol{x}), \boldsymbol{y}))) \tag{3.3}$$

where $\Pi_{\mathcal{B}(\boldsymbol{x},\epsilon)}$ denotes the projection to $\mathcal{B}(\boldsymbol{x}, \epsilon)$. In addition, $\boldsymbol{x}^0$ is the original example $\boldsymbol{x}$. Given the number of steps s, $\boldsymbol{x}^s$ is used as the adversarial example $\boldsymbol{x}'$.

Carlini and Wagner (2017b) proposed several loss functions for better optimization, instead of cross-entropy loss. The Carlini and Wagner attack (CW) mainly uses the loss function as follows:

$$\ell(f(\boldsymbol{x}')) = -\max(\max_{i \neq y} f(\boldsymbol{x}')_i - f(\boldsymbol{x}')_y, 0) \tag{3.4}$$

With the above loss maximization, CW also achieves a high attack success rate for diverse model structure and datasets.

By adopting adversarial attacks in the vision domain, prior studies add inaudible noise to the waveform (Alzantot et al., 2018; Yuan et al., 2018) or perturb the audio feature (Cisse et al., 2017a; Kreuk et al., 2018) to deceive speech models. Iter et al. (2017) applied an FGSM (Goodfellow et al., 2014) to generate adversarial MFCC features. However, they used spectrograms and MFCC features as inputs, and thus a process for reconstructing the waveforms from spectrograms or an MFCC is additionally required, which is considered a difficult problem to overcome. To resolve this issue, Gong and Poellabauer (2017) proposed a gradient sign method that generates end-to-end audio adversarial examples based on an FGSM (Goodfellow et al., 2014). They suggested substituting recurrent neural networks with convolutional neural networks to prevent the vanishing gradient. The attack achieves a

high error rate in gender recognition, emotion recognition, and speaker recognition tasks. Carlini and Wagner (2018) achieved high attack performance by modifying the CW attack with a new constraint on the decibels (dB) instead of applying pixel bounding.

$$\begin{aligned} &\underset{\boldsymbol{\delta}}{\text{minimize}} \; ||\boldsymbol{\delta}|| + c \cdot \ell(f(\boldsymbol{x} + \boldsymbol{\delta}), \boldsymbol{y}) \\ &\text{such that } dB_{\boldsymbol{x}}(\boldsymbol{\delta}) < \tau \end{aligned} \tag{3.5}$$

where $dB(\boldsymbol{x}) = \max_i 20 \cdot \log_{10}(\boldsymbol{x}_i)$, $dB_{\boldsymbol{x}}(\boldsymbol{\delta}) = dB(\boldsymbol{\delta}) - dB(\boldsymbol{x})$, and τ is a hyper-parameter for adjusting the perturbation size. Following this line of work, many adversarial attacks successfully generate audio adversarial examples (Kreuk et al., 2018; Yakura and Sakuma, 2019; Yuan et al., 2018).

Black-box adversarial attack

In real-world application, the attacker usually can not access the target model. Obviously, compared to the case when the attacker have access to all information (i.e., *white-box settings*), the limitation on the information yields a lower attack success rate. Thus, recently, researchers also focuses on *gray-box settings* and *black-box settings* that the attacker does not know the completely information of the target model in consideration of the most realistic cases. There are two main types of black-box attacks: query-based attack and transfer attack.

Query-based attacks use outputs or predictions by requesting multiple queries. Based on the information of querying outputs, attackers approximate the gradient of the target model (Taori et al., 2019) or use genetic algorithms (Alzantot et al., 2018). However, this type of attacks have a limitation, because they require a number of queries that can be easily detected or restricted.

Transfer attacks are motivated by the observation that some adversarial examples can deceive multiple models even though they are generated from one specific model (Szegedy et al., 2013; Goodfellow et al., 2014). Transfer attacks first generate adversarial examples on a source model (or surrogate model) under the white-box settings. Then, it deceives a target model by feeding the adversarial examples. Here, the performance of a transfer attack is called *transferability*. Given a source model $h_S = f_S(g_S(\cdot))$ and a target model $h_T = f_T(g_T(\cdot))$, we measure the transferability $\mathcal{T}_{\mathcal{A},S\to T}$ as follows:

$$\mathcal{T}_{\mathcal{A},S\to T} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}}[\mathbf{1}\{\max_i h_T(\mathcal{A}(\boldsymbol{x},\boldsymbol{y};h_S))_i \neq y\}] \tag{3.6}$$

where $\mathbf{1}\{\cdot\}$ is an indicator function, and $\mathcal{A}$ is an adversarial attack. Here, $\mathcal{A}(\boldsymbol{x},\boldsymbol{y};h_S)$ is adversarial examples generated on the source model h_S. Thus, $\mathcal{T}_{\mathcal{A},S\to T}$ denotes the attack success rate of the attack $\mathcal{A}$ with the source model h_T on the target model h_T. Note that $\mathcal{T}_{\mathcal{A},S\to S}$ corresponds to the attack success rate of $\mathcal{A}$ on S in the white-box setting.

Low transferability indicates that the audio adversarial example generated cannot deceive diverse models that have different structures or parameters from the source model, which leads to disrupt estimating reliable robustness of models. If the target model shares a similar structure as the source model, a transfer attack shows high performance because they tend to have similar feature representation (Tramèr et al., 2017b).

To increase the transferability, there are a line of works in the vision domain such as using the attention network (Wu et al., 2020d), focusing on input diversity (Xie et al., 2019; Dong et al., 2019), using multiple ensemble models (Hang et al., 2020),

or finding universal perturbations (Li et al., 2021). Note that they achieved a high transferability over various datasets (e.g., CIFAR and ImageNet) and models (e.g., ResNets and Transformers) (Springer et al., 2021; Wang et al., 2022). However, the transferability of audio adversarial examples has not been as actively presented as in the vision domain (Gong and Poellabauer, 2017). Indeed, Yang et al. (2019) that audio adversarial examples generated by prior adversarial attacks (Alzantot et al., 2018; Yuan et al., 2018; Carlini and Wagner, 2018) show a poor transferability. Specifically, existing attack methods were mainly evaluated in the noiseless environment (Yuan et al., 2018; Yakura and Sakuma, 2019; Cisse et al., 2017a) or used limited models (Carlini and Wagner, 2018; Kreuk et al., 2018). Therefore, in this chapter, we conduct extensive experiments on the transferability of existing audio adversarial attack methods with various models in noiseless and noisy environment. Based on the analyses, we propose a new attack method that generates high transferable audio adversarial examples that can measure more reliable robustness of the audio systems.

3.2.2 Adversarial Defense

Since Szegedy et al. (2013) identified the existence of adversarial examples, most defenses are broken by adaptive attacks (Athalye et al., 2018a; Tramer et al., 2020) and the state-of-art performance is still observed from variants of adversarial training (Madry et al., 2018) and TRADES (Zhang et al., 2019b) utilizing the training tricks (Pang et al., 2020; Gowal et al., 2020), weight averaging (Wu et al., 2020b), and using more data (Carmon et al., 2019; Rebuffi et al., 2021).

We consider a c-class classification task with a neural network $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathbb{R}^c$. The network $f_{\boldsymbol{\theta}}$ classifies a sample $\boldsymbol{x} \in \mathcal{X}$ as $\arg\max_{i \in \mathcal{Y}} [f_{\boldsymbol{\theta}}(\boldsymbol{x})]_i$, where $\mathcal{Y} = \{0, \cdots, c-1\}$.

We denote the true label with respect to $\boldsymbol{x}$ by y and the corresponding one-hot representation by $\boldsymbol{y} \in \{0,1\}^c$. That is, $y_i = \mathbf{1}\{i = y\}, \forall i \in \mathcal{Y}$, with an indicator function $\mathbf{1}\{C\}$ which outputs 1 if the condition C is true and 0 otherwise. Then the probability function $\boldsymbol{p}_{\boldsymbol{\theta}} = \text{softmax} \circ f_{\boldsymbol{\theta}} : \mathcal{X} \to [0,1]^c$ outputs a c-dimensional probability vector whose elements sum to 1.

Given two probability vectors $\boldsymbol{p}, \boldsymbol{q}$ in the c-dimensional probability simplex, we define the following values: $H_{\boldsymbol{p}}(\boldsymbol{q}) = -\boldsymbol{p}^T \log \boldsymbol{q}$ and $\text{KL}(\boldsymbol{p}||\boldsymbol{q}) = \boldsymbol{p}^T \log \frac{\boldsymbol{p}}{\boldsymbol{q}}$. These are called the cross-entropy and Kullback-Leibler (KL) divergence between $\boldsymbol{p}$ and $\boldsymbol{q}$, respectively. In addition, we denote the entropy of $\boldsymbol{p}$ as $H(\boldsymbol{p}) = H_{\boldsymbol{p}}(\boldsymbol{p}) = -\boldsymbol{p}^T \log \boldsymbol{p}$. Note that for a one-hot vector $\boldsymbol{y} \in \{0,1\}^c$, $\text{KL}(\boldsymbol{y}||\boldsymbol{q}) = \sum_{i \in \mathcal{Y}} y_i \log \frac{y_i}{q_i} = -\boldsymbol{y}^T \log \boldsymbol{q}$ is equivalent to the well-known cross-entropy, $H_{\boldsymbol{y}}(\boldsymbol{q})$.

Adversarial Training (AT) (Madry et al., 2018) is one of the most effective defense methods. Given a perturbation set $\mathbb{B}(\boldsymbol{x}, \epsilon)$, which denotes a ball around an example $\boldsymbol{x}$ with a maximum perturbation ϵ, it encourages the worst-case probability output over the perturbation set $\mathbb{B}(\boldsymbol{x}, \epsilon)$ to directly match the label $\boldsymbol{y}$ by minimizing the following loss:

$$\begin{aligned} \ell_{AT}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) &= \max_{\boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \epsilon)} H_{\boldsymbol{y}}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}')) \\ &= \max_{\boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \epsilon)} \text{KL}(\boldsymbol{y}||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}')). \end{aligned} \tag{3.7}$$

TRADES (Zhang et al., 2019b) was proposed based on the analysis of the trade-off between adversarial robustness and standard accuracy. TRADES minimizes the

following loss:

$$\begin{aligned} &\ell_{TRADES}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) \\ =&\mathrm{KL}(\boldsymbol{y}||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})) + \beta \max_{\boldsymbol{x}' \in \mathbb{B}(\boldsymbol{x}, \epsilon)} \mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}')). \end{aligned} \tag{3.8}$$

where β is the regularization hyper-parameter. Here, the first term aims to maximize the margin of clean examples, while the second term encourages the model to be smooth.

To solve this highly non-concave optimization in Eq. (3.7) and Eq. (3.8), an iterative projected gradient descent (PGD) (Madry et al., 2018) with n steps is widely used:

$$\boldsymbol{x}^{t+1} = \Pi_{\mathbb{B}(\boldsymbol{x}, \epsilon)}\big(\boldsymbol{x}^t + \alpha \cdot \mathrm{sign}(\nabla_{\boldsymbol{x}} \ell_{inner}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}))\big) \text{ for } t = 1, \cdots, n-1 \tag{3.9}$$

where $\Pi_{\mathbb{B}(\boldsymbol{x}, \epsilon)}$ refers the projection to the $\mathbb{B}(\boldsymbol{x}, \epsilon)$ and α is a step-size for each step. Here, $\boldsymbol{x}^0$ is the original example and $\boldsymbol{x}^n$ is used an adversarial example $\boldsymbol{x}^*$. We denote this as PGD^n. For example, AT aims to minimize the loss in Eq. (3.7) so that $\mathrm{KL}(\boldsymbol{y}||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^t))$ is used as $\ell_{inner}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})$.

Although some provable defensive methods have proposed, such as certified defenses (Salman et al., 2019; Chen et al., 2020; Lee et al., 2021a), they are not our main focus here because they are figuratively orthogonal from the above adversarial training frameworks.

3.3 Methodology

3.3.1 Transferable Adversarial Examples

In this subsection, we first perform an extensive analysis on the transferability of audio adversarial examples with diverse models under both silent and noisy condi-

tions. We then explain the underlying relationship between transferability and noise sensitivity of audio adversarial examples with respect to the loss landscape. Based on the observation, we propose a new adversarial attack method, *noise injected attack*, that generates more transferable audio adversarial examples by adopting noise injection.

Transferability and noise sensitivity

We first perform a thorough evaluation on the transferability of audio adversarial examples with various models and environments. We begin with an experiment on the Google speech command dataset (Warden, 2018). The dataset consists of 65,000 one-second utterances by thousands of people. Following (Alzantot et al., 2018; Yang et al., 2019), we filtered the 10 most commonly used classes.[3]

For a thorough evaluation, we consider seven models with two different types of feature extractors: a frequency-based feature model and a Wav2vec-based model.

- **Frequency-based model.** A frequency-based feature extractor outputs Mel-spectrogram, which is the feature for an input waveform. Thus, a time-varying sequential waveform is converted to the mel-sacle frequencies through the Mel-spectrogram layer. We feed them directly to a neural network-based classifier following (Hannun et al., 2014; Amodei et al., 2016). For a classifier, we use DenseNet, VGG-19 and Wide-ResNet 28-10 (WRN).

- **Wav2vec-based model.** The encoder network of pre-trained Wav2vec (Schneider et al., 2019) outputs a representative vector as the feature for an input waveform. For a classifier, we construct a long short-term memory model with

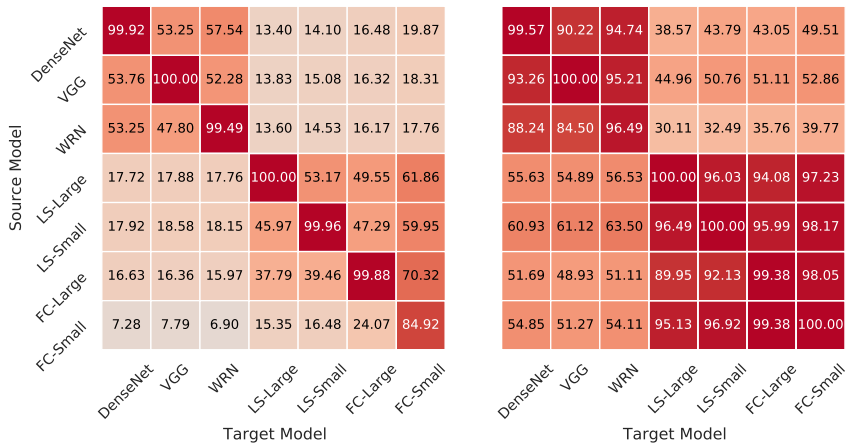
[3]https://www.kaggle.com/c/tensorflow-speech-recognition-challenge

self-attention followed by a fully connected layer. We call this type of models LS. The other type of classifier consists of only fully connected layers, which we call FC. We construct two different models for each type with different number of neurons. In total, four different classifiers were considered as Wav2vec-based models: LS-Large, LS-Small, FC-Large, and FC-Small.

For each trained model, following (Iter et al., 2017; Carlini and Wagner, 2018), we generated audio adversarial datasets with the most common baselines, PGD (Madry et al., 2018) and CW (Carlini and Wagner, 2018). We remark that almost adversarial attacks in the audio domain are the variants of PGD or CW. During attack process, we set the maximum perturbation ϵ to satisfy $dB_x(\boldsymbol{\delta}) = -20$ (Carlini and Wagner, 2018). Their step size α is fixed at $2\epsilon/s$, where the number of steps $s = 10$. Then, we estimate the transferability $\mathcal{T}_{\mathcal{A},S\to T}$ in Eq. (3.6) for each attack $\mathcal{A}$, source model S, and target model T.

First, we calculate the transferability $\mathcal{T}_{\mathcal{A},S\to T}$ under the noiseless setting. To do this, we directly feed generated audio adversarial datasets to the model without any noise, where the audio adversarial datasets is generated on the whole test dataset. Fig. 3.1 shows the transferability of the adversarial examples generated by each attack. Both CW and PGD show high white-box attack success rates over 95% for all types of feature extractors and structures as observed in prior work (Iter et al., 2017; Carlini and Wagner, 2018). Wav2vec-based models are also highly vulnerable to adversarial attacks because they use multi-layer convolutional neural networks. Here, we emphasize that this is the first study confirming the adversarial vulnerability of Wav2vec-based models.

However, both CW and PGD have low transferability (Yuan et al., 2018; Yang

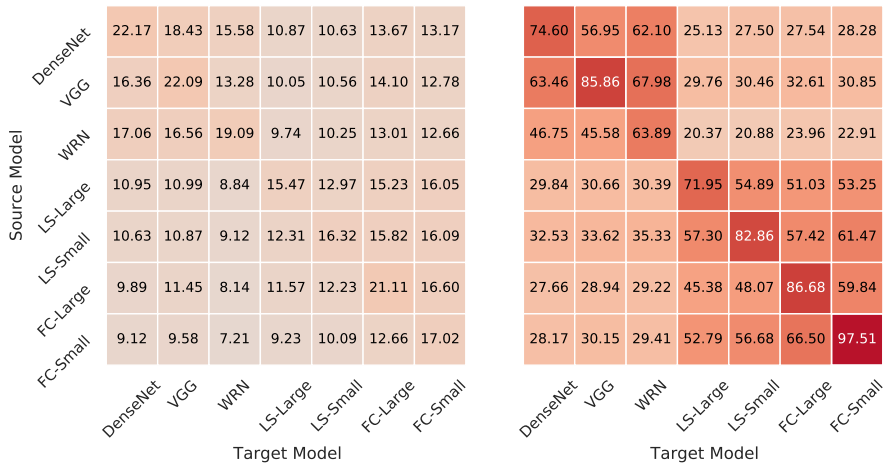


(a) CW (b) PGD

Figure 3.1: Transferability (%) of each adversarial attack from a source model (column) to a target model (row). Diagonal elements corresponds to the attack success rate under the white-box settings rather than transferability. A darker red square indicates a higher transferability. PGD shows higher transferability than CW.

et al., 2019), but with different characteristics. CW shows low transferability even for the same feature extractor-based models. $\mathcal{T}_{CW,DenseNet \to VGG}$ only shows 53.25%, although DenseNet and VGG share the same frequency-based feature extractor. This tendency is also observed in Wav2vec-based models. Moreover, for different feature extraction based models, the transferability significantly decreases. Specifically, $\mathcal{T}_{CW,DenseNet \to LS-Large}$ only shows 13.40%. By contrast, PGD achieves better transferability than CW. For example, $\mathcal{T}_{PGD,DenseNet \to FC-Large} = 43.05\%$, whereas $\mathcal{T}_{PGD,DenseNet \to WRN} = 94.74\%$. However, we note that the transferability to frequency-based models from LS-Large and FC-Large still shows the maximum value of 56.53%. In summary, both CW and PGD show limited transferability even under the noiseless setting, but PGD generates better transferable audio adversarial examples.

In many real-world scenarios, there is a high probability that the audio adversar-



(a) CW (b) PGD

Figure 3.2: Transferability (%) of each adversarial attack from a source model (column) to a target model (row) in the noisy environment. Diagonal elements corresponds to the attack success rate under the white-box settings rather than transferability. A darker red square indicates a higher transferability. PGD shows higher transferability than CW.

ial examples are played in the noisy environment (Foggia et al., 2015). Considering that the attack success rate of audio adversarial examples is generally degraded in noisy environment, it is thus imperative to estimate the transferability under the noise setting in the audio domain. Thus, we further investigate the transferability of each adversarial example after adding noise.

In this experiment, we simply add uniform noise $\boldsymbol{n} \sim \mathbf{U}[-\epsilon/5, \epsilon/5]$ to the adversarial example $\boldsymbol{x}'$ and evaluate the average transferability of $\boldsymbol{x}' + \boldsymbol{n}$ over three runs. As shown in Fig. 3.2, small uniform noise is sufficient to decrease the transferability of audio adversarial examples. Compared to Fig. 3.1, both CW and PGD show poor transferability. Specifically, CW fails to achieve the transferability of more than 20% . Moreover, the attack success rates of CW are also under 25%, which is significantly

reduced from the values under the noiseless setting Fig. 3.1a. In the case of the PGD, although slightly better than CW, it also shows a decreased transferability. The maximum value of transferabilty in Fig. 3.2b is $\mathcal{T}_{PGD,VGG\to WRN} = 67.98\%$, which was 95.21% in Fig. 3.1b. In addition, for different feature extractor-based models, the maximum transferabilty becomes $\mathcal{T}_{PGD,LS-Small\to WRN} = 35.33\%$, which is much worse than 63.50% in Fig. 3.1b. Thus, the transferability of existing audio adversarial examples are much poorer under the noisy environment, which is a common assumption in the real-world situation.

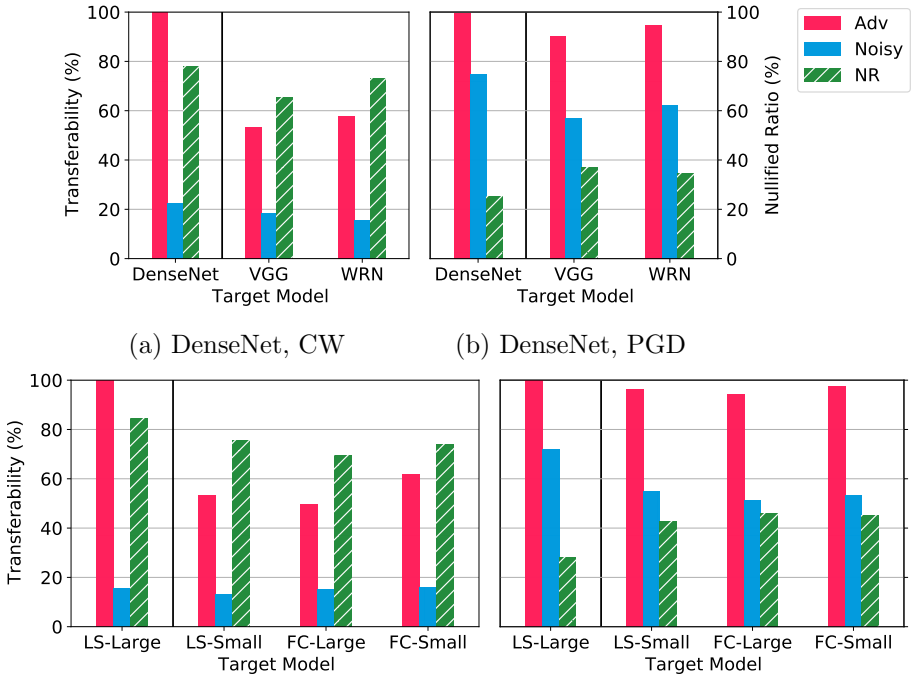
Interestingly, however, the reduced transferability of CW is more severe than PGD. This difference between CW and PGD is clear at a glance, as shown in Fig. 3.2a. Compared to Fig. 3.2b, all cells in Fig. 3.2a are yellow rather than red. the effect of adding noise to PGD is less than that of CW. To quantitatively compare the reduced transferability, we define a *nullified ratio (NR)*:

$$\text{NR} = \mathbb{E}[\mathbf{1}\{h(\boldsymbol{x}' + \boldsymbol{n}) = \boldsymbol{y}\} | h(\boldsymbol{x}') \neq \boldsymbol{y}] \tag{3.10}$$

Thus, NR measures the ratio of adversarial examples fail to deceive the model after adding noise among those which succeeded in deceiving the model under the noiseless setting.

The results are summarized in Fig. 3.3. The NR of CW exceeds 60% for all cases, while PGD shows the NR under 50%. Specifically, $\mathcal{T}_{CW,DenseNet\to VGG} = 53.25\%$ drops to 18.43% after adding additive noise, which corresponds to 65.39% of the NR much higher than that of PGD (25.08%). Thus, we can conclude that there CW is more sensitive to the noise than PGD in terms of transferability.

As the key observation in Fig. 3.1 and 3.3, the more transferable attack (PGD)



(a) DenseNet, CW (b) DenseNet, PGD

(c) LS-Large, CW (d) LS-Large, PGD

Figure 3.3: Transferability of adversarial examples $\boldsymbol{x}'$ (Adv) and noisy adversarial examples $\boldsymbol{x}' + \boldsymbol{n}$ (Noisy) with the nullified ratio (NR). For all cases, the NR of CW exceeds 60%, while PGD shows the NR under 50%. The title of each plot indicates the source model and the attack method. The first column corresponds to the source model so that Adv indicates the attack success rate under the white-box setting. The following groups are target models with the same feature extractor. All values are in percentage.

is hardly nullified by additive noise, while the less transferable attack (CW) is easily nullified by additive noise; that is, the more transferable attack is less sensitive to additive noise. Hence, we presume that there is a close connection between the transferability of audio adversarial examples and their sensitivity to additive noise.

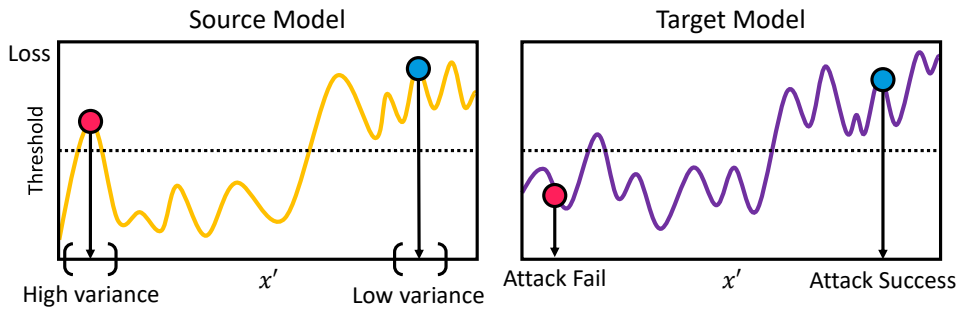


Figure 3.4: Illustration of transferability and the loss landscape. The adversarial example in a local optimum tends to have a low transferability (red), while the adversarial example that have a high loss in their vicinity tends to have a high transferability (blue) (Demontis et al., 2019). The variance of the loss value in its vicinity would be higher for red point than that of blue point.

Loss landscape analysis

Previously, we observed that more transferable are less sensitive to additive noise. To discover the underlying relationship between transferability and noise sensitivity, we provide an explanation on with respect to their loss landscape. As Demontis et al. (2019) argued, adversarial examples generated from a model with high variability of the loss landscape tend to get stuck in local optima, making them less transferable. We illustrate this phenomenon in Fig. 3.4. The red circle in a local optimum is difficult to be transferred to the target model due to the irregularity of the optimization. However, adversarial examples that have a high loss in their vicinity (blue circle) tend to be more transferable, and thus the target model also misclassifies the adversarial example by exceeding the decision threshold.

Now, we connect this loss landscape analysis to the noise sensitivity. If the adversarial example in the local optimum (red circle), it tends to have high variance of the loss value due to its variability. In contrast, when the adversarial example induces a

high loss in its vicinity (blue circle), it tends to have low variance of the loss value. Thus, considering that adding proper size of noise can provide the information of the loss value in its vicinity, less sensitive to additive noise (i.e., low variance of the loss in its vicinity for given noise) leads to more transferable audio adversarial examples.

Thus, a variance of the loss in the noise neighborhood of an adversarial example $\{\boldsymbol{x}+\boldsymbol{n}\}_{\boldsymbol{n}\sim\mathcal{N}}$ can be related to its transferability. To numerically verify this assumption, we propose a sample-wise variance (SV) to measure variability of the loss landscape of an adversarial example $\boldsymbol{x}'$ in its vicinity.

$$\mathrm{SV}(\boldsymbol{x}', \boldsymbol{y}) = \mathbb{E}_{\boldsymbol{n}\sim\mathcal{N}}[\ell(h(\boldsymbol{x}'+\boldsymbol{n}), \boldsymbol{y})^2] - \mathbb{E}_{\boldsymbol{n}\sim\mathcal{N}}[\ell(h(\boldsymbol{x}'+\boldsymbol{n}), \boldsymbol{y})]^2 \tag{3.11}$$

In other words, SV measures the variability of the loss landscape in the neighborhood by the additive noise $\boldsymbol{n}$. If the model has a high value of SV, then it implies that $\boldsymbol{x}'$ has a highly curved loss landscape in its vicinity (red circle in Fig. 3.4). On the contrary, a low SV indicates $\boldsymbol{x}'$ has a smoother loss landscape (blue circle in Fig. 3.4). We note that SV differs from the variance of the loss landscape in (Demontis et al., 2019), which estimates the model-wise variance of the loss landscape. By investigating the connection between the SV and transferability, we can experimentally prove that the more transferable, the less sensitive to additive noise.

Now, we verify the connection between the transferability and SV by controlling the number of steps. We note that manipulating the number of steps can easily adjust the transferability of an adversarial attack (Zhou et al., 2018). In addition, controlling the number of steps can rule out the effect of the inherited transferability of different adversarial attacks. In Fig. 3.5, we plot the transferability of adversarial examples and the expectation of SV ($\mathbb{E}[SV]$) for whole adversarial examples. To

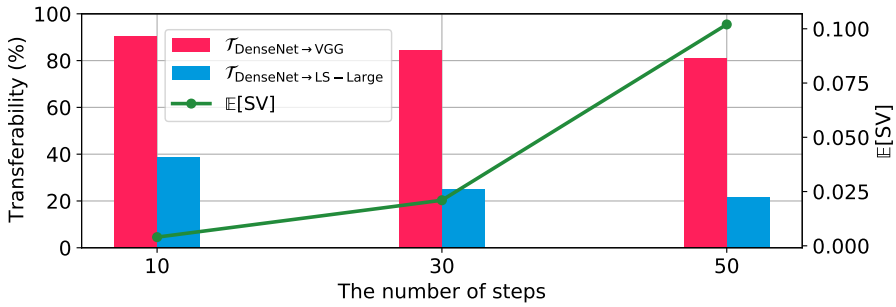


Figure 3.5: Transferability and SV of the source model. As the number of steps increases, the transferability decreases and the SV increases. Here, we used DenseNet as the source model and PGD as the attack method.

calculate SV, we sampled five additive noises $\boldsymbol{n} \sim \mathbf{U}[-\epsilon/5, \epsilon/5]$. As expected, we can observe a close connection between the transferability and SV of the source model. As the number of steps increases, the transferability decreases.

In summary, noise sensitivity of the adversarial example can be connected to its transferability in terms of the loss landscape in its neighborhood. A highly transferable adversarial example should have a high loss value in its noise neighborhood $\{\boldsymbol{x} + \boldsymbol{n}\}_{\boldsymbol{n}\sim\mathcal{N}}$, and thus it is less sensitive to additive noise.

Noise injected attack (NIA)

Based on the observation that more transferable audio adversarial examples are less sensitive to additive noise, we now propose a method to increase the transferability of audio adversarial examples. In order to generate a highly transferable adversarial example, we should enforce a high loss value in its vicinity. However, maximizing the loss of the neighborhood of adversarial example $\boldsymbol{x}$ is in general NP-hard. Thus, we instead increasing the loss of the noise neighborhood of the adversarial example $\{\boldsymbol{x} + \boldsymbol{n}\}_{\boldsymbol{n}\sim\mathcal{N}}$ by utilizing the gradient information of $\boldsymbol{x} + \boldsymbol{n}$.

The propose method, *noise injected attack (NIA)*, aims to increase not only the loss of adversarial examples but also the loss of its vicinity as follows:

$$\boldsymbol{x}^{t+1} = \Pi_{\mathcal{B}(\boldsymbol{x},\epsilon)}(\boldsymbol{x}^t + \alpha \cdot \mathrm{sgn}(\nabla_{\boldsymbol{x}^t} \ell(h(\boldsymbol{x}^t + \boldsymbol{n}^t), \boldsymbol{y}))) \tag{3.12}$$

where $\boldsymbol{n}^t$ is a noise drawn from an appropriate noise distribution $\mathcal{N}$ for each step t. The gradient obtained by $\ell(h(\boldsymbol{x}^t + \boldsymbol{n}^t), \boldsymbol{y})$ induces the increase of the loss $h(\boldsymbol{x}^t + \boldsymbol{n}^t)$, where $\boldsymbol{x}^t + \boldsymbol{n}^t$ is in the neighborhood of the adversarial example $\boldsymbol{x}^t$. Due to the randomness of $\boldsymbol{n}^t \sim \mathcal{N}$ for each step t, the noise neighborhood of adversarial example $\{\boldsymbol{x} + \boldsymbol{n}\}_{\boldsymbol{n}\sim\mathcal{N}}$ tends to have a high loss over the entire area during the gradient ascent process. Thus, the optimization is expected to increase the transferability by increasing the loss of the neighborhood of the adversarial example for given noise distribution so that it has low SV.

To push further, we provide a theoretical analysis of the proposed method in terms of maximizing the transferability. Given a source model h_S, existing adversarial attacks try to find the solution:

$$\max_{\boldsymbol{\delta}\in\mathcal{B}(0,\epsilon)} \ell(h_S(\boldsymbol{x} + \boldsymbol{\delta}), \boldsymbol{y}) \tag{3.13}$$

However, the optimal point can be stuck in the local optimum, which leads to a poor transferability. Thus, by generating an adversarial example $\boldsymbol{x}'$ where $\boldsymbol{x}' + \boldsymbol{n}$ in its neighborhood also has high loss values, we can expect to find a more transferable adversarial example. This can be achieved by the proposed method, which is formalized as follows:

$$\max_{\boldsymbol{\delta}\in\mathcal{B}(0,\epsilon)} \mathbb{E}_{\boldsymbol{n}\sim N} \ell(h_S(\boldsymbol{x} + \boldsymbol{\delta} + \boldsymbol{n}), \boldsymbol{y}) \tag{3.14}$$

The optimization yields the adversarial example where the set of its neighborhood $\boldsymbol{x}' + \boldsymbol{n}$ is also in a high-loss region for any random direction $\boldsymbol{n}$.

Then, under some mild conditions on the hypothesis space $\mathcal{H}$ as in (Madry et al., 2018; Wong et al., 2020; Kim et al., 2022a), Eq. (3.14) can be shown to be more aligned with the optimization for maximizing transferability given by

$$\max_{\delta \in \mathcal{B}(0,\epsilon)} \mathbb{E}_{h \sim \mathcal{H}} \ell(h(\boldsymbol{x} + \boldsymbol{\delta}), \boldsymbol{y}) \tag{3.15}$$

which motivates the proposed method can yield adversarial examples with a higher loss for a target model than the existing methods.

In addition, we investigate the fundamental differences between the proposed method and the common techniques to generate adversarial examples. Here, we compare our approach with several related lines of research in the prior literature.

Random initialization The random initialization during the first step of an adversarial attack is a common practice (Madry et al., 2018). It was also recently revealed that the random initialization in adversarial attack is a key to achieving robustness in adversarial training (Wong et al., 2020; Kim et al., 2021a). However, NIA injects different noise at every step, while existing adversarial attacks only add noise in the first step of the attack process.

Input transformation Some adversarial attacks in the vision domain tried to increase the transferability (Xie et al., 2019; Dong et al., 2019). Recently, Xie et al. (2019) proposed diverse inputs iterative fast gradient sign method (DIM) to boost the transferability of adversarial images. Although DIM mainly uses resize and padding to generate diverse inputs at every step, NIA is different in that it uses

adding noise to the input waveform at every step instead of spatial transformations to the input image. Similarly, Dong et al. (2019) also proposed translation-invariant method (TIM) based on the convolution operation on gradients with a kernel. However, it also depends on the spatial characteristic of images so that NIA shows better performance within the audio domain than DIM and TIM.

3.3.2 Improved Adversarial Training

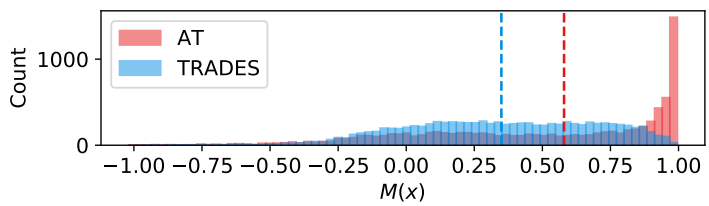
Understanding Margin and Smoothness in Adversarial Training

Similar robustness, but different margin and smoothness To illustrate the difference between AT and TRADES in terms of margin and smoothness, we first define a measure for margin and smoothness. To estimate margin, we adopt the probabilistic margin $M(\cdot)$ that is commonly used by prior studies (Carlini and Wagner, 2017b; Gowal et al., 2020; Liu et al., 2021a):

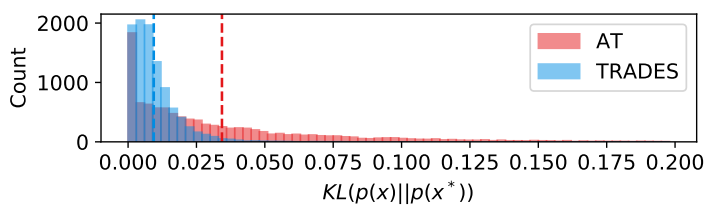
$$M(\boldsymbol{x}) := [\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})]_y - \max_{i \neq y} [\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})]_i \tag{3.16}$$

Thus, $M(\boldsymbol{x}) > 0$ indicates that the model correctly predicts the label of $\boldsymbol{x}$. On the contrary, the model outputs a wrong prediction when $M(\boldsymbol{x}) < 0$. To estimate smoothness, we use $\mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}) || \boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$ in (Zhang et al., 2019b), where $\boldsymbol{x}^*$ is an adversarial example of a clean example $\boldsymbol{x}$.

Fig. 3.6 illustrates the difference between AT and TRADES in terms of margin and smoothness. First, we trained models with the maximum perturbation $\epsilon = 8/255$ on CIFAR10. Then, for each model, we measured the margin and smoothness on an adversarial dataset $(\boldsymbol{x}^*, \boldsymbol{y})$ generated by using PGD^{50} with the same maximum perturbation $\epsilon = 8/255$. At the end of the training, although AT and TRADES have similar robustness (53.94% and 52.98%), they show totally different characteristics.



(a) Margin



(b) Smoothness

Figure 3.6: (CIFAR10) Margin and smoothness of AT and TRADES. (a) $M(\boldsymbol{x})$ for estimating margin (higher is better). (b) $\mathrm{KL}(\boldsymbol{p}_\theta(\boldsymbol{x})||\boldsymbol{p}_\theta(\boldsymbol{x}^*))$ for estimating smoothness (lower is better). Each vertical line indicates the average value of each measure. Each plot used 10,000 test examples.

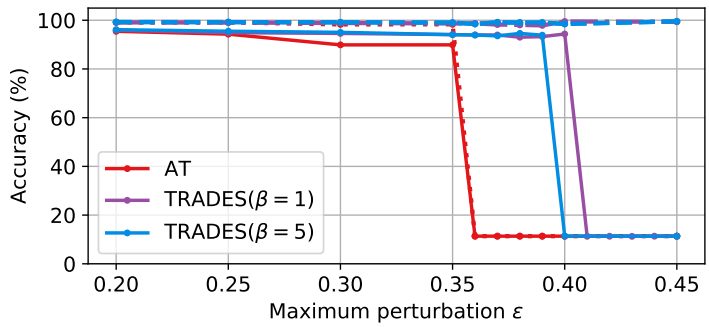


Figure 3.7: (MNIST) Stability of each method for a wide range of the maximum perturbation ϵ during training. Each model is trained on the given maximum perturbation ϵ and evaluated by using PGD^{50} with the same ϵ used during training. The dotted and solid lines indicate the clean accuracy and robust accuracy, respectively.

AT shows a larger margin that is distributed close to 1, whereas it has a poor smoothness than TRADES. On the contrary, TRADES shows a smaller margin with only a few examples around 1, whereas it has a better smoothness than AT. The vertical lines (the average values of their margin and smoothness) also show the difference between AT and TRADES. Thus, we can conclude that AT and TRADES have different characteristics in terms of margin and smoothness.

Same training settings, different optimization difficulty To push further, we evaluate their clean and robust accuracy under various maximum perturbations during the training on MNIST. As shown in Fig. 3.7, AT fails to achieve both standard and robust accuracy for $\epsilon > 0.35$. This is consistent with the observation that the regularization term for maximizing the margin of adversarial examples has some drawbacks in convergence (Shaeiri et al., 2020; Liu et al., 2020; Dong et al., 2021; Sitawarin et al., 2020; Shaeiri et al., 2020). In contrast, even for $\epsilon > 0.35$,

TRADES holds its robustness better than AT. Interestingly, TRADES shows high clean accuracy for $\epsilon \geq 0.4$ even it fails to gain robustness. However, TRADES also fails to achieve the robustness for $\epsilon \geq 0.4$. The result is similar even with a smaller weight ($\beta = 1$) on $\mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$.

To explain this phenomenon, we take a closer look at the regularization terms of AT and TRADES. First, AT directly increases the margin of adversarial examples $\boldsymbol{x}^*$ as in Eq. (3.7) by minimizing $\mathrm{KL}(\boldsymbol{y}||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$. However, due to lack of considering the connection between $\boldsymbol{x}$ and $\boldsymbol{x}^*$, AT has difficulty attaining smoothness, which is observed in Fig. 3.6. In addition, minimizing $\mathrm{KL}(\boldsymbol{y}||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$ yields the drawbacks in convergence (Shaeiri et al., 2020; Liu et al., 2020; Dong et al., 2021; Sitawarin et al., 2020; Shaeiri et al., 2020), which can be observed in Fig. 3.7. In contrast, TRADES adopts the regularization term $\mathrm{KL}(\boldsymbol{y}||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}))$ and $\mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$ in Eq. (3.8) instead of $\mathrm{KL}(\boldsymbol{y}||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$. Due to this loss function, TRADES shows much more stable performance than AT for $\epsilon < 0.4$ as shown in Fig. 3.7. However, TRADES still fails to optimize $\mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$ for $\epsilon \geq 0.4$, while it successfully optimizes $\mathrm{KL}(\boldsymbol{y}||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}))$. This implies that TRADES has trouble converging to the optimal point during maximizing the margin and minimizing the KL divergence simultaneously.

Negative effect of smoothness regularizer on maximizing the margin The degraded margin of TRADES and its failure cases for large perturbations lead us to postulate the hypothesis that there is a conflict between $\mathrm{KL}(\boldsymbol{y}||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}))$ and $\mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$. Now, we mathematically prove that the regularizer for smoothness $\mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$ in Eq. (3.8) has a negative effect on training a large margin.

For simplicity, we first consider the binary case. Without loss of generality, we may assume that the correct label for $\boldsymbol{x}$ is $y = 0$ and $\boldsymbol{p}_\theta(\boldsymbol{x}) = [p, 1-p]$. Then, the margin becomes $M(\boldsymbol{x}) = 2p - 1$. Under this binary case, the following proposition holds.

Proposition 3.1. *Let $\boldsymbol{p} = \boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})$, $\boldsymbol{p}^* = \boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*)$, $\nabla = \nabla_{\boldsymbol{\theta}}$. Assume the binary case, where $\boldsymbol{p} = [p, 1-p]$ and $\boldsymbol{p}^* = [p^*, 1-p^*]$. Then, $-\nabla KL(\boldsymbol{p}||\boldsymbol{p}^*)$ has a gradient direction opposite to $\nabla M(\boldsymbol{x})$.*

$$\nabla KL(\boldsymbol{p}||\boldsymbol{p}^*) = \frac{1}{2}\big(\log\frac{p}{p^*} - \log\frac{1-p}{1-p^*}\big)\cdot\nabla M(\boldsymbol{x}) + \boldsymbol{c}$$

Here, $\boldsymbol{c}$ is a linear combination of other gradient directions. Since $p > p^*$, $\log\frac{p}{p^*} - \log\frac{1-p}{1-p^*}$ is always positive. Thus, minimizing $KL(\boldsymbol{p}||\boldsymbol{p}^*)$ hinders the increase in $M(\boldsymbol{x})$.

Proof. First, $\nabla\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$ can be formalized as follows:

$$\begin{aligned}
\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*) &= p\log p + (1-p)\log(1-p) - p\log p^* - (1-p)\log(1-p^*)\\
\nabla\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*) &= \frac{\partial\mathrm{KL}}{\partial p}\cdot\nabla p + \frac{\partial\mathrm{KL}}{\partial p^*}\cdot\nabla p^*\\
&= \big(\log\frac{p}{p^*} - \log\frac{1-p}{1-p^*}\big)\cdot\nabla p + \big(-\frac{p}{p^*} + \frac{1-p}{1-p^*}\big)\cdot\nabla p^*\\
&= \big(\log\frac{p}{p^*} - \log\frac{1-p}{1-p^*}\big)\cdot\nabla p + \boldsymbol{c}
\end{aligned}$$

In addition, $\nabla M(\boldsymbol{x}) = 2\nabla p$, since $M(\boldsymbol{x}) = p - (1-p) = 2p - 1$. Thus,

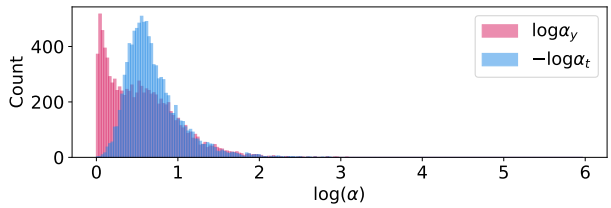
$$\nabla\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*) = \frac{1}{2}\big(\log\frac{p}{p^*} - \log\frac{1-p}{1-p^*}\big)\cdot\nabla M(\boldsymbol{x}) + \boldsymbol{c}.$$

□

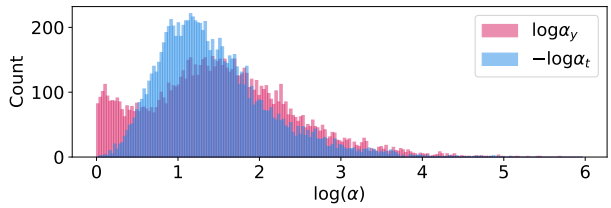
We can easily extend the binary case to the multi-class problem by introducing an element-wise division vector $\boldsymbol{\alpha} = \boldsymbol{p}/\boldsymbol{p}^*$, i.e., the i-th element of $\boldsymbol{\alpha}$ is $\alpha_i = p_i/p_i^*$.

Proposition 3.2. *Let the correct label y, $t = \arg\max_{i\neq y} p_i^*$ and $\alpha_i = p_i/p_i^*$. Assume that $p_y > p_y^*$ and $p_t < p_t^*$. Then, $-\nabla KL(\boldsymbol{p}||\boldsymbol{p}^*)$ is aligned with a gradient direction that penalizes p_i with the scale of $\log\alpha_i$, which minimizes the margin $M(\boldsymbol{x})$.*

$$\nabla KL(\boldsymbol{p}||\boldsymbol{p}^*) = (\nabla\boldsymbol{p})^T\log\boldsymbol{\alpha} + \boldsymbol{c}$$



(a) $\log \alpha_i$ for $\epsilon = 8/255$



(b) $\log \alpha_i$ for $\epsilon = 16/255$

Figure 3.8: (CIFAR10) Distribution of $\log \alpha_y = \log(p_y/p_y^*)$ for the true class y (red) and $-\log \alpha_t = -\log(p_t/p_t^*)$ for the target class $t = \arg\max_{i \neq y} p_i^*$ (blue). Each plot shows the distribution of $\log \alpha_y$ and $\log \alpha_t$ for whole test adversarial examples generated with different maximum perturbation ϵ. $\log \alpha_y$ is always positive and $\log \alpha_t$ is always negative. In addition, for a larger ϵ during the attack process, both $|\log \alpha_y|$ and $|\log \alpha_t|$ have a larger deviation from 0.

Here, $\boldsymbol{c}$ is a linear combination of other gradient directions. By the assumption, $\log \alpha_y > 0$ and $\log \alpha_t < 0$ so that $-(\nabla p_y)^T \log \alpha_y - (\nabla p_t)^T \log \alpha_t$ minimizes the margin $M(\boldsymbol{x})$.

Proof.

$$
\begin{aligned}
&\nabla \mathrm{KL}(\boldsymbol{p} || \boldsymbol{p}^*) \\
=&\nabla \left(\boldsymbol{p}^T \log \boldsymbol{p} - \boldsymbol{p}^T \log \boldsymbol{p}^* \right) \\
=&(\nabla \boldsymbol{p})^T \log \boldsymbol{p} + (\nabla \boldsymbol{p})^T \mathbf{1} - (\nabla \boldsymbol{p})^T \log \boldsymbol{p}^* - (\nabla \boldsymbol{p}^*)^T \frac{\boldsymbol{p}}{\boldsymbol{p}^*} \\
=&(\nabla \boldsymbol{p})^T \log \frac{\boldsymbol{p}}{\boldsymbol{p}^*} - (\nabla \boldsymbol{p}^*)^T \frac{\boldsymbol{p}}{\boldsymbol{p}^*} \\
=&(\nabla \boldsymbol{p})^T \log \boldsymbol{\alpha} - (\nabla \boldsymbol{p}^*)^T \boldsymbol{\alpha}.
\end{aligned} \tag{3.17}
$$

which leads to the conclusion with $\boldsymbol{c} = -(\nabla \boldsymbol{p}^*)^T \boldsymbol{\alpha}$ □

Note that the assumption, $p_y > p_y^*$ and $p_t < p_t^*$, is generally acceptable under

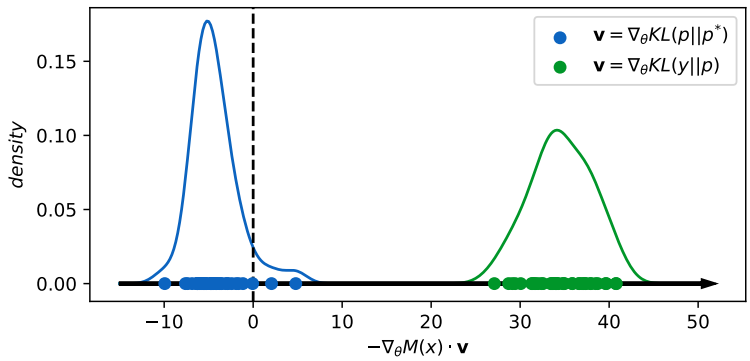


Figure 3.9: (CIFAR10) Projected gradients of each loss term of TRADES in Eq. (3.8) on the direction of increasing the margin. At the end of each epoch, we calculated $\nabla \mathrm{KL}(\boldsymbol{y}||\boldsymbol{p})$ and $\nabla \mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$, and $\nabla M(\boldsymbol{x})$. Then, we plotted their inner products. Minimizing the regularization term $\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$ (blue points) has a negative effect on maximizing the margin.

the characteristic of adversarial attack, which reduces p_y and increase $p_{i \neq y}$. We empirically proved this assumption in Fig. 3.8. From the model trained with TRADES under $\epsilon = 8/255$, we generated an adversarial dataset $(\boldsymbol{x}^*, \boldsymbol{y})$ from the whole test dataset by using PGD^{50} with the same maximum perturbation $\epsilon = 8/255$. Then, we plot the distribution of $\log \alpha_i$ for y and $t = \arg\max_{i \neq y} p_i^*$. As shown in Fig. 3.8, $\log \alpha_y$ is always positive and $\log \alpha_t$ is always negative for all test examples. Thus, we can conclude that the regularization term for smoothness $\mathrm{KL}(\boldsymbol{p_\theta}(\boldsymbol{x})||\boldsymbol{p_\theta}(\boldsymbol{x}^*))$ hinders the model from maximizing the margin.

Now, to provide empirical evidence of the negative effect, we visualize the effect of each loss term of TRADES on the margin in Fig. 3.9. The x-axis denotes the gradient direction that maximizes the margin, $\nabla M(\boldsymbol{x})$. The blue and green points represent the effect of each term's gradients, $\nabla \mathrm{KL}(\boldsymbol{y}||\boldsymbol{p})$ and $\nabla \mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$, on maximizing the margin. While the gradient descent direction of $\mathrm{KL}(\boldsymbol{y}||\boldsymbol{p})$ is aligned with $\nabla M(\boldsymbol{x})$,

the gradient descent of the other term $\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$ is in the opposite direction. Thus, the result confirms that minimizing the regularization term $\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$ has a negative effect on maximizing the margin.

Moreover, we prove that this negative effect gets more pronounced as the maximum perturbation ϵ increases. This can be otherwise said, if $\epsilon_1 < \epsilon_2$, then $\mathbb{B}(\boldsymbol{x}, \epsilon_1) \subset \mathbb{B}(\boldsymbol{x}, \epsilon_2)$ so that

$$\max_{\boldsymbol{x}_1^* \in \mathbb{B}(\boldsymbol{x}, \epsilon_1)} |\log[\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})/\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}_1^*)]_i| \leq \max_{\boldsymbol{x}_2^* \in \mathbb{B}(\boldsymbol{x}, \epsilon_1)} |\log[\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})/\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}_2^*)]_i| \tag{3.18}$$

for $i = y, t$. Thus, minimizing $\mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$ with a larger ϵ comes with a larger $|\alpha_i| = |[\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})/\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*)]_i|$. In summary, because the negative effect is proportional to the scale of $\log \alpha_i$ as proved in Proposition 3.2, a larger ϵ leads to a prohibitive negative effect on maximizing the margin.

Indeed, if we set a larger maximum perturbation $\epsilon = 16/255$, the values of $\log \alpha_y$ and $\log \alpha_t$ tends to have a larger deviation from 0 than that of $\epsilon = 8/255$ (see Figure 3.8). This is consistent with our observation that TRADES suffers the convergence problem and fails to achieve decent robustness for a larger perturbation in Fig. 3.7 and Section 3.4.2, respectively. Thus, from the above observations and analyses, we expect the model to converge to a better local minimum by mitigating the negative effect.

We theoretically and experimentally confirmed that there exists the negative effect of the smoothness regularizer on maximizing the margin. Now, we enable the model to converge to a better optimal point by mitigating the negative effect.

Bridged Adversarial Training

Mitigating the negative effect by bridging To mitigate the negative effect of $\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$ on maximizing the margin, we have to minimize the absolute value of $\log \boldsymbol{\alpha}$ as proved in Proposition 3.2. The key idea to control the absolute value of $\log \boldsymbol{\alpha}$ is bridging the gap between $\boldsymbol{p}$ and $\boldsymbol{p}^*$. Let us consider a new probability vector $\tilde{\boldsymbol{p}}$. If we use a new loss function $\mathrm{KL}(\boldsymbol{p}||\tilde{\boldsymbol{p}}) + \mathrm{KL}(\tilde{\boldsymbol{p}}||\boldsymbol{p}^*)$ instead of $\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$, the gradient of $\mathrm{KL}(\boldsymbol{p}||\tilde{\boldsymbol{p}}) + \mathrm{KL}(\tilde{\boldsymbol{p}}||\boldsymbol{p}^*)$ can be formalized as follows:

$$\begin{aligned}
&\nabla(\mathrm{KL}(\boldsymbol{p}||\tilde{\boldsymbol{p}}) + \mathrm{KL}(\tilde{\boldsymbol{p}}||\boldsymbol{p}^*)) \\
&\quad = (\nabla \boldsymbol{p})^T \log \boldsymbol{\alpha}^{(1)} + (\nabla \tilde{\boldsymbol{p}})^T \left(-\boldsymbol{\alpha}^{(1)} + \log \boldsymbol{\alpha}^{(2)}\right) - (\nabla \boldsymbol{p}^*)^T \boldsymbol{\alpha}^{(2)} \qquad (3.19)
\end{aligned}$$

where $\alpha_i^{(1)} = p_i/\tilde{p}_i$ and $\alpha_i^{(2)} = \tilde{p}_i/p_i^*$.

As shown in Eq. (3.19), $\nabla \boldsymbol{p}$ is now effected by $\log \boldsymbol{\alpha}^{(1)} = \log(\boldsymbol{p}/\tilde{\boldsymbol{p}})$ instead of $\log \boldsymbol{\alpha} = \log(\boldsymbol{p}/\boldsymbol{p}^*)$. Thus, now we can control the degree of the negative effect by introducing $\tilde{\boldsymbol{p}}$, while achieving the smoothness. For example, if $\tilde{\boldsymbol{p}}$ satisfies $p_y > \tilde{p}_y > p_y^*$ for a correct label y and $p_i < \tilde{p}_i < p_i^*$ for all $i \neq y$, the negative effect can be reduced because $|\log \alpha_i^{(1)}| < |\log \alpha_i|$ for all i. Considering this property, we name a probability vector $\tilde{\boldsymbol{p}}$ that reduces the negative effect as an *intermediate probability*.

In summary, minimizing the new loss $\mathrm{KL}(\boldsymbol{p}||\tilde{\boldsymbol{p}}) + \mathrm{KL}(\tilde{\boldsymbol{p}}||\boldsymbol{p}^*)$ can provide the smoothness between $\boldsymbol{p}$ and $\boldsymbol{p}^*$ with the reduced negative effect on maximizing the margin by introducing an intermediate probability $\tilde{\boldsymbol{p}}$ as a bridge. Thus, we name a new adversarial training method, which minimizes $\mathrm{KL}(\boldsymbol{p}||\tilde{\boldsymbol{p}}) + \mathrm{KL}(\tilde{\boldsymbol{p}}||\boldsymbol{p}^*)$ instead of $\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$, *bridged adversarial training (BAT)*.

Intuitively, more than one intermediate probability can induce the less negative effect of $\mathrm{KL}(\boldsymbol{p}||\tilde{\boldsymbol{p}})$. For a given sample $\boldsymbol{x}$, let $\gamma : [0, 1] \to \mathcal{X}$ be a continuous path from

Algorithm 2: Generalized Bridged Adversarial Training

Input: training data, $\mathcal{D}$; a continuous path, $\gamma(\cdot)$; a model parameter, $\boldsymbol{\theta}$; a maximum perturbation, ϵ; an adversarial attack, $\mathcal{A}_{\boldsymbol{\theta},\epsilon} : \mathcal{X} \times \mathcal{Y} \to \mathcal{X}$; a hyper-parameter for regularizer, β; the number of bridges, m.

for $(\boldsymbol{x}, \boldsymbol{y}) \sim \mathcal{D}$ **do**

$\boldsymbol{x}^* \leftarrow \mathcal{A}_{\boldsymbol{\theta},\epsilon}(\boldsymbol{x}, \boldsymbol{y})$ $\gamma(0) \leftarrow \boldsymbol{x}$ and $\gamma(1) \leftarrow \boldsymbol{x}^*$ $\ell \leftarrow \mathrm{KL}(\boldsymbol{y}||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}))$ $+ \sum_{k=0}^{m-1} \beta \mathrm{KL}\big(\boldsymbol{p}_{\boldsymbol{\theta}}(\gamma(\frac{k}{m}))||\boldsymbol{p}_{\boldsymbol{\theta}}(\gamma(\frac{k+1}{m}))\big)$ $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \nabla_{\boldsymbol{\theta}}\ell$

end

$\gamma(0) = \boldsymbol{x}$ to $\gamma(1) = \boldsymbol{x}^*$, where $\boldsymbol{x}^*$ is an adversarial example of $\boldsymbol{x}$. Now, we minimize the bridged loss $\sum_{k=0}^{m-1} \mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\gamma(\frac{k}{m}))||\boldsymbol{p}_{\boldsymbol{\theta}}(\gamma(\frac{k+1}{m})))$ instead of $\mathrm{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$. Here, m is a hyper-parameter for the number of intermediate probabilities. The generalized bridged adversarial training procedure is presented in Algorithm 2.

Bound on the robust error Here, we provide theoretical evidence that the proposed loss serves as an upper bound on the robust error of the model under the binary classification setting. To give a self-contained overview, we follow (Zhang et al., 2019b).

To give a self-contained overview, we follow (Bartlett et al., 2006). In the binary classification case, given a sample $\boldsymbol{x} \in \mathcal{X}$ and a label $y \in \{-1, 1\}$, a model can be denoted as $f : \mathcal{X} \to \mathbb{R}$. We use $\mathrm{sign}(f(\boldsymbol{x}))$ as a prediction value of y. Given a surrogate loss ϕ, the conditional ϕ-risk for $\eta \in [0, 1]$ can be denoted as $H(\eta) := \inf_{\alpha \in \mathbb{R}}(\eta\phi(\alpha)+(1-\eta)\phi(-\alpha))$. Similarly, we can define $H^-(\eta) := \inf_{\alpha(2\eta-1)\leq 0}(\eta\phi(\alpha)+(1-\eta)\phi(-\alpha))$. Now, we assume the surrogate loss ϕ is classification-calibrated, so that $H^-(\eta) > H(\eta)$ for any $\eta \neq 1/2$. Then, the ψ-transform of a loss function ϕ, which is the convexified version of $\hat{\psi}(\theta) = H^-(\frac{1+\theta}{2}) - H(\frac{1+\theta}{2})$, is continuous convex function on $\theta \in [-1, 1]$.

In the adversarial training framework, we train a model to reduce the robust error $\mathcal{R}_{rob}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\exists \boldsymbol{x'} \in \mathbb{B}(\boldsymbol{x},\epsilon) \text{ s.t. } f(\boldsymbol{x'})y \leq 0\}$ where $\mathbb{B}(\boldsymbol{x},\epsilon)$ is a ball around an example x with a maximum perturbation ϵ. Here, $\mathbf{1}\{C\}$ denotes an indicator function which outputs 1 if the condition C is true and 0 otherwise. As (Zhang et al., 2019b) proposed, $\mathcal{R}_{rob}(f)$ can be decomposed as follows:

$$\mathcal{R}_{rob}(f) = \mathcal{R}_{nat}(f) + \mathcal{R}_{bdy}(f) \tag{3.20}$$

where the natural classification error $\mathcal{R}_{nat}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{f(\boldsymbol{x})y \leq 0\}$ and boundary error $\mathcal{R}_{bdy}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{f(\boldsymbol{x})y > 0, \exists \boldsymbol{x'} \in \mathbb{B}(\boldsymbol{x},\epsilon) \text{ s.t. } f(\boldsymbol{x})f(\boldsymbol{x'}) \leq 0\}$. By definition, following inequality is satisfied:

$$\begin{aligned}
\mathcal{R}_{bdy}(f) &= \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\boldsymbol{x} \in \mathbb{B}(\mathrm{DB}(f),\epsilon), f(\boldsymbol{x})y > 0\} \\
&\leq \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\boldsymbol{x} \in \mathbb{B}(\mathrm{DB}(f),\epsilon)\} \\
&= \mathbb{E} \max_{\boldsymbol{x'} \in \mathbb{B}(\boldsymbol{x},\epsilon)} \mathbf{1}\{f(\boldsymbol{x'}) \neq f(\boldsymbol{x})\} \\
&= \mathbb{E} \max_{\boldsymbol{x'} \in \mathbb{B}(\boldsymbol{x},\epsilon)} \mathbf{1}\{\beta f(\boldsymbol{x'})f(\boldsymbol{x}) < 0\}.
\end{aligned} \tag{3.21}$$

Let, $\mathcal{R}^{\star}_{nat} := \inf_f \mathcal{R}_{nat}(f)$ and $\mathcal{R}^{\star}_{\phi} := \inf_f \mathcal{R}_{\phi}(f)$ where $\mathcal{R}_{\phi}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\phi\{f(\boldsymbol{x})y \leq 0\}$ is a surrogate loss with a surrogate loss function ϕ. Then, under Assumption 1, following inequality is satisfied by Eq. (3.20) and Eq. (3.21).

$$\mathcal{R}_{rob}(f) - \mathcal{R}^{\star}_{nat} \leq \psi^{-1}(\mathcal{R}_{\phi}(f) - \mathcal{R}^{\star}_{\phi}) + \mathbb{E} \max_{\boldsymbol{x'} \in \mathbb{B}(\boldsymbol{x},\epsilon)} \mathbf{1}\{\beta f(\boldsymbol{x'})f(\boldsymbol{x}) < 0\} \tag{3.22}$$

Given example $\boldsymbol{x}$, adversarial example $\boldsymbol{x}^*$ and a continuous path $\gamma(\cdot)$ such that $\gamma(0) = \boldsymbol{x}$ and $\gamma(1) = \boldsymbol{x}^*$, following inequality holds:

$$\mathbf{1}\{\beta f(\boldsymbol{x}^*)f(\boldsymbol{x}) < 0\} \leq \sum_{k=0}^{m-1} \mathbf{1}\{\beta f(\gamma(\frac{k}{m}))f(\gamma(\frac{k+1}{m})) \leq 0\} \tag{3.23}$$

where m is a hyper-parameter for dividing the path $\gamma(\cdot)$.

Proof of Eq. (3.23). First, it is clear when $f(\boldsymbol{x}^*)f(\boldsymbol{x}) \geq 0$. Thus, we consider the case of $f(\boldsymbol{x}^*)f(\boldsymbol{x}) < 0$. To prove the statement by contradiction, suppose that $1 = \mathbf{1}\{\beta f(\boldsymbol{x}^*)f(\boldsymbol{x}) < 0\} > \sum_{k=0}^{m-1} \mathbf{1}\{\beta f(\gamma(\frac{k}{m}))f(\gamma(\frac{k+1}{m})) \leq 0\}$, or equivalently, $f(\gamma(\frac{k}{m}))f(\gamma(\frac{k+1}{m})) > 0$ for all $k = 0, \cdots, m-1$. In other words, we have the same sign values for every $(\frac{k}{m}, \frac{k+1}{m})$ pairs which leads a contradiction with the assumption that f has different sign values at the end points $\gamma(0) = \boldsymbol{x}$ and $\gamma(1) = \boldsymbol{x}^*$. □

Now, we establish a new upper bound on $\mathcal{R}_{rob}(f) - \mathcal{R}^\star_{nat}$.

Theorem 3.3. *Given a sample $\boldsymbol{x}$ and a positive β, let $\gamma : [0,1] \to \mathcal{X}$ be a continuous path from $\gamma(0) = \boldsymbol{x}$ to $\gamma(1) = \boldsymbol{x}^*$ where $\boldsymbol{x}^* = \arg\max_{\boldsymbol{x'} \in \mathbb{B}(\boldsymbol{x},\epsilon)} \mathbf{1}\{\beta f(\boldsymbol{x'})f(\boldsymbol{x}) < 0\}$. Then, for any non-negative classification-calibrated loss function ϕ such that $\phi(0) \geq 1$, we have*

$$\begin{aligned}\mathcal{R}_{rob}(f) - \mathcal{R}^\star_{nat} \leq \psi^{-1}(\mathcal{R}_\phi(f) - \mathcal{R}^\star_\phi) \\ + \mathbb{E}_{(\boldsymbol{x},y)} \sum_{k=0}^{m-1} \phi(\beta f(\gamma(\frac{k}{m}))f(\gamma(\frac{k+1}{m})))\end{aligned}$$

where $\mathcal{R}^\star_{nat} := \inf_f \mathcal{R}_{nat}(f)$, $\mathcal{R}^\star_\phi := \inf_f \mathcal{R}_\phi(f)$ and ψ^{-1} is the inverse function of the ψ-transform of ϕ.

Proof. By Eq. (3.22), the first inequality holds. Similarly, the second inequality holds by Eq. (3.23), and the last inequality holds because we choose a classification-calibrated loss ϕ.

$$\begin{aligned}\mathcal{R}_{rob}(f) - \mathcal{R}^\star_{nat} &\leq \psi^{-1}(\mathcal{R}_\phi(f) - \mathcal{R}^\star_\phi) + \mathbb{E}_{(\boldsymbol{x},y)} \mathbf{1}\{\beta f(\boldsymbol{x}^*)f(\boldsymbol{x}) < 0\} \\ &\leq \psi^{-1}(\mathcal{R}_\phi(f) - \mathcal{R}^\star_\phi) + \mathbb{E}_{(\boldsymbol{x},y)} \sum_{k=0}^{m-1} \mathbf{1}\{\beta f(\gamma(\frac{k}{m}))f(\gamma(\frac{k+1}{m})) \leq 0\} \\ &\leq \psi^{-1}(\mathcal{R}_\phi(f) - \mathcal{R}^\star_\phi) + \mathbb{E}_{(\boldsymbol{x},y)} \sum_{k=0}^{m-1} \phi(\beta f(\gamma(\frac{k}{m}))f(\gamma(\frac{k+1}{m})))\end{aligned}$$

□

Theorem 3.3 tells us that our proposed method provides an upper bound on the robust error of the model. To push further, we prove that the suggested loss is tighter than that of TRADES under a weak assumption on the path $\gamma(\cdot)$.

Following (Zhang et al., 2019b), we use KL divergence loss (KL) as a classification-calibrated loss. To do this, we define $p(x) := \sigma(f(x))$ where σ is a sigmoid function. Then, a model output with softmax can be denoted as $\boldsymbol{p}(x) := [p(x), 1 - p(x)]$. In this setting, we can prove that the suggest loss is tighter than that of TRADES under a weak assumption on $\gamma(\cdot)$.

Assumption 1. *$[\boldsymbol{p}(\gamma(t))]_y$ is a decreasing function of $t \in [0, 1]$, where $[\boldsymbol{p}(\cdot)]_y$ indicates the probability corresponding to the correct label y.*

Theorem 3.4. *Under Assumption 1, the KL divergence loss has the following property:*

$$\sum_{k=0}^{m-1} KL(\boldsymbol{p}(\gamma(\frac{k}{m}))||\boldsymbol{p}(\gamma(\frac{k+1}{m})) \leq KL(\boldsymbol{p}(\gamma(0))||\boldsymbol{p}(\gamma(1))).$$

Proof. Let $p_1(x)$, $p_2(x)$, and $p_3(x)$ denotes three different distribution with possible outcomes $x = \{-1, 1\}$ and $0 < p_1(x = 1) \leq p_2(x = 1) \leq p_3(x = 1) < 1$. Then,

$$\begin{aligned}
&\mathrm{KL}(p_1||p_2) + \mathrm{KL}(p_2||p_3) - \mathrm{KL}(p_1||p_3) \\
&= \sum_{x\in\{0,1\}} p_1(x) \ln \frac{p_1(x)}{p_2(x)} + \sum_{x\in\{0,1\}} p_2(x) \ln \frac{p_2(x)}{p_3(x)} - \sum_{x\in\{0,1\}} p_1(x) \ln \frac{p_1(x)}{p_3(x)} \\
&= \sum_{x\in\{0,1\}} (p_2(x) - p_1(x)) \ln p_2(x) + \sum_{x\in\{0,1\}} (p_1(x) - p_2(x)) \ln p_3(x) \\
&= - \sum_{x\in\{0,1\}} (p_1(x) - p_2(x))(\ln p_2(x) - \ln p_3(x)) \\
&\leq 0
\end{aligned}$$

The last inequality holds because $p_1(x) - p_2(x)$ and $p_2(x) - p_3(x)$ have the same sign regardless of x. Likewise, for $0 < p_1(x = -1) \leq p_2(x = -1) \leq p_3(x = -1) < 1$, the statement also holds true. Thus, by mathematical induction, $\sum_{k=0}^{m-1} \mathrm{KL}(\boldsymbol{p}(\gamma(\frac{k}{m}))||\boldsymbol{p}(\gamma(\frac{k+1}{m})) \leq \mathrm{KL}(\boldsymbol{p}(\gamma(0))||\boldsymbol{p}(\gamma(1)))$ under Assumption 1. □

For the multi-class problem, we can extend Theorem 3.4 by assuming $[\boldsymbol{p}(\gamma(u))]_i$ as a monotonic function for each individual component $i \in \mathcal{Y}$.

Table 3.1: Detailed structure of Wav2vec-based models.

Name	Layers
LS-Large	LS(512, 512)-FC(512, 12)
LS-Small	LS(512, 64)-FC(64, 12)
FC-Large	FC(512×98, 1000)-FC(1000,500)-FC(500,12)
FC-Small	FC(512×98, 12)

3.4 Experiments

3.4.1 Transferability

Experimental setup

The Google speech commands dataset consists of 65,000 one-second utterances by thousands of people. The target speech command are such as "yes", "no", "up", "down", "left", "right", "on", "off", "stop", and "go". For the frequency-based extractor, we set the number of Mel-filterbanks to 32, the size of the FFT to 2048, and the length of the hop between windows to 512, and it outputs Mel-spectrogram with a size of 32 × 32. For the Wav2vec-based extractor, given a raw audio waveform as an input, the encoder of pre-trained Wav2vec constructs the representative vector with the shape of 512×98. The detailed structure of each classifier is summarized in Table 3.1.

The VCTK dataset consists of speech data uttered by 110 English speakers. Following (Li et al., 2020b), the first 10 speakers in the VCTK dataset were used in this experiment. We split the dataset into training and test datasets, with a 8:2 ratio. The model should classify the input utterances as the target speaker. For the frequency-based extractor, we set the number of Mel-filterbanks to 30, the size of the FFT to 2048, and the length of the hop between windows to 512. Following frequency-based extractor, X-vectors (Snyder et al., 2018) is used as classifier (Model-F). For the

Wav2vec-based model, we used the encoder of pre-trained Wav2vec as the extractor and LS-Large as the classifier (Model-W).

For both datasets, the models were trained with 70 epochs and the Adam optimizer was used with an initial learning rate of 0.005. We stress that Wav2vec was fixed, and only the classifier was trained during the training session. We used the cosine-annealing learning rate schedule. For the attacks, we set the maximum perturbation ϵ to satisfy $dB_x(\boldsymbol{\delta}) = -20$ which is similar to the value used in (Carlini and Wagner, 2018). Their step size α is fixed at $2\epsilon/T$, where T is the number of steps which is set to 10. For NIA, a uniform distribution $\mathbf{U}[-\epsilon, \epsilon]$ is used as the noise distribution $\mathcal{N}$. All adversarial attacks require approximately 0.78s for generating one audio adversarial example on DenseNet in average. All experiments were performed using PyTorch (Paszke et al., 2019) and Torchattacks (Kim, 2020) on a single NVIDIA TITAN V under Ubuntu 16.04.4 LTS. The hardware used in experiments is Intel(R) Xeon(R) Gold 6126 CPU, 2.60GHz, 48 cores and 394GB memory.

Experimental results

Here, we provide the transferability of comparison methods and the proposed method. In addition to CW and PGD, we also considered DIM (Xie et al., 2019) and TIM (Dong et al., 2019) as the comparison methods, which generate the most transferable adversarial examples in the vision domain. We converted their 2D transformations into 1D transformations. For DIM, we 1D resize and 1D padding at every step. The resize ratio was set to 0.9, then zero-pads the resized audio waveform to have the same shape of the original waveform. For TIM, we evaluated its attack success rate for diverse kernel lengths $k \in [3, 5, 7, 9, 11, 15]$, because TIM is very sensitive to its kernel length and shows the best performance when $k = 3$. Thus, we fixed the kernel

Table 3.2: Transferability (%) of each adversarial attack for different source and target model on the Google speech commands dataset. The best results are shown in bold. Green-colored cells correspond to the case when NIA shows the best performance.

Source Model	Attack	Target Model						
		Frequency-based			Wav2vec-based			
		DenseNet	VGG	WRN	LS-Large	LS-Small	FC-Large	FC-Small
DenseNet	CW	**99.92**	53.25	57.54	13.40	14.10	16.48	19.87
	PGD	99.57	90.22	94.74	38.57	43.79	43.05	49.51
	DIM	99.34	**92.64**	**95.17**	41.18	47.45	46.83	53.33
	TIM	95.40	77.80	82.12	43.55	48.66	52.24	57.03
	NIA	95.01	89.83	91.62	**53.06**	**57.58**	**56.33**	**58.71**
VGG	CW	53.76	**100.00**	52.28	13.83	15.08	16.32	18.31
	PGD	93.26	**100.00**	95.21	44.96	50.76	51.11	52.86
	DIM	96.84	99.96	97.78	48.77	56.33	56.14	55.98
	TIM	83.48	99.73	81.26	44.25	49.24	53.72	54.15
	NIA	**98.32**	99.84	**98.91**	**64.20**	**69.81**	**67.32**	**66.77**
WRN	CW	53.25	47.80	**99.49**	13.60	14.53	16.17	17.76
	PGD	88.24	84.50	96.49	30.11	32.49	35.76	39.77
	DIM	89.21	86.99	94.90	33.97	37.63	38.92	42.66
	TIM	79.74	75.07	92.29	36.85	42.89	48.66	49.63
	NIA	**89.55**	**88.35**	92.99	**43.36**	**49.01**	**48.77**	**50.44**
LS-Large	CW	17.72	17.88	17.76	**100.00**	53.17	49.55	61.86
	PGD	55.63	54.89	56.53	**100.00**	96.03	94.08	97.23
	DIM	67.12	66.38	67.39	99.88	96.80	96.01	**98.91**
	TIM	36.27	33.70	36.27	99.45	87.03	80.02	88.62
	NIA	**72.46**	**72.07**	**72.96**	99.57	**97.55**	**96.07**	96.46
LS-Small	CW	17.92	18.58	18.15	45.97	99.96	47.29	59.95
	PGD	60.93	61.12	63.50	96.49	**100.00**	95.99	98.17
	DIM	70.20	69.81	71.48	97.94	99.92	98.21	98.79
	TIM	40.71	38.61	39.54	88.86	99.69	87.46	91.74
	NIA	**78.38**	**77.02**	**79.08**	**98.56**	99.96	**98.25**	**98.87**
FC-Large	CW	16.63	16.36	15.97	37.79	39.46	**99.88**	70.32
	PGD	51.69	48.93	51.11	89.95	92.13	99.38	98.05
	DIM	65.72	64.20	65.21	95.68	95.29	99.14	98.71
	TIM	32.76	30.54	31.83	73.24	77.29	98.87	93.57
	NIA	**79.28**	**78.54**	**77.02**	**95.95**	**96.38**	99.18	**98.75**
FC-Small	CW	7.28	7.79	6.90	15.35	16.48	24.07	84.92
	PGD	54.85	51.27	54.11	95.13	96.92	99.38	**100.00**
	DIM	65.17	62.80	65.37	97.62	98.44	99.38	**100.00**
	TIM	36.62	31.63	35.88	81.92	85.55	92.29	**100.00**
	NIA	**77.17**	**75.54**	**77.13**	**98.29**	**98.83**	**99.69**	**100.00**

length $k = 3$ with the Gaussian kernel.

The results on the Google speech commands and VCTK datasets are summarized in Table 3.2 and 3.3. For both datasets, NIA achieves the best transferability for most cases. Specifically, on the Google speech commands dataset, when the source model

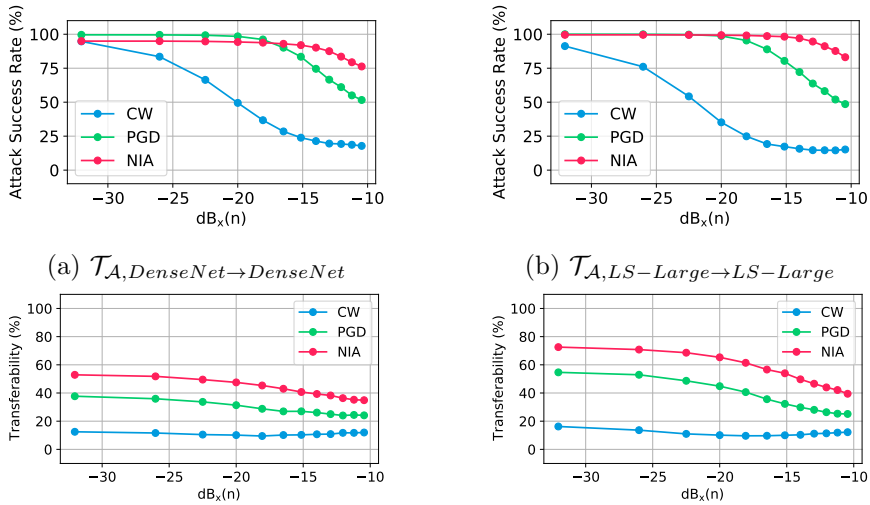
Table 3.3: Transferability (%) of each adversarial attack for different source and target model on the VCTK dataset. The best results are shown in bold. Green-colored cells correspond to the case when NIA shows the best performance.

Source Model	Attack	Target Model	
		Frequency-based (Model-F)	Wav2vec-based (Model-W)
Frequency-based (Model-F)	**CW**	**100.00**	65.68
	PGD	99.87	82.64
	DIM	99.87	81.97
	TIM	98.65	79.41
	NIA	**100.00**	**86.27**
Wav2vec-based (Model-W)	**CW**	61.51	**100.00**
	PGD	82.73	**100.00**
	DIM	84.25	99.73
	TIM	80.75	96.64
	NIA	**85.20**	**100.00**

is CW, PGD, DIM, and TIM only show the $\mathcal{T}_{NIA,DenseNet\rightarrow LS-Large}$ of 13.40%, 38.57%, 41.18%, and 43.55%, respectively. However, NIA shows largely increased transferability, $\mathcal{T}_{NIA,DenseNet\rightarrow LS-Large} = 53.06\%$, even the source model and the target model have different feature extractors. This tendency is observed for most cases on the Google speech commands dataset.

In addition, on the VCTK dataset, NIA shows the best transferability as shown in Table 3.3. Specifically, NIA shows $\mathcal{T}_{NIA,Model-F\rightarrow Model-W} = 86.27\%$, which is higher than that of PGD (82.64%). Similarly, NIA shows the best transferability $\mathcal{T}_{NIA,Model-W\rightarrow Model-F} = 85.20\%$, which is higher than that of PGD (82.73%) and DIM (84.25%). Thus, we can conclude that NIA successfully boosts the transferability.

Based on the fact that audio adversarial examples are easily nullified by additive noise, we conducted an experiment on the Google speech command dataset with varying magnitude of uniform noise $\boldsymbol{n}$. As shown in Fig. 3.10, the attack success rate



(a) $\mathcal{T}_{\mathcal{A},DenseNet \to DenseNet}$

(b) $\mathcal{T}_{\mathcal{A},LS-Large \to LS-Large}$

(c) $\mathcal{T}_{\mathcal{A},DenseNet \to LS-Large}$

(d) $\mathcal{T}_{\mathcal{A},LS-Large \to DenseNet}$

Figure 3.10: Attack success rate (3.10a and 3.10b) and transferability (3.10c and 3.10d) of each attack method under the noisy setting. x-axis indicates the relative magnitude of additive noise $\boldsymbol{n}$ to the original audio sample x.

decreased as the magnitude of the uniform noise increased in all cases. However, NIA shows a stable performance against additive noise, not only for transferability but also for the white-box attack setting compared to CW and PGD. In particular, NIA shows the best transferability for varying magnitudes of noise (Fig. 3.10c and 3.10d). This implies that NIA yields lower NR, which is connected to the transferability of adversarial examples in real-world scenarios.

Similarly, on the VCTK dataset, we evaluate the transferability of each attack in the noisy environment. We used the same setting $\boldsymbol{n} \sim \mathbf{U}[-\epsilon/5, \epsilon/5]$. The results are summarized in Table 3.4. Again, NIA shows the best performance for all cases even for the white-box attack setting. Moreover, NIA shows the lowest NR than any other comparison methods, which supports the connection between the transferability and the noise sensitivity.

Table 3.4: Transferability (%) of each adversarial attack in the noisy environment. The NR (%) in Eq. (3.10) is also reported in parentheses. The best results are shown in bold. Green-colored cells correspond to the case when NIA shows the best performance.

Source Model	Attack	Target Model	
		Frequency-based (Model-F)	Wav2vec-based (Model-W)
Frequency-based (Model-F)	**CW**	73.76 (26.24)	56.53 (13.93)
	PGD	99.33 (0.54)	73.22 (11.40)
	DIM	99.33 (0.54)	73.62 (10.19)
	TIM	97.44 (1.23)	69.72 (12.20)
	NIA	**100.00 (0.00)**	**79.00 (8.43)**
Wav2vec-based (Model-W)	**CW**	48.59 (21.00)	73.22 (26.78)
	PGD	72.01 (12.58)	91.66 (8.34)
	DIM	74.02 (12.14)	92.19 (7.56)
	TIM	70.14 (13.14)	85.06 (11.98)
	NIA	**75.56 (11.31)**	**99.19 (0.81)**

Furthermore, we estimate sample-wise variance (SV) with varying the number of steps to compare the overall transferability of adversarial attacks. Fig. 3.11 shows the results on the Google speech commands dataset, where DenseNet is the source model h_S. As the number of steps increases, the transferability of CW and PGD decreases. Surprisingly, however, NIA shows continuously increasing performance as the number of steps increases. Even with a large number of steps, NIA shows a high transferability against the same feature extractor-based models (approximately 90%) and different feature extractor-based models (approximately 60%). Moreover, NIA achieves a higher transferability with a lower SV compared to CW and PGD for all number of steps.

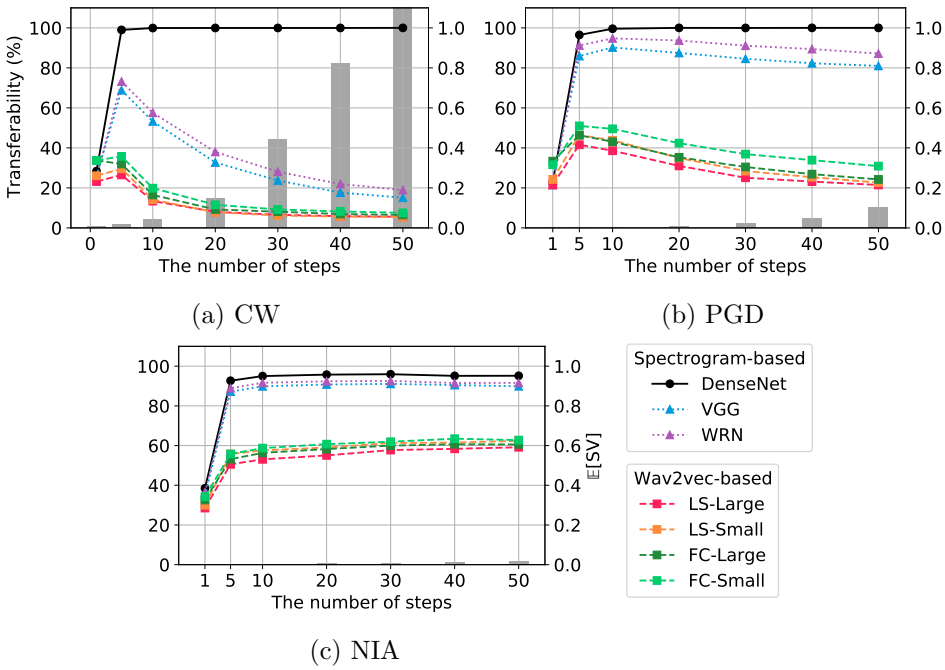


Figure 3.11: Transferability and SV for different number of steps. Each line denotes the attack success rate $\mathcal{T}_{\mathcal{A},DenseNet \to T}$ for a target model T. Each color corresponds to a different target model. Gray bars correspond to the expected SV ($\mathbb{E}[SV]$) for the source model.

3.4.2 Adversarial Robustness

Experimental setups

For MNIST, we train LeNet (LeCun et al., 1998) for 50 epochs with the Adam optimizer. Note that only MART is trained SGD with the initial learning rate 0.01 following (Wang et al., 2019b), because MART converges to a constant function when the Adam optimizer is used. The initial learning rate is 0.001 and it is divided by 10 at 30 and 40 epochs. During training, we use PGD^{40} with $\epsilon = 0.3$ and 0.45, which are commonly used in prior work (Wang et al., 2019b; Sitawarin et al., 2020). We use PGD^{40} with $\epsilon = 0.3$ and the step-size $\alpha = 0.02$ to generate adversarial

examples in the training session. No preprocessing or input transformation is used.

For CIFAR10, we train WRN-28-10 with two different training schemes. The one is the step-wise decay setting, which is generally used, and the other one is cyclic learning rate schedule (Smith, 2017), which is a recently proposed training scheme that enables the model to converge faster (Wong et al., 2020). We report the results with cyclic learning rate schedule, because its performance is slightly better than the step-wise decay setting. We use 0.3 as the maximum learning rate and a total of 30 epochs for training. During training, we use PGD^{10} with $\epsilon = 8/255$ and $16/255$, which are commonly used in prior work (Madry et al., 2018; Wang et al., 2019b). The step-size is set to $\alpha = 2/255$. Horizontal flip and random cropping with padding of 4 are used for data augmentation.

For Tiny ImageNet, we used PreActResNet18 (He et al., 2016b). Following the settings of (Rice et al., 2020; Pang et al., 2020), we train a model 110 epochs with SGD optimizer and an initial learning rate is 0.1 and decayed with a factor of 0.1 at the 100th and 105th epoch. We use PGD^{10} with $\epsilon = 8/255$ and the step-size $\alpha = 2/255$ to generate adversarial examples in the training session as used in (Kim et al., 2021a). Horizontal flip and random cropping with a padding of 4 are used for data augmentation. For Tiny ImageNet, we find that the performance under $\beta > 5$ is sometimes better than that of $\beta = 5$. Thus, we perform grid search on $\beta = \{1, 5, 10, 20, 40\}$ and choose the best β that records the highest robustness against PGD^{50} with $\epsilon = 8/255$. Similarly, we found that $m = 3$ produces the best performance on Tiny ImageNet among $m = \{2, 3, 5, 10\}$.

We basically evaluate the robustness of all models with PGD^{50}. Furthermore, we also consider AutoAttack (Croce and Hein, 2020), which is a combination of

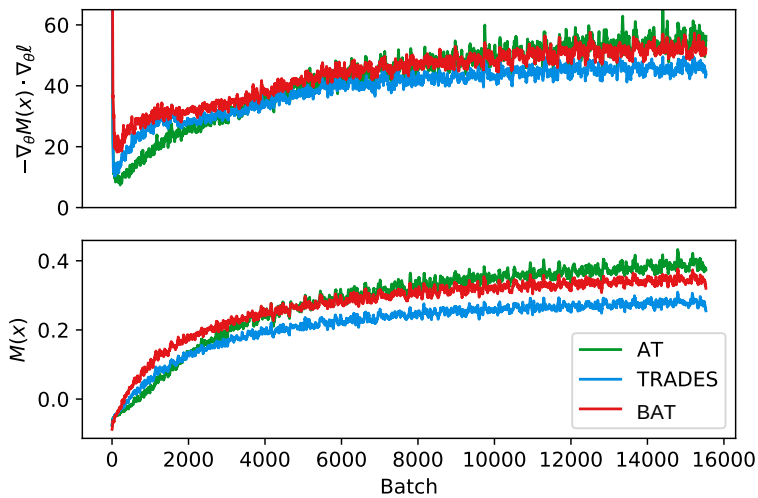


Figure 3.12: (CIFAR10) Analysis on the margin during the first 40 epochs. Top: the expected margin increase $-\nabla_\theta M(\boldsymbol{x}) \cdot \nabla_\theta \ell$ of each method. Bottom: the actual margin $M(\boldsymbol{x})$ of each method. The proposed method shows a larger margin than TRADES by mitigating the negative effect.

three white-box attacks (Croce and Hein, 2020, 2019) and one black-box attack (Andriushchenko et al., 2020). Note that AutoAttack is by far the most reliable attack to measure robustness. Unless otherwise specified, we use the same ϵ that was used in the training session during evaluation.

Experimental results

Effect of gradient and actual margin Here, we remark that the gradient of $-\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$ has a direction opposite to the gradient of the margin $M(\boldsymbol{x})$ as described in Proposition 3.2. Moreover, we experimentally confirmed this negative effect in Fig. 3.9. In this paragraph, we further measure the actual effect of the gradient on margin and its value during the training. First, we observe the effect of the gradient descent $\nabla_{\boldsymbol{\theta}} \ell$ on the margin maximization, measured by $-\nabla_{\boldsymbol{\theta}} M(\boldsymbol{x}) \cdot \nabla_{\boldsymbol{\theta}} \ell$.

This indicates the expected margin increase by the weight update with the loss ℓ. Then, we measure the actual margin $M(\boldsymbol{x})$.

Fig. 3.12 shows that the proposed method mitigates the negative effect of the regularization term during training. Compared to TRADES, the proposed method shows a higher expected increase in the margin, and this enables the model to achieve a large margin. Thus, by introducing the intermediate probability $\tilde{\boldsymbol{p}}$, we successfully encourage the model to reduce the negative effect of the regularization term on maximizing the margin.

Gradient magnitude A larger norm of gradient also serves to explain the advantage of the proposed method. As prior works discovered (Liu et al., 2020; Dong et al., 2021), a larger gradient norm in the initial training phase enables the model to escape a suboptimal region. To provide a fair comparison for different training methods, we normalize the norm of gradient by the norm of the loss value as follows:

$$\left\| \frac{\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})}{\ell(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta})} \right\|_2 \tag{3.24}$$

As shown in Fig. 3.13, the gradients of AT show the smallest normalized gradient norm among different training methods. This implies that AT has difficulty escaping from the initial suboptimal region (Liu et al., 2020). It is also supported by the experiments under the large maximum perturbation setting in Fig. 3.7. Compared to AT, TRADES shows a higher norm of the gradients. This is consistent with the observation that TRADES provides more gradient stability with the continuous loss landscape (Dong et al., 2021). However, as shown in Fig. 3.7, TRADES also has difficulty reaching the global optima with high clean accuracy and robustness under

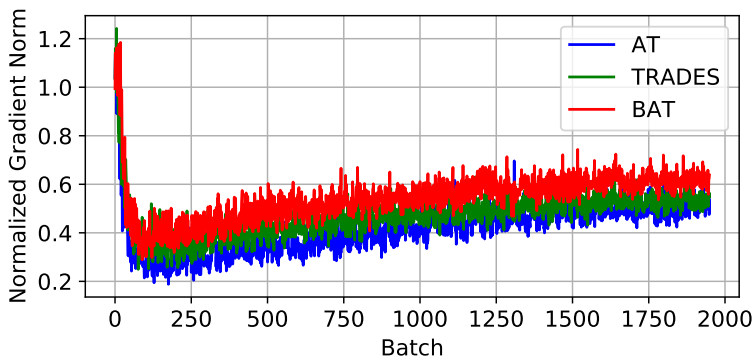
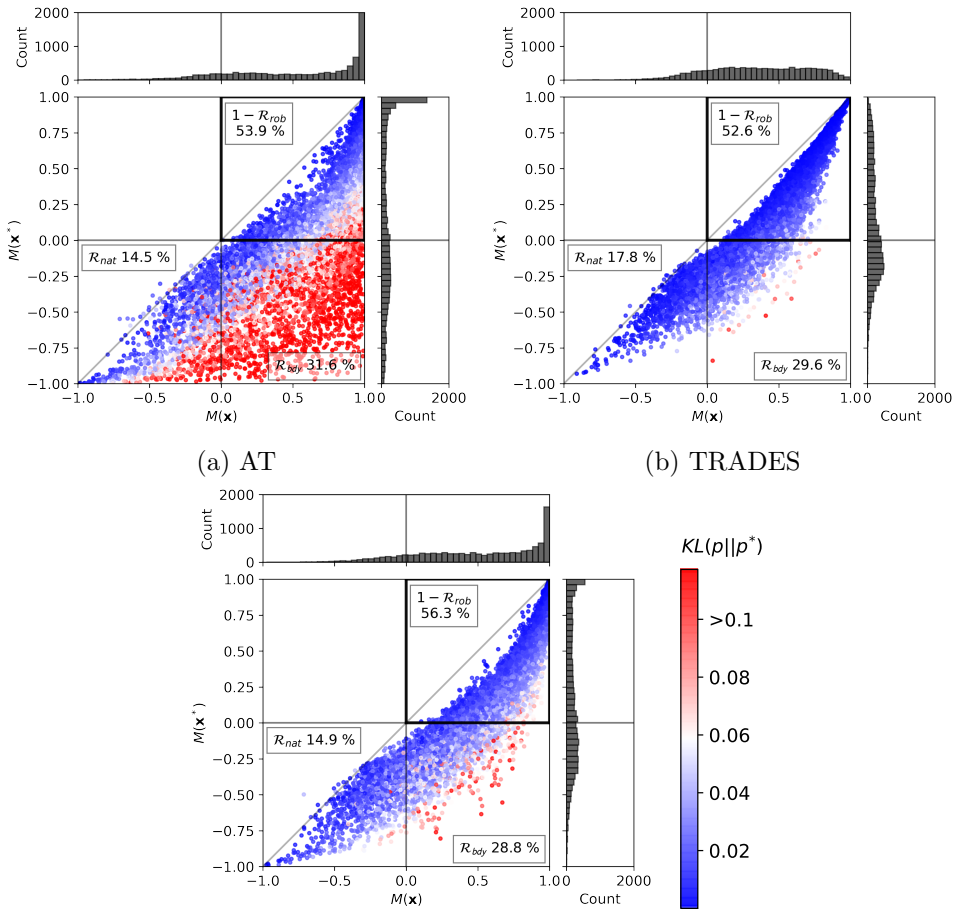


Figure 3.13: (CIFAR10) Normalized gradients of each loss term during the first 2000 batches. The proposed method (BAT) shows the largest gradient magnitude, which can help the model quickly escapes the initial suboptimal region (Dong et al., 2021).

the large maximum perturbation setting. In contrast, the proposed method shows the highest normalized gradient norm gained by mitigating the negative effect in Fig. 3.13. Considering its stable performance even for larger perturbation conditions, we believe that this might be explain the advantages of the proposed method.

Balanced margin and smoothness Previously, we verified that the proposed method successfully mitigates the negative effect during the initial training phase. Now, we investigate whether the proposed method achieves sufficient smoothness while mitigates the negative effect until the end of training.

First, for each trained model, we generate a corresponding adversarial dataset $(\boldsymbol{x}^*, \boldsymbol{y})$ by using PGD^{50} with $\epsilon = 8/255$ and the step-size $\alpha = 2/255$. Then, we plot pairs of the margins of clean examples $\boldsymbol{x}$ and corresponding adversarial examples $\boldsymbol{x}^*$ with their KL divergence. In each plot, the upper and the right histograms show the distribution of $M(\boldsymbol{x})$ and $M(\boldsymbol{x}^*)$, respectively. We colored each point by the KL divergence $\text{KL}(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})||\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}^*))$ to measure the smoothness. The red points have



(a) AT (b) TRADES

(c) BAT

Figure 3.14: (CIFAR10) Distribution of the margins $M(\boldsymbol{x})$ and $M(\boldsymbol{x}^*)$. Each point indicates each test example, and the color of each point indicates the KL divergence loss $\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$. The darker red ones indicate a higher KL divergence loss.

high KL divergence (poor smoothness) and the blue points have low KL divergence (better smoothness).

Now, we provide a detailed explanation of the plot. Each quadrant corresponds to:

- **Quadrant I:** $M(\boldsymbol{x}^*) > 0$. $\rightarrow$ Adversarial robustness (1 - $\mathcal{R}_{rob}$).
- **Quadrant III:** $M(\boldsymbol{x}) < 0$. $\rightarrow$ Natural classification error ($\mathcal{R}_{nat}$).
- **Quadrant IV:** $M(\boldsymbol{x}) > 0$ and $M(\boldsymbol{x}^*) < 0$. $\rightarrow$ Boundary error ($\mathcal{R}_{bdy}$)

Note that there is no point in the second quadrant (**Quadrant II**) because adversarial attacks generally do not make incorrect examples ($M(\boldsymbol{x}) < 0$) correctly classified ($M(\boldsymbol{x}^*) > 0$). Here, $\mathcal{R}_{nat}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\arg\max_i f(\boldsymbol{x})_i \neq y\}$ is the natural classification error and $\mathcal{R}_{rob}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\exists \boldsymbol{x'} \in \mathbb{B}(\boldsymbol{x}, \epsilon) \text{ s.t. } \arg\max_i f(\boldsymbol{x'})_i \neq y\}$ is the robust error. Then, $\mathcal{R}_{rob}(f)$ can be decomposed as follows:

$$\mathcal{R}_{rob}(f) = \mathcal{R}_{nat}(f) + \mathcal{R}_{bdy}(f) \tag{3.25}$$

where $\mathcal{R}_{bdy}(f) := \mathbb{E}_{(\boldsymbol{x},y)}\mathbf{1}\{\arg\max_i f(\boldsymbol{x})_i = y, \exists \boldsymbol{x'} \in \mathbb{B}(\boldsymbol{x}, \epsilon) \text{ s.t. } \arg\max_i f(\boldsymbol{x})_i \neq \arg\max_i f(\boldsymbol{x'})_i\}$ is the boundary error in (Zhang et al., 2019b). Thus, the ultimate purpose of adversarial training is to move all the points to the first quadrant.

As shown in Fig. 3.14, the proposed method provide a balanced margin and smoothness with better robustness. Compared to TRADES, which has only a few samples with a high margin $M(\boldsymbol{x})$ and $M(\boldsymbol{x}^*)$, the proposed method shows better margin distributions near 1 for both clean and adversarial examples. This implies that our method successfully mitigates the negative effect of the regularization term on maximizing the margin as discussed in Section 3.3.2. As a result, our method

Table 3.5: (CIFAR10) Sensitivity of β.

Method	Clean	FGSM	PGD^{50}
$\beta = 1$	**89.03**	56.98	50.33
$\beta = 2$	87.08	**59.14**	53.32
$\beta = 3$	85.89	57.46	52.52
$\beta = 4$	84.72	57.88	53.29
$\beta = 5$	83.69	58.04	**53.86**
$\beta = 6$	82.87	57.73	53.72
$\beta = 7$	82.10	57.11	53.75
$\beta = 8$	81.49	57.42	53.78

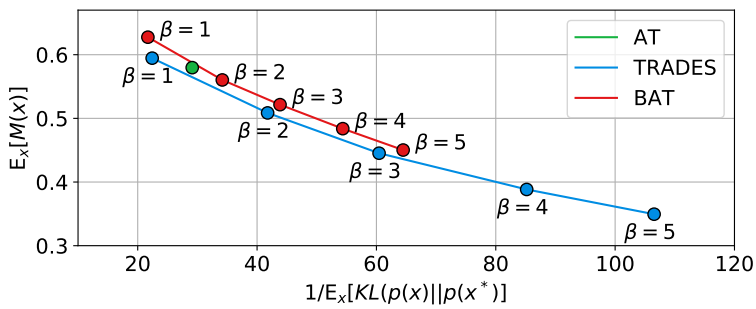


Figure 3.15: Sensitivity of β in terms of margin and smoothness. x-axis denotes the inverse of the expectation of the KL divergence and y-axis denotes the expectation of the margin. Each expectation is calculated on the test set. Higher is better for both axes.

achieves a lower natural classification error (14.9%) than that of TRADES (17.8%). Compared to AT, the proposed method has fewer red points, which implies that the proposed method provides smoothness. Due to the increased smoothness, the boundary error of the proposed method ($\mathcal{R}_{bdy} = 28.8\%$) is lower than that of AT ($\mathcal{R}_{bdy} = 31.6\%$).

Sensitivity analysis Here, we evaluate the sensitivity on the hyper-parameters of the proposed method, β and m. Table 3.5 and Table 3.6 shows the sensitivity of β

Table 3.6: (CIFAR10) Sensitivity of m.

Method	Clean	FGSM	PGD^{50}
$m = 1$	82.77	59.07	55.79
$m = 2$	85.14	60.18	**56.01**
$m = 3$	85.98	**59.35**	54.25
$m = 4$	86.30	58.35	53.40
$m = 5$	**87.14**	57.21	51.16

and m, respectively. All experiments are performed on CIFAR10 with cyclic learning rate decay as same in Section 3.4.2. Here, we only performed each experiment once.

For the analysis on β, we fixed $m = 2$. As shown in Table 3.5, as β increases, the standard accuracy decreases. To push further, we measure the margin and smoothness with varying β. For better visualization, we used the inverse of the expectation of the KL divergence (because the smoothness is better when $\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$ is lower). Thus, as shown in Fig. 3.15, we plot the inverse of the KL divergence (x-axis) and the margin (y-axis) for each model as shown in Fig. 3.15. The proposed method shows better values in terms of margin and smoothness with varying β.

Table 3.6 shows the sensitivity of m. Here, we fix $\beta = 5$ from the observation in Table 3.5. Due to decreased negative effect, as m increases, the model shows a higher accuracy. However, we can not observe a similar increasing tendency of adversarial robustness with m. In addition, we note that increasing m requires more time for calculating intermediate probabilities. $m = 2$ only requires 1.3 seconds per iteration, which is only 8% increase of $m = 1$ (1.2 seconds per iteration).

Adversarial Robustness Here, we verify the robustness of the proposed method. Here, we adopt three benchmark datasets, i.e, MNIST, CIFAR10, and Tiny ImageNet. They are widely used datasets to evaluate the performance of adversarially

Table 3.7: Robustness accuracy (%) on MNIST. All models trained using PGD^{40} with $\epsilon = 0.3$ and 0.45, then evaluated by each attack with the same $\epsilon = 0.3$ and 0.45, respectively.

Method	Clean	PGD^{50}	AutoAttack
(Training $\epsilon = 0.3$)			
AT	98.79±0.23	91.69±1.25	88.64±0.61
TRADES	**98.89±0.01**	93.70±0.01	**92.31±0.21**
MART	98.78±0.16	92.37±1.21	89.62±1.44
BAT	98.79±0.06	**93.97±0.16**	92.19±0.13
(Training $\epsilon = 0.45$)			
AT	11.35±0.00	11.35±0.00	11.35±0.00
TRADES	**99.42±0.05**	11.34±0.01	0.53±0.26
MART	13.13±2.54	7.80±3.73	0.93±1.32
BAT	97.72±0.26	**88.20±0.57**	**76.09±1.65**

trained models. To push further, in addition to AT and TRADES, we also consider MART (Wang et al., 2019b), which aims to maximize the margin and recently achieved the best performance by focusing on misclassified examples.

For all values, we report the average and the standard deviation of the performance over three runs with different random seeds. We use PyTorch (Paszke et al., 2019) and Torchattacks (Kim, 2020) for all experiments.

As shown in Table 3.7, for MNIST with $\epsilon = 0.3$, all defenses show high robustness. However, for a large $\epsilon = 0.45$, all comparison methods converge to a constant function or fail to gain robustness. In other words, the existing methods have difficulty converging to the global optimal. For the cases of AT and MART, they have the term that maximizes the margin of adversarial examples so that it can have difficulty in convergence (Dong et al., 2021). In contrast, TRADES also fails to achieve stable robustness, because a larger perturbation brings stronger negative effect of

Table 3.8: Robustness accuracy (%) on CIFAR10. All models trained using PGD^{10} with $\epsilon = 8/255$ and $16/255$, then evaluated by each attack with the same $\epsilon = 8/255$ and $16/255$, respectively.

Method	Clean	PGD^{50}	AutoAttack
(Training $\epsilon = 8/255$)			
AT	**85.65±0.33**	53.64±0.03	50.87±0.22
TRADES	82.22±0.12	52.14±0.08	48.90±0.35
MART	77.51±0.46	53.87±0.08	48.25±0.06
BAT	84.84±0.28	**55.64±0.37**	**52.41±0.02**
(Training $\epsilon = 16/255$)			
AT	72.43±0.01	29.01±0.13	24.24±0.51
TRADES	70.01±0.44	24.52±0.06	14.63±0.22
MART	65.97±0.54	**32.65±0.40**	23.23±0.14
BAT	**77.56±0.01**	30.79±0.35	**25.06±0.37**

Table 3.9: Robustness accuracy (%) on Tiny ImageNet. All models trained using PGD^{10} with $\epsilon = 8/255$, then evaluated by each attack with the same $\epsilon = 8/255$.

Method	Clean	PGD^{50}	AutoAttack
AT	**46.68±0.02**	15.26±0.22	11.52±0.11
TRADES	40.39±0.05	20.48±0.03	12.03±0.13
MART	42.45±0.12	19.43±0.42	11.32±0.51
BAT	42.47±0.04	**21.52±0.10**	**12.53±0.26**

$\mathrm{KL}(\boldsymbol{p}||\boldsymbol{p}^*)$ as we discussed in Section 2. However, the proposed method shows stable results even for $\epsilon = 0.45$. Considering that the difference between TRADES and the proposed method is the usage of bridging, this result tells us that the convergence becomes much easier by using the proposed bridged loss.

The proposed method also shows the best robustness on CIFAR10 (Table 3.8). Specifically, for $\epsilon = 16/255$, the proposed method achieves 77.56% of standard accuracy, which is 5% higher than AT. Compared to TRADES and MART, it is 6% and 12% higher, respectively. Simultaneously, it also achieves the best robustness 25.06% against AutoAttack. Note that the robustness of TRADES is only 14.63%, which shows the weakness of TRADES for a larger perturbation.

3.5 Chapter Summary

In this study, we investigated and analyzed the adversarial robustness of deep learning models in various domains. From extensive empirical and theoretical analyses, we improved the performance of both adversarial attack and adversarial defense. The results of our work can be effectively used to develop a new adversarial attack or defense method in consideration of diverse models and its loss landscape.

Chapter 4

Generalization and Loss Landscape

4.1 Chapter Overview

With a large number of parameters, deep learning models have shown remarkable improvements across a variety of domains. However, over-parameterized deep learning models may suffer from poor generalization, even though they enjoy near zero training loss. To alleviate this overfitting problem, prior studies have suggested diverse techniques such as augmentation (Zhang et al., 2017b; Yun et al., 2019) and regularization (Srivastava et al., 2014; Ioffe and Szegedy, 2015; Barrett and Dherin, 2020).

Recently, researchers have become interested in exploring the relationship between the geometric characteristics of the loss surface and the generalization performance (Hochreiter and Schmidhuber, 1994; Keskar et al., 2017; Li et al., 2018). Theoretical and empirical studies have demonstrated that the generalization performance is potentially related to the sharpness of the loss landscape and that an optimum with a flatter loss landscape can lead to better generalization. Based on these prior studies, Foret et al. (2020) recently proposed a new training framework called *sharpness-aware minimization* (SAM) that substantially improves the generalization performance on various tasks (Foret et al., 2020; Zhuang et al., 2021;

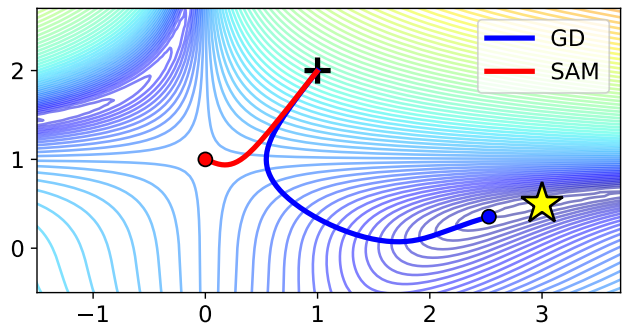


Figure 4.1: The optimization has begun at the point indicated by the plus sign, and the global minimum is indicated by the yellow star. SAM appears to be stuck at the saddle point, rather than converging to the global minimum.

Chen et al., 2022). The key idea of SAM is to minimize the maximum loss near its neighborhood in the weight space instead of minimizing the loss of the current weight. Thus, SAM can reach an optimum with a flatter loss landscape, resulting in improved generalization performance. Owing to its simplicity and ease of implementation, SAM has been adopted by many practitioners for training neural networks and improving the generalization performance.

However, we identify an instability in the convergence process of SAM dynamics near a saddle point, which may cause the suboptimal minimum problem. As shown in Fig. 4.1, the Beale function, a widely used optimization problem, is optimized using both vanilla gradient descent (GD) and SAM. GD reaches the global minimum (yellow star) but SAM is trapped in the saddle point. As deep learning models are highly non-linear and thus have multiple local minima and saddle points (Du et al., 2017; Kleinberg et al., 2018a), this convergence instability near a saddle point should be investigated.

To provide a comprehensive understanding of the convergence instability near

a saddle point in SAM dynamics, we analyze the dynamics of SAM near saddle points from the viewpoint of a dynamical system and show that SAM may not converge due to its gradient oscillation. We then prove that saddle points can become attractors under certain circumstances. We extend our investigation to the utilization of stochastic optimization with mini-batch sampling and establish the property of SAM diffusion, demonstrating that SAM requires more time to escape saddle points than stochastic gradient descent (SGD). Based on the diffusion, we further identify the importance of often overlooked training tricks, momentum and batch size, on the convergence instability of SAM. To the best of our knowledge, this is the first work that identifies and investigates the convergence instability of SAM near a saddle point. [1]

The main contributions of this study can be summarized as follows:

- We identify the convergence instability of SAM and theoretically prove the difficulty of escaping saddle points under SAM dynamics from the perspective of a dynamical system.

- We investigate the difficulty of saddle point escape under stochastic dynamical systems by establishing SAM diffusion and identify the importance of momentum and batch size for escaping saddle points with SAM diffusion.

- We conduct various experiments to support our theoretical results on a range of optimization settings, from a basic optimization problem to neural network-based benchmark tasks.

[1] This work was preprinted in arXiv (Kim et al., 2023).

4.2 Preliminaries

4.2.1 Generliazation and Sharpness-Aware Minimization

Given a loss function $\ell(\cdot)$ and weight parameters $\boldsymbol{w}$, the traditional optimization, the so-called empirical risk minimization, tries to minimize the following objective:

$$\min_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}; \mathcal{S}) := \frac{1}{n} \sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{S}} \ell(\boldsymbol{w}; \boldsymbol{x}, \boldsymbol{y}), \tag{4.1}$$

where $\mathcal{S} = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^n$ is a sampled training dataset with n instances. Under the general i.i.d. assumption on data, Eq. (4.1) generally yields a feasible solution of $\boldsymbol{w}$ on the true distribution of data $\mathcal{D}$,

$$\min_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}; \mathcal{D}) := \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim\mathcal{D}} \left[\ell(\boldsymbol{w}; \boldsymbol{x}, \boldsymbol{y})\right]. \tag{4.2}$$

However, in common practice, the i.i.d assumption is often violated by the limitation of training data and model structure. Therefore, a *generalization gap* can be defined as

$$\mathcal{E}(\boldsymbol{w}) = \mathcal{L}(\boldsymbol{w}; \mathcal{S}) - \mathcal{L}(\boldsymbol{w}; \mathcal{D}). \tag{4.3}$$

A low generalization gap indicates a high generalization performance. Therefore, the primary objective of machine learning is to attain the optimal solution that minimizes both the training loss $\mathcal{L}(\boldsymbol{w}; \mathcal{S})$ and the generalization gap $\mathcal{E}(\boldsymbol{w})$. Although the training loss can efficiently be minimized to near zero even with a vanilla training algorithm (Ishida et al., 2020), it still remains an open question how to minimize the generalization gap effectively. To address this issue, recent studies have focused on the *flatness* of the loss landscape as a potential solution (Hochreiter and Schmidhuber, 1994; Dziugaite and Roy, 2017) and provided experimental evidence that a

flatter loss landscape tends to have a better generalization performance (Jiang et al., 2019).

Among several algorithms (Chaudhari et al., 2019; Izmailov et al., 2018; Foret et al., 2020) that penalize sharp minima and seek flat minima, *sharpness-aware minimization* (SAM) (Foret et al., 2020) has been shown to be effective in reaching flat minima and has demonstrated significant improvements in generalization across various tasks and model structures, such as medical tasks (Anand et al., 2022) and transformers (Chen et al., 2022). SAM aims to minimize the worst-case loss over its parameter neighborhood rather than minimizing the loss of current parameter. Let us denote a vanilla gradient descent algorithm that minimizes the loss of current weight $\boldsymbol{w}_t$ at time t as

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \nabla \ell(\boldsymbol{w}_t), \tag{4.4}$$

where η is a learning rate and $\nabla \ell(\boldsymbol{v})$ is a gradient with respect to its input vector $\boldsymbol{v}$ unless specified otherwise. In contrast, SAM minimizes the loss of *perturbed weight* $\boldsymbol{w}_t^p$ by using the first-order Taylor approximation:

$$\boldsymbol{w}_t^p = \boldsymbol{w}_t + \rho \nabla \ell(\boldsymbol{w}_t). \tag{4.5}$$

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \nabla \ell(\boldsymbol{w}_t^p), \tag{4.6}$$

where ρ is a given neighborhood radius. Note that ρ can be normalized with the gradient norm as introduced in (Foret et al., 2020), i.e., $\rho / \|\nabla \ell(\boldsymbol{w})\|$; however, a constant ρ in Eq. (4.5) shows similar or higher performance than the normalized version (Andriushchenko and Flammarion, 2022). Therefore, SAM consistently enforces $\boldsymbol{w}$ to have a reduced perturbed loss $\ell(\boldsymbol{w}^p)$ within its ρ-neighborhood. This effectively makes the loss landscape smoother in the vicinity of its optimum, leading to a better

generalization performance than that of any other existing methods.

Following the success of SAM, further investigations have been conducted to explore its algorithm and enhance its generalization performance or computational efficiency. Kwon et al. (2021) proposed a transformation of parameters to achieve scale-invariant sharpness. Zhuang et al. (2021) argued that subtracting $\nabla \ell(\boldsymbol{w})$ from $\nabla \ell(\boldsymbol{w}^p)$ can reduce certain drawbacks in optimization. Liu et al. (2022) explored the effectiveness of a random perturbation during weight perturbation. Du et al. (2022) demonstrated that perturbing only some parameters can also improve the generalization performance with a reduced computational burden.

Despite the proposed variants and their improved generalization performance, some unclear points of the mechanism behind SAM still remain. For instance, Andriushchenko and Flammarion (2022) argued that the proposed generalization bound of SAM is incomplete, and instead connected the success of SAM to its implicit bias for different sizes of mini-batch. Our work also sheds light on understanding the mechanism of SAM with respect to its optimization properties, and to the best of our knowledge, this is the first work that identifies the convergence instability near a saddle point in SAM.

4.2.2 Escaping Saddle Points

The non-linearity of the loss landscape of neural networks results in the presence of multiple local minima and saddle points (Du et al., 2017; Kleinberg et al., 2018b). Prior studies have demonstrated that both gradient descent (GD) (Lee et al., 2016; Du et al., 2017) and stochastic gradient descent (SGD) (Xie et al., 2020; Ziyin et al., 2022) can be hindered by the saddle point. Therefore, many researchers have focused on escaping saddle points and reaching better minima. For instance, Du et al. (2017)

demonstrated that GD can take exponential time to escape saddle points even when using random initialization schemes. Xie et al. (2022) established SGD diffusion and the importance of momentum in overcoming the saddle point issue. Ziyin et al. (2022) also demonstrated that the learning rate can affect to the saddle point escape in SGD.

Similar to these works, this study presents the difficulty of SAM in escaping the saddle point under both asymptotic and stochastic system dynamics. We introduce a dynamical system-based geometric method to investigate when SAM can be hindered by saddle points. Moreover, we extend the diffusion theory in SGD (Xie et al., 2022) to SAM, illustrating the convergence instability near a saddle point in SAM dynamics. Additionally, we discuss the importance of carefully adopting training tricks in SAM, which emphasizes the observation presented by (Ziyin et al., 2022).

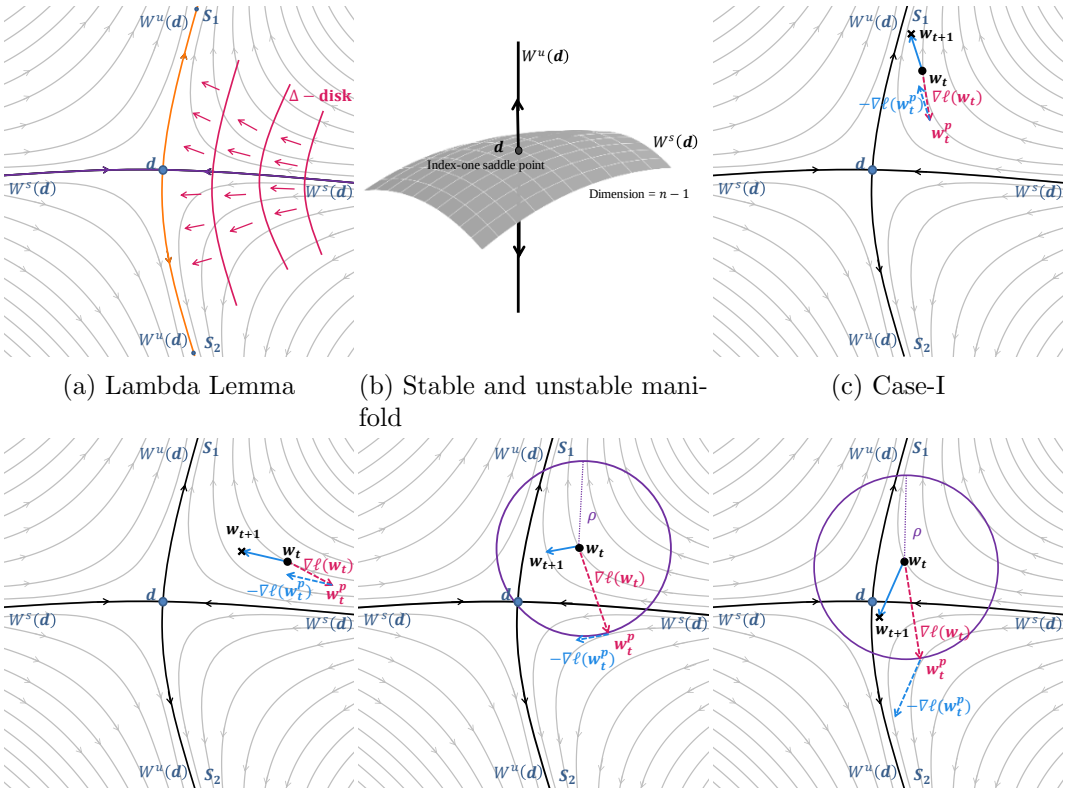
4.3 Methodology

4.3.1 Asymptotic Behavior of SAM Dynamics

To investigate the underlying mechanism of SAM, we first apply a qualitative theory of dynamical systems to identify a case of convergence instability in SAM dynamics. Given the loss function $\ell(\cdot)$, we consider the following gradient flow:

$$\frac{d\boldsymbol{w}}{dt} = -\nabla\ell(\boldsymbol{w}) \tag{4.7}$$

We call a weight vector $\boldsymbol{w}$ that satisfies $\nabla\ell(\boldsymbol{w}) = 0$ *equilibrium point* of system (4.7). Without loss of generality, we assume that (4.7) is *hyperbolic* so that the Hessian matrix of ℓ at each equilibrium point has no zero eigenvalues, which is a generic property that holds for typical loss functions. Then, the Hessian matrix has real



(a) Lambda Lemma (b) Stable and unstable manifold (c) Case-I

(d) Case-II (e) Case-III-i (f) Case-III-ii

Figure 4.2: (a), (b) Phase portrait of system (4.7) associated with the loss function ℓ. (c),(d) Phase portrait of system (4.7) associated with the loss function ℓ (Case-I and Case-II). (e), (f) Phase portrait of system (4.7) associated with the loss function ℓ (Case-III). Case-III causes the convergence instability near a saddle point.

eigenvalues because it is a real symmetric matrix.

If the Hessian matrix $H_\ell(\boldsymbol{w})$ at $\boldsymbol{w}$ has exactly k negative eigenvalues, then it is called an *index-k saddle point* of system (4.7). The *stable manifold* and the *unstable manifold* of an index-k saddle point are defined as

$$W^s(\boldsymbol{w}) := \{\boldsymbol{w}_0 : \lim_{t\to\infty} \boldsymbol{w}_t = \boldsymbol{w}\}, \tag{4.8}$$

$$W^u(\boldsymbol{w}) := \{\boldsymbol{w}_0 : \lim_{t\to-\infty} \boldsymbol{w}_t = \boldsymbol{w}\}. \tag{4.9}$$

where the dimension is $n-k$ and k, respectively. Then, the *basin of attraction* of a stable (index-zero) equilibrium point $\boldsymbol{s}$ can be defined as

$$A(\boldsymbol{s}) := \{\boldsymbol{w}_0 : \lim_{t\to\infty} \boldsymbol{w}_t = \boldsymbol{s}\},$$

Its basin boundary $\partial A(\boldsymbol{s})$ consists of the (closure of) stable manifolds of index-one saddle points $\boldsymbol{d}_i$ on its boundary as

$$\partial A(\boldsymbol{s}) = \bigcup_i \mathrm{cl}(W^s(\boldsymbol{d}_i)).$$

This implies that near the basin boundary, index-one saddle points $\boldsymbol{d}_i$ behave like the attractors.

Given the two adjacent local minima $\mathbf{s}_1$ and $\mathbf{s}_2$, i.e., $\partial A(\mathbf{s}_1) \cap \partial A(\mathbf{s}_2)$, there exists an index-one equilibrium point $\mathbf{d}$ such that the 1-D unstable manifold $W^u(\mathbf{d})$ converges to both $\mathbf{s}_1$ and $\mathbf{s}_2$ with respect to system (4.7). Therefore, it is sufficient to consider the gradient flows near an index-one saddle point $\boldsymbol{d}$ to analyze the behavior of SAM near the basin boundary. We illustrate the basin boundaries for two adjacent local minima $\mathbf{s}_1$ and $\mathbf{s}_2$ and their stable and unstable manifolds in Fig. 4.2a and Fig. 4.2b, respectively.

To this end, we use extensively the Lambda Lemma, a key and deep theory to analyze the behavior of dynamical systems qualitatively. The Lambda Lemma in our version is stated as follows:

Lemma 4.1. ***Lambda Lemma (Guckenheimer and Holmes, 2013; Palis and De Melo, 2012).*** *If Δ is a 1-D disk meeting $W^s(\boldsymbol{d})$ transversely, then the gradient flows of system (4.7) starting from Δ arbitrarily close to $W^u(\boldsymbol{d})$.*

In other words, the gradient flows near the basin boundary but in $A(\mathbf{s}_1)$ directs to the vector sum of $W^u(\boldsymbol{d})$ and $\mathbf{s}_1$. See Figure 4.2a for the illustration. Thus, the

qualitative behavior of the gradient flows starting from $\boldsymbol{w}_t \in A(\boldsymbol{s}_1)$ falls into one of the following cases:

- **Case-I.** When $\boldsymbol{w}_t$ is "away from" an index-one saddle point $\boldsymbol{d}$ and its stable manifold $W^s(\boldsymbol{d})$: $-\nabla\ell(\boldsymbol{w}_t) \sim -\nabla\ell(\boldsymbol{w}_t^p)$ and $\boldsymbol{w}_{t+1}$ converges to $\mathbf{s}_1$ iteratively. (See Figure 4.2c).

- **Case-II.** When $\boldsymbol{w}_t$ is "away from" an index-one saddle point $\boldsymbol{d}$ but near $W^s(\boldsymbol{d})$: $-\nabla\ell(\boldsymbol{w}_t) \sim -\nabla\ell(\boldsymbol{w}_t^p)$ and $\boldsymbol{w}_{t+1}$ approaches $W^u(\boldsymbol{d})$ by the Lambda Lemma iteratively. The subsequent weight vector update iteration will fall into the case of (I) or (III). (See Figure 4.2d).

- **Case-III.** When $\boldsymbol{w}_t$ is near an index-one saddle point $\boldsymbol{d}$ (i.e., $W^s(\boldsymbol{d}) \cap B_\rho(\boldsymbol{w}_t) \neq \emptyset$): $\boldsymbol{w}_t^p \in A(\mathbf{s}_2)$ outside of $A(\mathbf{s}_1)$ and so $-\ell(\boldsymbol{w}_t^p)$ directs to $\boldsymbol{s}_2$ and $W^u(\boldsymbol{d})$ by the Lambda Lemma. In this case, there exist two sub-cases.

  (i) When $\boldsymbol{w}_{t+1} \in A(\mathbf{s}_1)$: $\boldsymbol{w}_{t+1}$ approaches $\boldsymbol{d}$ and $W^s(\boldsymbol{d})$ iteratively and the subsequent weight vector update iteration will fall into the next case (ii). (See Figure 4.2e).

  (ii) When $\boldsymbol{w}_{t+1} \in A(\mathbf{s}_2)$: starting from $\boldsymbol{w}_{t+1}$, the subsequent weight vector update falls into Case-III but the roles of $\mathbf{s}_1$ and $\mathbf{s}_2$ are reversed. Thus, the gradient oscillates near $W^u(\boldsymbol{d})$ iteratively. (See Figure 4.2f).

Therefore, SAM can be hindered by the saddle point $\mathbf{d}$ when the perturbed weight $\boldsymbol{w}^p$ falls into a different basin of attraction across a basin boundary (Case-III), whereas GD smoothly directs to the stable equilibrium point. Furthermore, if

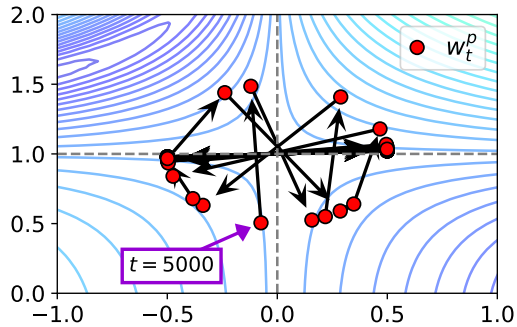


Figure 4.3: Trajectory of $\boldsymbol{w}_t^p$ after the optimization step $t = 5000$, when SAM is beginning to become stuck in the saddle point during SAM optimization in Fig. 4.1. It exhibits exactly the same behavior as that of Case-III in Fig. 4.2.

$\boldsymbol{w}^p$ is consistently located in different basins, $\boldsymbol{w}_t$ will not be able to escape the saddle point under SAM dynamics.

To verify our investigation empirically, we introduce the optimization of the Beale function, a widely adopted benchmark scenario in AdaGrad (Duchi et al., 2011) and Adam (Kingma and Ba, 2014). This function has two benefits. First, it has a single saddle point at $(0, 1)$, which allows us to verify whether the parameter is stuck in the saddle point and the convergence instability near a saddle point. Secondly, it has four basins, as illustrated in Fig. 4.2, with only two basins (top left and bottom right) containing a minimum. We use a learning rate of η=1e-4, as smaller learning rates often perform better on low dimensional problems. We observe similar results for other learning rates that make both GD and SAM converge.

The optimization result is illustrated in Fig. 4.1. While GD successfully converges to the global minimum, SAM is trapped in the saddle point rather than the global minimum. To verify the geometric analysis presented in Fig. 4.2, the trajectory of the perturbed weight $\boldsymbol{w}_t^p$ is plotted in Figure 4.3. After SAM becomes stuck in the

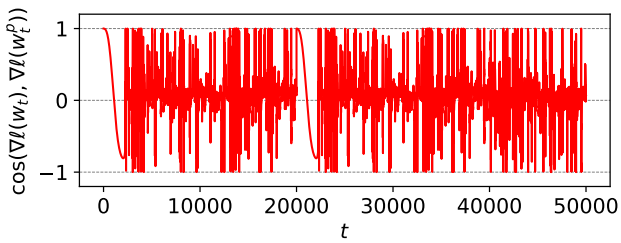


Figure 4.4: Gradient oscillation during SAM optimization in Fig. 4.1. The line corresponds to $\cos(\nabla\ell(\boldsymbol{w}_t), \nabla\ell(\boldsymbol{w}_t^p))$ for optimization step t. The oscillation continues until the end of the optimization.

saddle point, $\boldsymbol{w}_t^p$ continuously crosses the basin boundaries, which is consistent with Fig. 4.2f. Additionally, in Fig. 4.4, the cosine between $\nabla\ell(\boldsymbol{w}_t)$ and $\nabla\ell(\boldsymbol{w}_t^p)$ during the optimization step t is plotted. The cosine value dramatically oscillates between -1 and 1, which is consistent with the gradient oscillation described in Case-III-(ii). These results demonstrate that SAM becomes stuck in the saddle point as if it were a convergence point.

4.3.2 Saddle Point Becomes Attractor in SAM Dynamics

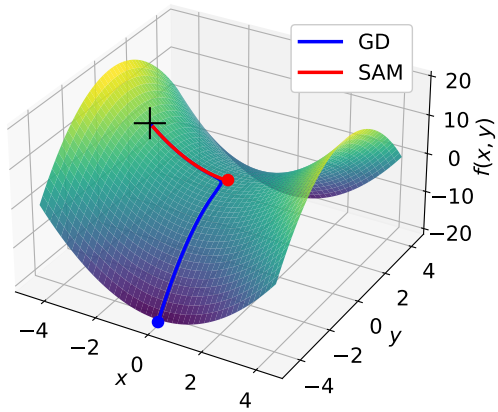
Previously, we observed that SAM becomes stuck in a saddle point even in the simple optimization task as if the saddle point were a convergence point. Motivated by this observation, we here mathematically derive a condition for when a saddle point becomes an attractor under SAM dynamics as follows:

Theorem 4.2. *Let $\boldsymbol{d}$ be an index-one saddle point of system (4.7) with a negative eigenvalue λ_1 of the Hessian matrix $H_\ell(\boldsymbol{d})$ of the loss function ℓ. Then, the saddle point $\boldsymbol{d}$ is an attractor of SAM dynamics in Eq. (4.5) if $\rho \geq -1/\lambda_1$.*

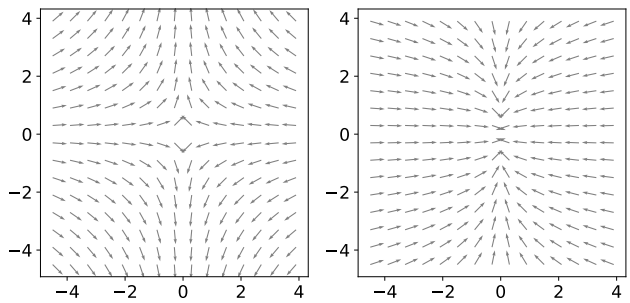
Proof. Given the gradient flow of SAM in Eq. (4.5), we have

$$\frac{d\boldsymbol{w}}{dt} = -\nabla\ell(\boldsymbol{w} + \rho\nabla\ell(\boldsymbol{w})) \tag{4.10}$$

$$\frac{d^2\boldsymbol{w}}{dt^2} = -\nabla^2\ell(\boldsymbol{w} + \rho\nabla\ell(\boldsymbol{w}))[I + \rho\nabla^2\ell(\boldsymbol{w})]. \tag{4.11}$$



(a) Trajectory of GD and SAM



(b) Gradient flow of GD (c) Gradient flow of SAM

Figure 4.5: Optimization on $f(x, y) = x^2 - y^2$ with the saddle point at (0,0). (a) Loss surface with the initial point denoted as the plus symbol $(-3, -\epsilon)$, where $\epsilon = 0.01$. (b) Divergence of the gradient flow of GD near the saddle point. (c) Convergence of the gradient flow of SAM with $\rho = 1.0$ to the saddle point, thus making the saddle point an attractor.

Since the Hessian matrix is a real symmetric matrix, $H_\ell(\boldsymbol{d}) = Q\Lambda Q^T$, where Q is a real orthogonal matrix and $\Lambda = \mathbf{diag}[\lambda_1, \cdots, \lambda_n]^T$ is a real diagonal matrix consisting of eigenvalues as its elements. Given a saddle point $\boldsymbol{d}$, we have $\nabla\ell(\boldsymbol{d}) = 0$ and thus Eq. (4.11) becomes

$$\frac{d^2\boldsymbol{w}}{dt^2}|_{\boldsymbol{w}=\boldsymbol{d}} = -\nabla^2\ell(\boldsymbol{d} + \rho\nabla\ell(\boldsymbol{d}))[I + \rho\nabla^2\ell(\boldsymbol{d})] \tag{4.12}$$

$$= -Q(\Lambda + \rho\Lambda^2)Q^T. \tag{4.13}$$

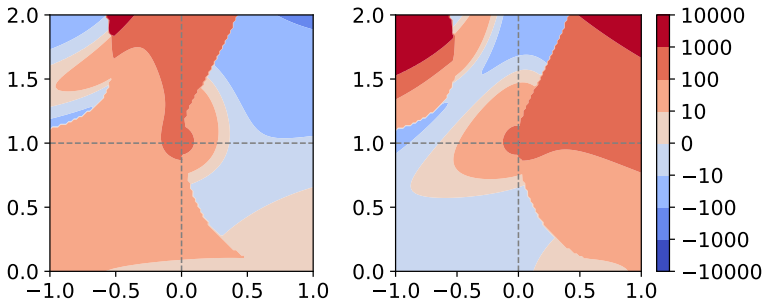
As all the diagonal elements of $\Lambda + \rho\Lambda^2$ are positive, $\boldsymbol{d}$ becomes an attractor of SAM dynamics. □

Notice that the above results can be easily generalized to any type of saddle points provided that

$$\lambda_j + \rho\lambda_j^2 \geq 0 \tag{4.14}$$

for all the negative eigenvalues λ_j at $\boldsymbol{d}$. Moreover, this condition is mild practically since the normalized radius $\tilde{\rho} = \rho/\|\nabla\ell(\boldsymbol{w})\|$ in the standard SAM dynamics (Foret et al., 2020), which becomes very large near the saddle point. This result further suggests that more points can become attractors under SAM dynamic because the term $\rho\Lambda^2$ always results in positive diagonal values. Thus, Theorem 4.2 tells us that a saddle point can become an attractor under SAM dynamics, whereas it is not a general optimum in traditional approaches.

Fig. 4.5 illustrates the empirical verification of Theorem 4.2. We perform an optimization with GD and SAM on a simple function $f(x, y) = x^2 - y^2$, which has a saddle point at $(0, 0)$. In Fig. 4.5a, GD (blue-colored) successfully escapes the saddle point due to the benefit of a good initial point $(-3, \epsilon)$. Specifically, the gradient flow of GD (Fig. 4.5b) demonstrates that the saddle point is not a stable equilibrium point, and thus a slight perturbation at the initial point $\epsilon = 0.01$ is sufficient to help



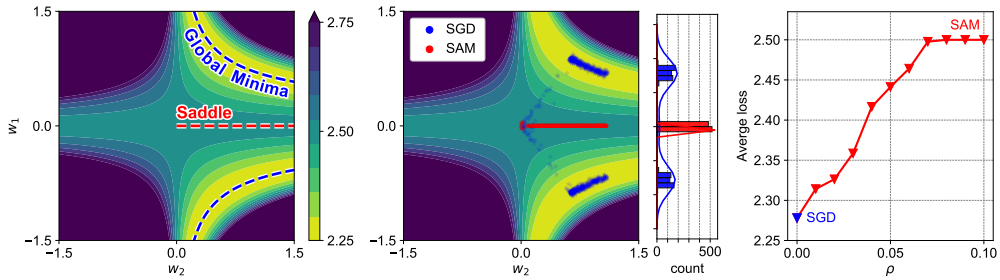
(a) $\lambda_1 + \rho\lambda_1^2$ (b) $\lambda_2 + \rho\lambda_2^2$

Figure 4.6: Value of $\lambda + \rho\lambda^2$ for the Beale function in Fig. 4.1. Near the saddle point at (0, 1), for both eigenvalues of the Hessian, $\lambda + \rho\lambda^2$ is positive, which indicates that the saddle point becomes an attractor.

GD escape the saddle point.

In contrast, SAM becomes stuck in the saddle point (red-colored in Fig. 4.5a). In this optimization, eigenvalues of the Hessian at (0, 0) are $\boldsymbol{\lambda} = \{2, -2\}$ and thus $\lambda + \rho\lambda^2 > 0$ for all eigenvalues $\lambda \in \boldsymbol{\lambda}$. Therefore, by Theorem 4.2, the saddle point will become an attractor under SAM dynamics. The gradient flow (Fig. 4.5c) also supports our theorem as the saddle point has now become a point of convergence, which implies that SAM will not be able to escape it.

We further explain the previous optimization result depicted in Fig. 4.1 by calculating the eigenvalues of the Hessian for each point in the parameter space. For each point in the parameter space, we calculate the eigenvalues of the Hessian $\boldsymbol{\lambda} = \{\lambda_1, \lambda_2\}$ and visualize the heat map of $\lambda + \rho\lambda^2$ in Fig. 4.6. For both $\lambda = \lambda_1$ and λ_2, $\lambda + \rho\lambda^2$ is positive near the saddle point, which is consistent with the behavior of the saddle point as an attractor, as illustrated in Fig. 4.5. This suggests that the convergence instability can arise even for a more complicated loss function because the saddle point may become an attractor under SAM dynamics.



(a) Loss landscape (b) Convergence distribution (c) Average loss

Figure 4.7: Toy neural network experiment. (a) Loss landscape for different values of each neuron. (b) Distributions of converged points for SGD and SAM($\rho = 0.1$) with the marginal distribution of the parameter $\boldsymbol{w}_1$. (c) Average loss of converged points for different ρ. $\rho = 0$ indicates SGD.

4.4 Experiments

4.4.1 Stochastic Behavior of SAM Dynamics

Previously, we confirmed the asymptotic behavior of SAM dynamics and its convergence instability near a saddle point. From now on, we focus on mini-batch sampling, which has been found to provide high performance in various domains (Hoffer et al., 2017; Ziyin et al., 2022), and thus it is a commonly used technique for training deep learning models.

To investigate the behavior of SAM under stochastic system dynamics, we first establish SAM diffusion with the stochastic differential equation. Note that the basis formulations and notations are borrowed from (Risken, 1996; Sato and Nakagawa, 2014; Xie et al., 2020, 2022). We assume that the perturbed weight $\boldsymbol{w}^p$ is precisely calculated with Eq. (4.5), and thus the stochastic differential equation of SAM dynamics is formalized as follows:

$$d\boldsymbol{w} = -\nabla \ell(\boldsymbol{w}^p)dt + [\eta C(\boldsymbol{w}^p)]^{\frac{1}{2}} dW_t, \tag{4.15}$$

where η is a learning rate and $dW_t \sim \mathcal{N}(0, Idt)$ for the identity matrix I. Given the divergence operator $\nabla\cdot$ and the diffusion matrix

$$D(\boldsymbol{w}^p) = \frac{\eta C(\boldsymbol{w}^p)}{2}, \tag{4.16}$$

we use the Fokker-Planck equation that describes the probability density of the weight $\boldsymbol{w}$ as follows (Xie et al., 2020):

$$\frac{\partial P(\boldsymbol{w}, t)}{\partial t} = \nabla \cdot [P(\boldsymbol{w}, t)\nabla \ell(\boldsymbol{w}^p)] + \nabla \cdot \nabla [D(\boldsymbol{w}^p)P(\boldsymbol{w}, t)]. \tag{4.17}$$

Now, we assume the following **second-order Taylor approximation** of the loss function ℓ near saddle points $\boldsymbol{d}$, which holds locally around critical points and is commonly adopted in (Mandt et al., 2017; Xie et al., 2020, 2022):

$$\ell(\boldsymbol{w}) = \ell(\boldsymbol{d}) + \frac{1}{2}(\boldsymbol{w} - \boldsymbol{d})^T H_\ell(\boldsymbol{d})(\boldsymbol{w} - \boldsymbol{d}).$$

Under the assumption, we draw SAM diffusion and prove that SAM escapes the saddle point more slowly than SGD.

Theorem 4.3. ***(SAM diffusion)*** *Given a saddle point $\boldsymbol{d}$ as the initial parameter under the dynamics of Eq. (4.15), under the second-order Taylor approximation assumption near $\boldsymbol{d}$, the probability density function of $\boldsymbol{w}$ after time t is the Gaussian distribution, i.e., $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{d}, Q\mathbf{diag}(\boldsymbol{\sigma}^2(t))Q^T)$ with $\boldsymbol{\sigma}^2(t) = [\sigma_1^2(t), \cdots, \sigma_n^2(t)]^T$ where*

$$\sigma_j^2(t) = \frac{\eta|\lambda_j|}{2B\lambda_j(1+\rho\lambda_j)^2}[1 - \exp(-2\lambda_j(1+\rho\lambda_j)^2 t)] \tag{4.18}$$

for the batch size B and the j-th eigenvalue of $H(\boldsymbol{d})$, λ_j.

Proof. The solution of the Fokker-Planck Equation Eq. (4.17) should be formalized as follows:

$$P(\boldsymbol{w}, t) = (\prod_{j=1}^{n} 2\pi\sigma_j)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{w} - \boldsymbol{d})^T Q\mathbf{diag}(\boldsymbol{\sigma}^2(t))Q^T(\boldsymbol{w} - \boldsymbol{d})\right). \tag{4.19}$$

To be self-contained, we mainly followed Appendix A.1 in (Xie et al., 2022). Let

us denote the loss function of the i-th training sample $\ell_i(\boldsymbol{w})$ among N samples in total. By using the Fisher Information Matrix (FIM) (Pawitan, 2001), the following approximation can be adopted near the critical point $\boldsymbol{d}$ as described in (Jastrzkebski et al., 2017; Zhu et al., 2018; Xie et al., 2022).

$$C(\boldsymbol{w}) \approx \frac{1}{B}\left[\frac{1}{N}\sum_{i=1}^{N}\nabla\ell_i(\boldsymbol{w})\nabla\ell_i(\boldsymbol{w})^T\right] = \frac{1}{B}\text{FIM}(\boldsymbol{w}) \approx \frac{1}{B}[H(\boldsymbol{w})]^+. \tag{4.20}$$

Here, given $A = V\mathbf{diag}([\lambda_1, \cdots, \lambda_n]^T)V^T$, we denote $V\mathbf{diag}([|\lambda_1|, \cdots, |\lambda_n|]^T)V^T$ as $[A]^+$. By the above equation and Assumption 4.4.1, near the critical point, $D(\boldsymbol{w})$ is independent of $\boldsymbol{w}$, i.e.,

$$D = \frac{\eta}{2B}[H]^+. \tag{4.21}$$

where $H(\boldsymbol{d}) = H$ for simplicity.

Following (Xie et al., 2022), without loss of generality, we consider one-dimensional solution. Under the second-order Taylor approximation assumption, we have

$$\begin{aligned}\nabla\ell(\boldsymbol{w}^p) &= \nabla\left[\ell(\boldsymbol{d}) + \frac{1}{2}(\boldsymbol{w}^p - \boldsymbol{d})^T H(\boldsymbol{w}^p - \boldsymbol{d})\right] \\ &= H[\boldsymbol{w} + \rho\nabla\ell(\boldsymbol{w})]\nabla[\boldsymbol{w} + \rho\nabla\ell(\boldsymbol{w})] = H[I + \rho H]^2\boldsymbol{w}.\end{aligned}$$

Then, the right term of Eq. (4.17) can be formalized as

$$\begin{aligned}&\nabla\cdot[P(\boldsymbol{w},t)\nabla\ell(\boldsymbol{w}^p)] + \nabla\cdot\nabla[D(\boldsymbol{w}^p)P(\boldsymbol{w},t)] \\ &= P(\boldsymbol{w},t)H[I+\rho H]^2 - H[I+\rho H]^2\cdot\frac{\boldsymbol{w}}{\sigma^2}P(\boldsymbol{w},t) + D\left(\frac{\boldsymbol{w}^2}{\sigma^4} - \frac{1}{\sigma^2}\right)P(\boldsymbol{w},t) \\ &= \left(1 - \frac{\boldsymbol{w}^2}{\sigma^2}\right)H[I+\rho H]^2P(\boldsymbol{w},t) + D\left(\frac{\boldsymbol{w}^2}{\sigma^4} - \frac{1}{\sigma^2}\right)P(\boldsymbol{w},t) \\ &= \left(-\sigma^2 H[I+\rho H]^2 + D\right)\left(\frac{\boldsymbol{w}^2}{\sigma^4} - \frac{1}{\sigma^2}\right)P(\boldsymbol{w},t).\end{aligned}$$

On the other hand, the left term of Eq. (4.17) can be formalized as

$$\frac{\partial P(\boldsymbol{w},t)}{\partial t} = \frac{1}{2}\left(\frac{\boldsymbol{w}^2}{\sigma^4} - \frac{1}{\sigma^2}\right)P(\boldsymbol{w},t)\frac{\partial\sigma^2}{\partial t}.$$

Thus, the solution of Eq. (4.17) is

$$\frac{\partial\sigma^2}{\partial t} = 2D - 2\sigma^2 H[I+\rho H]^2,$$

By using the initial condition $\sigma^2(0) = 0$ and Eq. (4.21), we can obtain

$$\sigma_j^2(t) = \frac{\eta|\lambda_j|}{2B\lambda_j(1+\rho\lambda_j)^2}[1 - \exp(-2\lambda_j(1+\rho\lambda_j)^2 t)].$$

□

Let us denote the mean squared displacement of the displacement $\Delta \boldsymbol{w}_j(t) = \boldsymbol{w}_j(t) - \boldsymbol{w}_j(0)$ by $\langle \Delta \boldsymbol{w}_j^2(t) \rangle$. Then, we have the following result.

Corollary 4.4. ***(SAM escapes saddle point more slowly than SGD)*** *The mean squared displacement of SAM, denoted by $\Delta_{SGD} := \langle \Delta \boldsymbol{w}_j^2(t) \rangle_{SGD}$, is smaller than that of SGD, i.e., $\Delta_{SAM} := \langle \Delta \boldsymbol{w}_j^2(t) \rangle_{SAM}$, near the saddle point $\boldsymbol{d}$, which satisfies the following inequality:*

$$\Delta_{SGD} - \Delta_{SAM} = \frac{2\eta t^2 |\lambda_j|^3}{B}\rho > 0. \tag{4.22}$$

Proof. By the definition of the mean squared displacement, $\langle \Delta \boldsymbol{w}_j^2(t) \rangle$ is equal to $= \sigma_j^2(t)$. Thus, given Theorem 4.3,

$$\begin{aligned}
\Delta_{SGD} &= \frac{\eta|\lambda_j|}{2B\lambda_j}[1 - \exp(-2\lambda_j t)] \\
\Delta_{SAM} &= \frac{\eta|\lambda_j|}{2B\lambda_j(1+\rho\lambda_j)^2}[1 - \exp(-2\lambda_j(1+\rho\lambda_j)^2 t)].
\end{aligned}$$

As $|\lambda_j| t \ll 1$ near ill-conditioned saddle points (Xie et al., 2022), the difference $\Delta_{SGD} - \Delta_{SAM}$ can be formalized as follows:

$$\begin{aligned}
\Delta_{SGD} - \Delta_{SAM} &\approx \frac{\eta|\lambda_j|}{2B\lambda_j(1+\rho\lambda_j)^2}[(1+\rho\lambda_j)^2[2\lambda_j t - 2(\lambda_j t)^2] \\
&\qquad - [2\lambda_j(1+\rho\lambda_j)^2 t - 2(\lambda_j(1+\rho\lambda_j)^2 t)^2]] \\
&= \frac{\eta\lambda_j|\lambda_j|}{B}\left[(t+\rho\lambda_j t)^2 - t^2\right] = \frac{\eta\lambda_j|\lambda_j|}{B}\left[2\rho\lambda_j t^2 + (\rho\lambda_j t)^2\right] \\
&\approx \frac{2\eta t^2|\lambda_j|^3\rho}{B}.
\end{aligned}$$

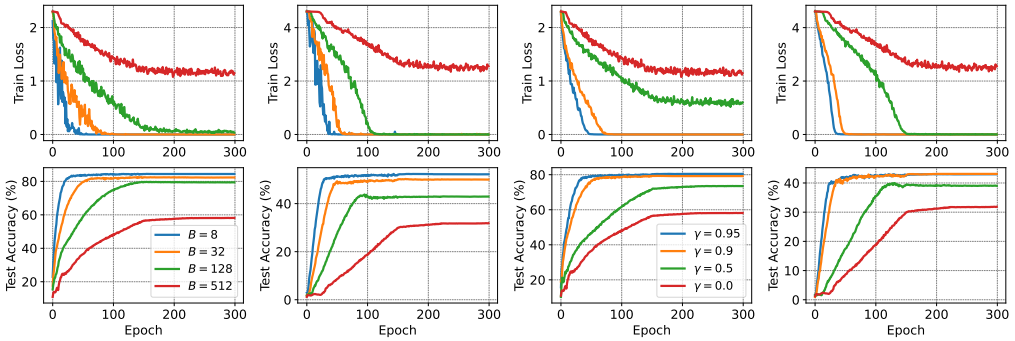
□

Corollary 4.4 tells us that SAM requires more time to escape the saddle point $\boldsymbol{d}$ compared to SGD, since Δ_{SGD} is always larger than Δ_{SAM}. Interestingly, this

is consistent with the results from Sections 4.3.1 and 4.3.2, which show that as ρ increases, the perturbed weight is more likely to be placed in the disjoint basin boundaries (Section 4.3.1) and the saddle point is more likely to become an attractor (Section 4.3.2). In Corollary 4.4, the difference between mean square displacements $\Delta_{SGD} - \Delta_{SAM}$ increases as ρ increases, indicating that the gap between the required time to escape the saddle point between SGD and SAM increases as ρ increases.

To validate our theoretical results in stochastic dynamical systems, we first conduct an experiment using a neural network and artificial settings as proposed by Ziyin et al. (2022). This experiment involves a two-layer neural network with one neuron, with a non-linear coordinate-wise activation function φ. Specifically, $\varphi(x) = x^2$ is used. The training set consists of an input x which is fixed to 1, and $y \in \{-1, 2\}$ with probability of 0.5. The loss function is the mean squared error, and thus the loss landscape yields global minima $\ell(\boldsymbol{w}) = 2.25$ for $\{(w_1, w_2)|w_1^2 = 1/2w_2\}$ and saddle points $\ell(\boldsymbol{w}) = 2.50$ for $\{(w_1, w_2)|w_1 = 0, w_2 \geq 0\}$, as illustrated in Fig. 4.7a. Due to the presence of inherent data uncertainty in this experiment, we can expect more practical results than those from previous experiments. Considering Theorem 4.3, we initialize the parameters near the saddle point, w_1 in the range $[-0.1, 0.1]$ and w_2 in the range $[0, 1]$ uniformly. Other training settings remain the same as those in (Ziyin et al., 2022).

Fig. 4.7b shows the converged parameter distributions for SGD and SAM over 1,000 random seeds. The results demonstrate that the converged parameters of SAM are mostly saturated in the saddle point area, whereas SGD successfully converges to the global minima. This implies that the convergence instability near a saddle point is present even in practical training settings with neural networks.



(a) (CIFAR-10) Batch size (b) (CIFAR-100) Batch size (c) (CIFAR-10) Momentum (d) (CIFAR-100) Momentum

Figure 4.8: Effect of varying batch size B and momentum γ. To measure the pure effect of batch size and momentum, batch normalization and data augmentation are turned off. A smaller batch size and larger momentum lead to better performance of SAM.

Additionally, we plot the average loss of converged points of SAM with varying ρ (Fig. 4.7c). Here, we normalize the radius with the gradient norm during the optimization for easier comparison with prior studies (Foret et al., 2020; Zhuang et al., 2021), but we observe similar results for the constant radius ρ. It is worth noting that the range of ρ in Fig. 4.7c is commonly used in other benchmark experiments such as the CIFAR classifications (Foret et al., 2020; Zhuang et al., 2021). As ρ increases, the average loss of SAM increases. Furthermore, for $\rho > 0.7$, the parameters always become stuck in the saddle point area, resulting in an average loss of 2.50. This result is in agreement with Corollary 4.4, as a larger ρ makes SAM more difficult to escape the saddle point in terms of diffusion, causing the convergence instability near a saddle point from the stochastic dynamic systems view.

4.4.2 Convergence Instability and Training Tricks

The convergence instability of SAM near saddle points, which we observed under both asymptotic and stochastic dynamical systems in previous sections, can lead to suboptimal minimum problems and performance degradation (Du et al., 2017; Kleinberg et al., 2018a; Ziyin et al., 2022). However, this is in contradiction to the claims of prior studies (Foret et al., 2020; Zhuang et al., 2021), which state that SAM and its variants generally perform better than other methods on various benchmark datasets. We theoretically and empirically demonstrate that often used training tricks, such as momentum and batch size, not only help SAM to escape the saddle point but are also the key to its success.

First, we extend SAM diffusion presented in Theorem 4.3 to include momentum and investigate the relationship between the mean squared displacement of SAM, momentum, and batch size.

Theorem 4.5. ***(SAM diffusion, momentum, and batch size)*** *Given a momentum hyper-parameter γ and batch size B, the mean squared displacement of SAM is given by*

$$\Delta_{SAM} = C_1 \frac{(1 - e^{-C_2(1-\gamma)})^2}{(1-\gamma)^3 B} + C_3 \frac{(1 - e^{-\frac{C_4}{1-\gamma}})}{(1-\gamma)B}, \tag{4.23}$$

where $C_1 = \frac{\eta^2 |\lambda_j|}{2}$, $C_2 = \frac{\eta}{t}$, $C_3 = \frac{\eta |\lambda_j|}{2\lambda_j (1+\rho\lambda_j)^2}$, and $C_4 = 2\lambda_j (1+\rho\lambda_j)^2 t$ are positive constants and λ_j denotes eigenvalue of the Hessian matrix $H_\ell(\boldsymbol{d})$ of loss function ℓ at saddle point $\boldsymbol{d}$. Therefore, Δ_{SAM} ***increases as (1) momentum increases and/or (2) batch size decreases.*** *Furthermore, as $(1-\gamma)B \to 0$, we have*

$$\Delta_{SAM} \propto \frac{1}{(1-\gamma)B}. \tag{4.24}$$

Proof. Similar to Theorem 4.3, we consider the on-dimensional case near a critical point. As the momentum dynamics with a momentum γ and dampening τ can be

written as follows:

$$\begin{aligned}\boldsymbol{m}_t &= \gamma \boldsymbol{m}_{t-1} + (1-\tau)\boldsymbol{g}_t \\ \boldsymbol{w}_{t+1} &= \boldsymbol{w}_t - \eta \boldsymbol{m}_t\end{aligned}$$

We let $dt = \eta$, $\phi = \frac{1-\gamma}{dt}$, and $M = \frac{dt}{1-\tau}$. Then, the stochastic differential equation and its Fokker-Planck Equation becomes

$$\begin{aligned}M d\dot{\boldsymbol{w}} &= -\phi d\boldsymbol{w} - \frac{\partial \ell(\boldsymbol{w})}{\partial \boldsymbol{w}} dt + [2D]^{1/2} dW_t \\ \frac{\partial P(\boldsymbol{w}, \boldsymbol{b}, t)}{\partial t} &= -\nabla_{\boldsymbol{w}} \cdot [\boldsymbol{b} P(\boldsymbol{w}, \boldsymbol{b}, t)] + \nabla_{\boldsymbol{b}} [\phi \boldsymbol{m} + M^{-1} \nabla_{\boldsymbol{w}} \ell(\boldsymbol{w})] P(\boldsymbol{w}, \boldsymbol{b}, t) \\ &\quad + \nabla_{\boldsymbol{b}} \cdot M^{-2} D \cdot \nabla_{\boldsymbol{b}} P(\boldsymbol{w}, \boldsymbol{b}, t).\end{aligned}$$

where $\dot{\boldsymbol{w}} := \frac{d\boldsymbol{w}}{dt}$. Further, combining the proof of Theorem 2 in (Xie et al., 2022) and Theorem 4.3, we have the mean squared displacement of SAM with ρ for dampening $\tau = 0$,

$$\begin{aligned}\langle \Delta \boldsymbol{w}_j^2(t) \rangle &= \frac{\eta |\lambda_j|}{2\phi^3 M^2 B} [1 - \exp(-\phi t)]^2 \\ &\quad + \frac{\eta |\lambda_j|}{2\phi M B \lambda_j (1+\rho\lambda_j)^2} \left[1 - \exp\left(-\frac{2\lambda_j (1+\rho\lambda_j)^2 t}{\phi M} \right) \right] \\ &= \frac{\eta^2 |\lambda_j|}{2(1-\gamma)^3 B} \left[1 - \exp\left(-\frac{1-\gamma}{\eta} t \right) \right]^2 \\ &\quad + \frac{\eta |\lambda_j|}{2(1-\gamma) B \lambda_j (1+\rho\lambda_j)^2} \left[1 - \exp\left(-\frac{2\lambda_j (1+\rho\lambda_j)^2 t}{1-\gamma} \right) \right].\end{aligned}$$

Let $C_1 = \frac{\eta^2 |\lambda_j|}{2}$, $C_2 = \frac{\eta}{t}$, $C_3 = \frac{\eta |\lambda_j|}{2\lambda_j (1+\rho\lambda_j)^2}$, and $C_4 = 2\lambda_j (1+\rho\lambda_j)^2 t$. Note that C_1, C_2, C_3, and C_4 are positive. The above equation can be reformulated as follows:

$$\langle \Delta \boldsymbol{w}_j^2(t) \rangle = \frac{C_1}{(1-\gamma)^3 B} \left[1 - \exp\left(-C_2 (1-\gamma) \right) \right]^2 + \frac{C_3}{(1-\gamma) B} \left[1 - \exp\left(-\frac{C_4}{1-\gamma} \right) \right].$$

The second term in the right-hand side is an increasing function with respect to $\gamma \in [0, 1]$, since both $\frac{C_3}{(1-\gamma)}$ and $1 - \exp\left(-\frac{C_4}{1-\gamma} \right)$ are increasing functions with respect to γ. For the first term in the right-hand, we use the following function.

$$h(\gamma) = \frac{1}{(1-\gamma)^3} \left[1 - e^{-C_2 (1-\gamma)} \right]^2.$$

Table 4.1: (CIFAR-10) Test accuracy for different momentum γ and radius ρ. The bold and underlined numbers denote the maximum and minimum accuracy for each ρ, respectively. Batch normalization and data augmentation are turned on.

SAM	Momentum γ			Gap
Radius ρ	0.0	0.9	0.95	(Max - Min)
0.01	93.77±0.08	**94.99±0.09**	94.94±0.09	1.22
0.05	93.56±0.03	**95.01±0.11**	94.94±0.11	1.45
0.1	92.29±0.08	94.84±0.15	**95.08±0.07**	2.80
0.5	86.20±1.66	88.76±1.14	**91.98±0.03**	5.79

We have $h(0) = (1 - e^{-C_2})^2 > 0$ and

$$h'(\gamma) = \frac{e^{-2C_2(1-\gamma)}[e^{C_2(1-\gamma)} - 1]}{(1-\gamma)^4} \left[3(e^{C_2(1-\gamma)} - 1) - 2C_2(1-\gamma)\right] > 0,$$

since $e^{C_2(1-\gamma)} - 1 \geq C_2(1-\gamma)$ for $\gamma \in [0, 1]$ and $C_2 > 0$. Therefore, $h(\gamma)$ is an increasing function with respect to $\gamma \in [0, 1]$. Thus, $\langle \Delta \boldsymbol{w}_j^2(t) \rangle$ is also an increasing function with respect to γ. Furthermore, as $(1-\gamma)B \to 0$, we have

$$\begin{aligned}\Delta_{SAM} &\approx \frac{C_1}{(1-\gamma)^3 B}\left[1 - (1 - C_2(1-\gamma))\right]^2 + \frac{C_3}{(1-\gamma)B}\left[1 - (1 - \frac{C_4}{1-\gamma})\right] \\ &\approx \frac{C_1 C_2^2 (1-\gamma)^2}{(1-\gamma)^3 B} + \frac{C_3}{(1-\gamma)B} = \frac{C_1 C_2^2 + C_3}{(1-\gamma)B} \propto \frac{1}{(1-\gamma)B}.\end{aligned}$$

□

Theorem 4.5 tells us that increasing momentum γ and decreasing batch size B can reduce the time to escape a saddle point. This result is consistent with the work of Andriushchenko and Flammarion (2022), which observed that a smaller batch size significantly increases the performance of SAM. While we derived the advantage of small batch size from the diffusion term, Andriushchenko and Flammarion (2022) provided theoretical proof of the benefit of a small batch size from the concept of implicit bias. Integrating these theoretical analyses might be possible, but we leave it as future work.

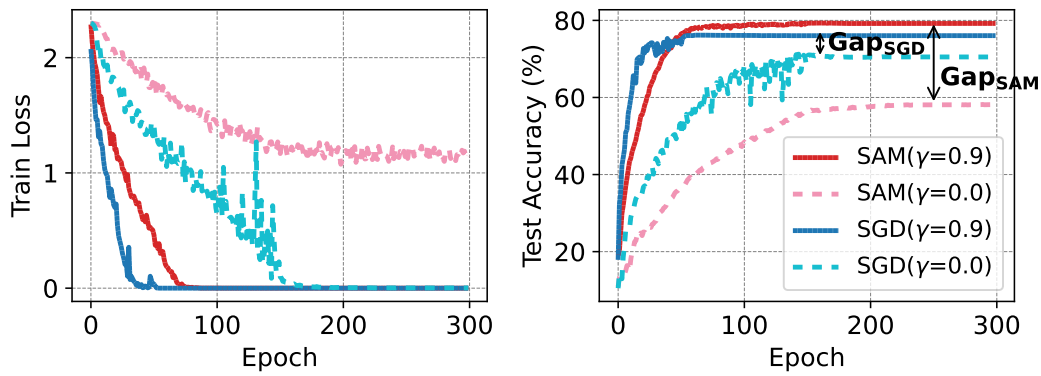


Figure 4.9: (CIFAR-10) Effect of momentum γ for SGD and SAM. Batch normalization and data augmentation are turned off. The momentum increases more dramatically the performance of SAM compared to that of SGD.

To empirically verify Theorem 4.5, we evaluate the effects of momentum and batch size on the commonly used benchmarks, CIFAR-10 and CIFAR-100. To compare the pure effect of momentum and batch size, we turn off both batch normalization and data augmentation. We fix the radius $\rho = 0.1$ and use the momentum $\gamma = 0.0$ and batch size $B = 512$ as default, and then vary the batch size and momentum independently. We normalize the radius with the gradient norm during the training phase, but we observe similar results for the constant radius ρ similar to (Andriushchenko and Flammarion, 2022). All the models are trained for 300 epochs to ensure the convergence.

Fig. 4.8 provides a summary of the training loss and test accuracy for each batch size and momentum. Figures 4.8a and 4.8b demonstrate that SAM struggles to minimize the training loss as the batch size increases. Notably, for a batch size of $B = 512$, SAM displays convergence instability with the train loss exceeding 1, which results in a poor generalization performance of less than 60% accuracy.

In addition, momentum has a significant impact on the training loss and test

accuracy (Figures 4.8c and 4.8d). As momentum increases, both the train loss and test accuracy increase. This effect is particularly pronounced when compared to that on SGD. Fig. 4.9 shows the train loss and test accuracy for SAM and SGD with $\gamma = 0.0$ and 0.9. The momentum improves the accuracy of SGD by approximately 5%, whereas it enhances the accuracy of SAM by more than 20%.

Considering Corollary 4.4, which demonstrates that the mean square displacement of SAM increases as ρ increases, Fig. 4.9 implies that a higher momentum could be beneficial for SAM as ρ increases. We evaluate the test accuracy of SAM for different radius ρ and momentum γ. Here, to obtain their best performance, we turn on batch normalization and data augmentation. Table 4.1 shows the average and standard deviation of test accuracy over three different random seeds. The minimum accuracies are observed only for $\gamma = 0.0$, whereas the best performance is observed for $\gamma = 0.9$ or 0.95. Furthermore, as ρ increases, the gap between the best and worst performances increases, and a higher momentum yields better performance. Considering the fact that $\rho = 0.1$ with $\gamma = 0.95$ yields the best accuracy over all combinations, using a higher momentum with a larger ρ can be beneficial to improve the generalization performance under SAM dynamics.

4.5 Chapter Summary

SAM is an arguably promising training method with improved generalization abilities. To gain insight into its mechanism, we analyzed the behavior of SAM, which revealed the presence of convergence instability near a saddle point. Through theoretical and empirical analyses, we demonstrated that this convergence problem can occur in various tasks, from a simple optimization to complicated neural network

training. Additionally, we demonstrated the importance of training tricks, such as momentum and batch size, in relation to the diffusion theory. We believe that our work sheds light on the mechanism of SAM and hope to be integrated in future work and develop new methods to improve the generalization performance.

Chapter 5

Sharpness-Aware Minimization with Multi-Ascent

5.1 Chapter Overview

Modern deep neural networks are highly overparameterized so that models can achieve near-zero training loss. However, minimizing training loss is not sufficient to achieve better *generalization* (Ishida et al., 2020; Foret et al., 2020). To understand and further improve the generalization performance of neural networks, recent studies discuss the geometric properties of loss surface and argue that the *sharpness*, which measures the flatness of a minimum, might be a core component of generalization (McAllester, 1999; Dinh et al., 2017; Keskar et al., 2017; Jiang et al., 2019). Extensive theoretical and empirical analyses on the sharpness lead to new sharpness-aware training methods to improve generalization performance (Izmailov et al., 2018; He et al., 2019a; Chaudhari et al., 2019).

Recently, Foret et al. (2020) proposed Sharpness-Aware Minimization (SAM) that achieves state-of-the-art generalization performance. Given weight $\boldsymbol{w}$ and a loss function ℓ, SAM is designed to minimize $\max_{\|\boldsymbol{v}\|=1} \ell(\boldsymbol{w} + \rho\boldsymbol{v})$, which is the maximum loss within radius ρ in the weight space. To solve the inner maximization problem, Foret et al. (2020) proposed a single-step gradient-based approximation, which results in a perturbed weight $\boldsymbol{w}^{p_1} = \boldsymbol{w} + \rho\boldsymbol{v}_1$ with ascent direction $\boldsymbol{v}_1 =$

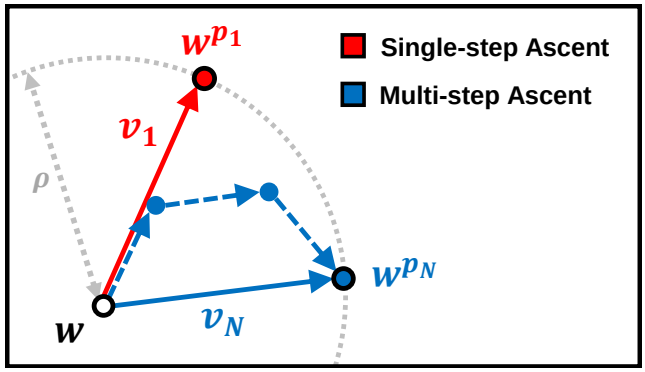


Figure 5.1: Illustration of ascent step with different iterations.

$\nabla\ell(\boldsymbol{w})/\|\nabla\ell(\boldsymbol{w})\|$. To find a higher perturbed loss, the authors also investigated a multi-step gradient-based approximation. As illustrated in Fig. 5.1, the multi-step ascent with N iterations yields $\boldsymbol{v}_N$ and $\boldsymbol{w}^{p_N}$. However, the experimental results in (Foret et al., 2020; Wu et al., 2020b; Andriushchenko and Flammarion, 2022) show that multi-step ascent during training does not provide better performance. Considering that a multi-step optimization generally induces a better approximation than a single-step optimization, this raises the question: *"Why better approximation of the inner maximization does not bring any benefit to SAM?"*

In this chapter, we take a closer look at the number of ascent steps in SAM. We first analyze an effect of the number of ascent steps on the inner maximization, then investigate the difference between models trained with single-step and multi-step ascent. Unlike the previous statements that multi-step ascent does not affect the model, we discover that different number of ascent steps yields different perturbed loss surfaces. Fig. 5.2 illustrates the difference between models trained with single-step and multi-step ascent. Let us consider two different trained weights $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ with different number of ascent steps $N = 1$ and 2. Then, if we evaluate their

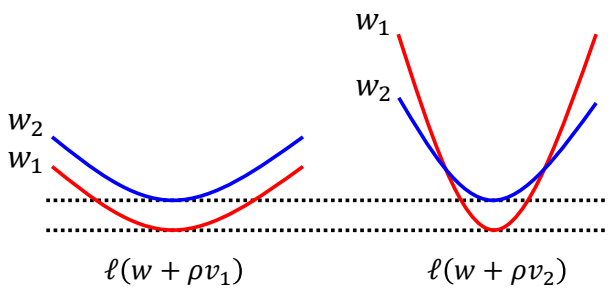


Figure 5.2: Perturbed loss $\ell(\boldsymbol{w}+\rho\boldsymbol{v})$ for each $\boldsymbol{v}$ with different number of ascent steps. If we consider the single-step ascent $\boldsymbol{v}_1$, $\boldsymbol{w}_1$ shows lower perturbed losses than $\boldsymbol{w}_2$ for all radius ρ (left). However, with multi-step ascent $\boldsymbol{v}_2$, $\boldsymbol{w}_2$ has lower perturbed losses than $\boldsymbol{w}_1$ over certain radius (right).

perturbed loss with $\boldsymbol{v}_1$ (corresponds to the red arrow in Fig. 5.1), then $\boldsymbol{w}_1$ is better than $\boldsymbol{w}_2$ because it has a lower loss for all radius ρ. However, if we consider multi-step ascent $\boldsymbol{v}_2$ (corresponds to the blue arrow in Fig. 5.1), then $\boldsymbol{w}_2$ shows a lower loss than $\boldsymbol{w}_1$ over certain radius.

Based on these observations, we demonstrate that perturbed weights obtained by different number of ascent steps have their unique gradient information and bring different effects on the perturbed loss surface. Inspired by the gradient diversity in perturbed weight space, we propose a new training method that utilizes all gradient information during multi-step ascent. We verify the theoretical and empirical advantages of the proposed method, and demonstrate that it improves generalization performance across various models and datasets.

5.2 Preliminaries

Sharpness-Aware Minimization (SAM) seeks flat minima possessing low loss values in its neighborhood (Foret et al., 2020). The key idea is to minimize the maximum

loss within radius ρ in the weight space as follows:

$$\min_{\boldsymbol{w}} \max_{\|\boldsymbol{v}\|=1} \ell(\boldsymbol{w} + \rho\boldsymbol{v}) \tag{5.1}$$

To solve this min-max optimization problem sequentially, SAM uses *ascent step* and *descent step* for the inner maximization and outer minimization, respectively. Simply, SAM calculates the perturbed weight $\boldsymbol{w}^p = \boldsymbol{w} + \rho\boldsymbol{v}$ where $\boldsymbol{v}$ is an *ascent direction*, and then, SAM minimizes the loss of the perturbed weight, $\ell(\boldsymbol{w}^p)$, by utilizing the gradient $\nabla_{\boldsymbol{w}^p}\ell(\boldsymbol{w}^p)$. At time t, SAM is formulated as follows:

$$\underline{\text{Ascent step}}: \quad \boldsymbol{w}_t^p = \boldsymbol{w}_t + \rho\boldsymbol{v} \tag{5.2}$$

$$\underline{\text{Descent step}}: \quad \boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta\nabla\ell(\boldsymbol{w}_t^p) \tag{5.3}$$

where, η is the learning rate and $\nabla\ell(\boldsymbol{\theta}) := \nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta})$ for any $\boldsymbol{\theta}$ unless otherwise specified.

To calculate an ascent direction $\boldsymbol{v}$ in Eq. (5.2), Foret et al. (2020) basically recommends to use $\boldsymbol{v}_1 = \nabla\ell(\boldsymbol{w})/\|\nabla\ell(\boldsymbol{w})\|$ to $\boldsymbol{w}$ to maximize the loss $\ell(\boldsymbol{w})$. This *single-step ascent* can be formalized as follows:

$$\boldsymbol{w}_t^{p_1} = \boldsymbol{w}_t + \rho \cdot \frac{\nabla\ell(\boldsymbol{w}_t)}{\|\nabla\ell(\boldsymbol{w}_t)\|} \tag{5.4}$$

In addition, Foret et al. (2020) also explored *multi-step ascent* direction $\boldsymbol{v}_N$ with the number of inner maximization iterations N as follows:

$$\boldsymbol{w}^{p_N^n} = \boldsymbol{w}^{p_N^{n-1}} + \rho_n \cdot \frac{\nabla\ell(\boldsymbol{w}^{p_N^{n-1}})}{\|\nabla\ell(\boldsymbol{w}^{p_N^{n-1}})\|} \tag{5.5}$$

$$\boldsymbol{v}_N = \frac{\boldsymbol{w}^{p_N^N} - \boldsymbol{w}}{\|\boldsymbol{w}^{p_N^N} - \boldsymbol{w}\|} \tag{5.6}$$

where $\boldsymbol{w}^{p_N^0} = \boldsymbol{w}$, $\boldsymbol{w}^{p_N} = \boldsymbol{w} + \rho\boldsymbol{v}_N$, and ρ_n denotes the radius at each iteration for

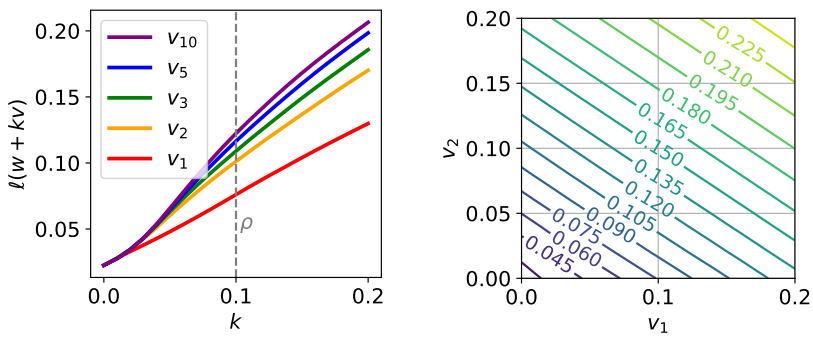
$n \in \{1, ..., N\}$. Unless specified otherwise, $\rho_n = \rho/N$.

However, Foret et al. (2020) argued that using multi-step ascent during training is not effective as single-step ascent, which is also observed in (Wu et al., 2020b). Noteworthy, the concurrent work (Andriushchenko and Flammarion, 2022) focuses the effect of batch-size used in ascent step and m-sharpness (Foret et al., 2020). They also explored the effectiveness of multi-step ascent, but it does not improve the performance of SAM. This phenomenon is particularly interesting because multi-step ascent does not bring any improvement, while it would give a higher perturbed loss in Eq. (5.1). Thus, in this study, we reveal an effect of the number of ascent steps and investigate the characteristics of multiple perturbed weights and their gradients.

There has been some variants of SAM (Kwon et al., 2021; Zhuang et al., 2021; Du et al., 2022) that improve the performance in different ways. Kwon et al. (2021) proposed Adaptive Sharpness-Aware Minimization (ASAM) that adjusts parameter re-scaling in the inner maximization regions. Du et al. (2022) proposed Efficient Sharpness-Aware Minimization (ESAM), which focused on reducing the computational cost of SAM. Zhuang et al. (2021) proposed surrogate Gap guided Sharpness-Aware Minimization (GSAM) that minimizes the gap $\ell(\boldsymbol{w}^p) - \ell(\boldsymbol{w})$ instead of $\ell(\boldsymbol{w}^p)$. However, while all of them focused on single-step ascent, our work sheds new light on understanding the algorithm of SAM by considering both single-step and multi-step ascent. Furthermore, deviating from the previous studies that considered a perturbed loss with only one ascent step, we suggest that different number of ascent steps should be considered for measuring the perturbed loss.

Table 5.1: Analysis on different numbers of ascent steps during training and evaluation. The number with the bold denotes the minimum perturbed loss. Evaluated on whole test examples.

	N_{tr}	Accuracy (%)	Time / iter (sec)	Perturbed loss with N_{te} ascent steps 0	1	3	5
CIFAR -10	1	96.54$_{\pm 0.07}$	0.13$_{\pm 0.00}$	**0.018$_{\pm 0.002}$**	0.097$_{\pm 0.004}$	0.125$_{\pm 0.005}$	0.130$_{\pm 0.005}$
	2	96.58$_{\pm 0.13}$	0.19$_{\pm 0.00}$	0.019$_{\pm 0.000}$	**0.095$_{\pm 0.001}$**	**0.119$_{\pm 0.001}$**	0.124$_{\pm 0.001}$
	3	96.61$_{\pm 0.01}$	0.23$_{\pm 0.01}$	0.021$_{\pm 0.001}$	**0.095$_{\pm 0.004}$**	**0.119$_{\pm 0.005}$**	**0.123$_{\pm 0.005}$**
CIFAR -100	1	85.44$_{\pm 0.07}$	1.10$_{\pm 0.00}$	**0.166$_{\pm 0.008}$**	**0.744$_{\pm 0.018}$**	1.285$_{\pm 0.022}$	1.416$_{\pm 0.021}$
	2	84.91$_{\pm 0.08}$	1.55$_{\pm 0.00}$	0.226$_{\pm 0.008}$	0.810$_{\pm 0.007}$	1.198$_{\pm 0.026}$	1.280$_{\pm 0.030}$
	3	85.17$_{\pm 0.01}$	2.04$_{\pm 0.00}$	0.251$_{\pm 0.004}$	0.816$_{\pm 0.019}$	**1.180$_{\pm 0.018}$**	**1.262$_{\pm 0.027}$**



(a) $\ell(\boldsymbol{w} + k\boldsymbol{v}_N)$ (b) $\ell(\boldsymbol{w} + x\boldsymbol{v}_1 + y\boldsymbol{v}_2)$

Figure 5.3: (CIFAR-10) Loss of perturbed weights in the direction of $\boldsymbol{v}_N$. (a) Perturbed loss in the direction of $\boldsymbol{v}_N$ with different number of ascent steps N. As N increases, a perturbed loss value also increases. (b) Loss of perturbed weights generated by $\boldsymbol{v}_1$ (x-axis) and $\boldsymbol{v}_2$ (y-axis). $\boldsymbol{v}_2$ increases a perturbed loss value more rapidly than any linear combination.

5.3 Methodology

5.3.1 Revisiting Number of Ascent Steps in SAM

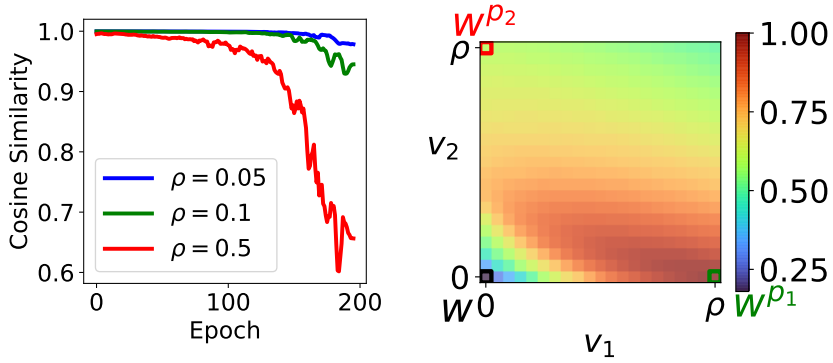
Multi-step ascent leads to a better inner maximization First, we verify whether a multi-step optimization yields a better approximation than a single-step optimization for the inner maximization in Eq. (5.1). To do this, we manipulate the number of ascent steps N for generating $\boldsymbol{v}_N$ with a model trained with SAM. Here,

we use ResNet-18 trained on CIFAR-10 with $\rho = 0.1$. Then, for each $\boldsymbol{v}_N$, we plot the loss of perturbed weights $\boldsymbol{w} + k\boldsymbol{v}_N$, which is the linear interpolation of $\boldsymbol{w}$ and $\boldsymbol{w} + \rho\boldsymbol{v}_N$. To consider a wider range of the weight space, we set $k \in [0, 2\rho]$. First, we measure $\ell(\boldsymbol{w} + k\boldsymbol{v}_N)$ in Fig. 5.3a. As N increases, the loss value also increases. In addition, a higher N induces a higher loss value for every k. This tells us that multi-step ascent yields a better approximation for the inner maximization in Eq. (5.1).

In addition, we plot perturbed losses generated by linear combinations of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ in Fig. 5.3b. The direction that generates the maximum loss increase is $\boldsymbol{v}_2$ rather than any linear combination of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. We note that similar results are observed with a vanilla trained model. Thus, with widely used ρ, the loss surface near weight $\boldsymbol{w}$ and its ρ-neighborhood is locally linear for $\boldsymbol{v}_N$.

Multi-step ascent during training provides a lower perturbed loss for multi-step ascent directions Now, we evaluate the performance of the models with different number of ascent steps N_{tr} during training. Indeed, as shown in Table 5.1, their test accuracy are not significantly differentiated even the computation cost linearly increases as N_{tr} increases. This is consistent with the results in previous studies (Foret et al., 2020; Wu et al., 2020b).

However, considering the objective of SAM in Eq. (5.1), this result is interesting because a higher N_{tr} would yield a better inner maximization approximation as we observed in Fig. 5.3. To further investigate this phenomenon, we vary the number of ascent steps N_{te} during evaluation for each trained model. Interestingly, we discover that they show different behavior in their perturbed losses. Specifically, SAM with



(a) $\text{Cos}(\nabla\ell(\boldsymbol{w}^{p_1}), \nabla\ell(\boldsymbol{w}^{p_2}))$ (b) $\text{Cos}(\nabla\ell(\boldsymbol{w}^{p_1}), \nabla\ell(\cdot))$

Figure 5.4: (CIFAR-10) Cosine similarity between gradients of diverse perturbed weights. (a) Cosine similarity between $\nabla\ell(\boldsymbol{w}^{p_1})$ and $\nabla\ell(\boldsymbol{w}^{p_2})$ with different ρ during training. (b) Cosine similarity between $\nabla\ell(\boldsymbol{w}^{p_1})$ and the gradient of other perturbed weights in the grid spanned by $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$.

single-step ascent ($N_{tr} = 1$) achieves the lowest cross entropy loss for $\boldsymbol{w}$ (corresponds to $N_{te} = 0$). In contrast, SAM with multi-step ascent ($N_{tr} = 2, 3$) show a higher loss than $N_{tr} = 1$ for all the cases. However, if the number of ascent steps increases (i.e., $N_{te} = 1, 3$, and 5), SAM with multi-step ascent show a lower perturbed loss. Thus, as described in Fig. 5.2, we can conclude that using different number of ascent steps during training leads to different perturbed loss landscapes.

Gradient diversity of perturbed weights Inspired by the above results, we analyze the gradient of each perturbed weight. We estimate the cosine similarity between $\nabla\ell(\boldsymbol{w}^{p_1})$ and $\nabla\ell(\boldsymbol{w}^{p_2})$ during training. As shown in Fig. 5.4a, we discover that the cosine similarity decreases as training proceeds. Especially, a larger training ρ leads to a lower cosine similarity. The cosine similarity between $\nabla\ell(\boldsymbol{w}^{p_1})$ and $\nabla\ell(\boldsymbol{w}^{p_2})$ drops to 0.6 for $\rho = 0.5$. In Fig. 5.4b, we plot the cosine similarity of each perturbed weight in the perturbed space generated by $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ for $\rho = 0.5$. We

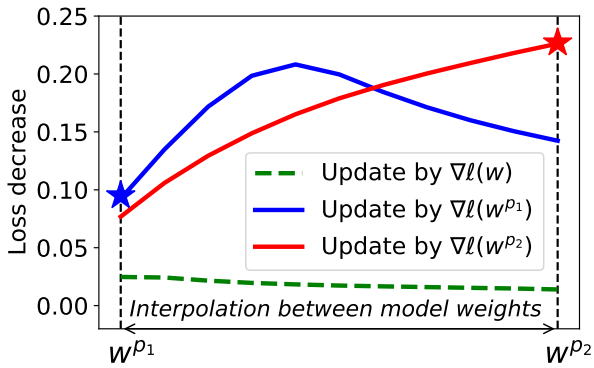


Figure 5.5: (CIFAR-10) Loss decrease for each update gradient. Loss decrease of perturbed weights $\lambda \boldsymbol{w}^{p_1} + (1-\lambda)\boldsymbol{w}^{p_2}$ by using each update gradient. For example, the blue star indicates $\ell(\boldsymbol{w}^{p_1}) - \ell(\boldsymbol{w}^{p_1} - \eta \nabla \ell(\boldsymbol{w}^{p_1}))$ where η is a learning rate. Each gradient shows different effects on perturbed losses.

discover that not only $\nabla \ell(\boldsymbol{w}^{p_1})$ and $\nabla \ell(\boldsymbol{w}^{p_2})$, but also other perturbed weights have different gradients.

To push further, we estimate the effect of each gradient on the perturbed loss landscape. In Fig. 5.5, we first interpolate the perturbed weights $\boldsymbol{w}^{p_1}$ and $\boldsymbol{w}^{p_2}$. Then, we measure the loss decreases by using different gradients to update weights at the end of training. First of all, $\nabla \ell(\boldsymbol{w})$ fails to sufficiently reduce the losses of all interpolated weights including both $\boldsymbol{w}^{p_1}$ and $\boldsymbol{w}^{p_2}$, which is consistent with the observation that the gradient of SGD generally cannot minimize the perturbed loss in (Zhuang et al., 2021). In contrast, the gradients of perturbed weights $\nabla \ell(\boldsymbol{w}^{p_1})$ and $\nabla \ell(\boldsymbol{w}^{p_2})$ successfully minimizes both $\ell(\boldsymbol{w}^{p_1})$ and $\ell(\boldsymbol{w}^{p_2})$. However, as marked with stars, the gradients of each perturbed point ($\nabla \ell(\boldsymbol{w}^{p_1})$ and $\nabla \ell(\boldsymbol{w}^{p_2})$) show better loss decreases in the vicinity of their own weights $\boldsymbol{w}^{p_1}$ and $\boldsymbol{w}^{p_2}$, respectively.

The above results are consistent with Table 5.1 in that $N_{tr} = 1$ sufficiently reduces the perturbed loss with $N_{te} \leq 1$, while it fails to achieve low loss values for

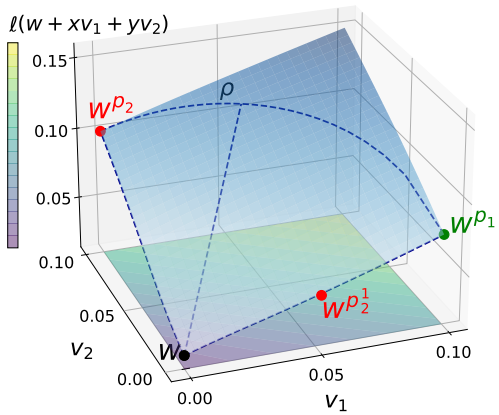


Figure 5.6: Loss surface of multiple ascent directions.

Algorithm 3: Multi-ascent Sharpness-Aware Minimization

Parameter: Parameter $\boldsymbol{w}_t$ at step t, learning rate η, number of ascent steps N, series of radii ρ_n for $n = \{1, \cdots, N\}$.

$\boldsymbol{w}_t^{p_0} \leftarrow \boldsymbol{w}_t$, $\boldsymbol{g}_0 \leftarrow \nabla \ell(\boldsymbol{w}_t)$

for $n = 1$ *to* N **do**

  $\boldsymbol{w}_t^{p_n} \leftarrow \boldsymbol{w}_t^{p_{n-1}} + \rho_n \cdot \frac{\boldsymbol{g}_{n-1}}{\|\boldsymbol{g}_{n-1}\|}$

  $\boldsymbol{g}_n \leftarrow \nabla \ell(\boldsymbol{w}_t^{p_n})$

end

$\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t - \eta \cdot \frac{1}{N} \sum_{n=1}^{N} \boldsymbol{g}_n$

$N_{te} \geq 2$. In contrast, $N_{tr} \geq 2$ successfully reduces the perturbed loss with $N_{te} \geq 2$. In this point of view, instead of minimizing one specific perturbed loss, diverse perturbed points should be considered to achieve the optimal flatness in terms of minimizing perturbed losses in ρ-neighborhood.

5.3.2 Multi-ascent Sharpness-Aware Minimization

Previously, we found that perturbed weights obtained by different number of ascent steps provide distinct gradient information that effectively minimizes their own

perturbed losses. From this observation, we suggest a simple method that considers multiple perturbed weights to improve the performance of SAM. Given the number of ascent steps N, the proposed method minimizes $\sum_{n=1}^{N} \ell(\boldsymbol{w}^{p_N^n})$, where $\boldsymbol{w}^{p_N^n}$ is a series of perturbed weights in Eq. (5.5). With a similar gradient approximation in (Foret et al., 2020), the update process can be formalized as follows:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \cdot \frac{1}{N} \sum_{n=1}^{N} \nabla \ell(\boldsymbol{w}_t^{p_N^n}) \tag{5.7}$$

We name this method Multi-ascent Sharpness-Aware Minimization (MSAM). Algorithm 3 shows the pseudo-code.

Comparison between MSAM and multi-step ascent SAM The most important difference between MSAM and multi-step ascent SAM is whether it uses the gradient of only one single perturbed weight or multiple perturbed weights. Specifically, MSAM minimizes $\sum_{n=1}^{N} \ell(\boldsymbol{w}^{p_N^n})$, while multi-step ascent SAM only minimizes $\ell(\boldsymbol{w}^{p_N})$. Simply, if $N = 2$, the proposed method minimizes both $\ell(\boldsymbol{w}^{p_2^1})$ and $\ell(\boldsymbol{w}^{p_2})$, while multi-step ascent SAM only minimizes $\ell(\boldsymbol{w}^{p_2})$ as illustrated in Fig. 5.6.

The ascent gradient of n-th inner iteration, $\nabla \ell(\boldsymbol{w}^{p_N^n})$, is used as an ascent direction to obtain $\boldsymbol{w}^{p_N^{n+1}}$ in Eq. (5.5). MSAM simply re-uses these gradients $\nabla \ell(\boldsymbol{w}^{p_N^n})$ as descent gradients. Thus, it requires the same forward/backward computation to multi-step ascent SAM. However, unfortunately, for both SAM and MSAM, $N \geq 2$ has a higher computational cost than single-step ascent SAM.

Upper bound on generalization performance We begin with a theoretical analysis that considers multiple ascent points and also provides an upper bound on generalization.

Theorem 5.1. *(PAC-Bayesian Theorem (McAllester, 1999; Dziugaite and Roy, 2017)) For given training dataset $\mathcal{S}$ drawn from population distribution $\mathcal{D}$, define the training loss $L_{\mathcal{S}}(\boldsymbol{w}) := \frac{1}{|\mathcal{S}|}\sum_{\boldsymbol{x}\in\mathcal{S}}\ell(\boldsymbol{w},\boldsymbol{x})$ and the population loss $L_{\mathcal{D}}(\boldsymbol{w}) := \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[\ell(\boldsymbol{w},\boldsymbol{x})]$. With $1-a$ probability, for any prior $\mathcal{P}$ and posterior $\mathcal{Q}$, the following inequality holds:*

$$\mathbb{E}_{\boldsymbol{w}\sim\mathcal{Q}}\left[L_{\mathcal{D}}(\boldsymbol{w})\right] \leq \mathbb{E}_{\boldsymbol{w}\sim\mathcal{Q}}\left[L_{\mathcal{S}}(\boldsymbol{w})\right] + \sqrt{\frac{KL(\mathcal{Q}\|\mathcal{P}) + \log\frac{|\mathcal{S}|}{a}}{2(|\mathcal{S}|-1)}} \tag{5.8}$$

where $KL(\cdot\|\cdot)$ is Kullback–Leibler divergence.

Corollary 5.2. *(Stated informally) Suppose $L_{\mathcal{D}}(\boldsymbol{w}) \leq \mathbb{E}_{\boldsymbol{\delta}\sim\mathcal{N}(0,\sigma^2 I)}[L_{\mathcal{D}}(\boldsymbol{w}+\boldsymbol{\delta})]$, where $\boldsymbol{\delta}\in\mathbb{R}^k$ and k is the number of parameters. Then, for any $0\leq\lambda\leq 1$, the following inequality holds with high probability*

$$L_{\mathcal{D}}(\boldsymbol{w}) \leq \lambda \max_{\|\boldsymbol{\delta}\|\leq\rho_1} L_{\mathcal{S}}(\boldsymbol{w}+\boldsymbol{\delta}) + (1-\lambda)\max_{\|\boldsymbol{\delta}\|\leq\rho_2} L_{\mathcal{S}}(\boldsymbol{w}+\boldsymbol{\delta}) \tag{5.9}$$

$$+\sqrt{\frac{KL(\mathcal{Q}\|\mathcal{P}) + \log\frac{|\mathcal{S}|}{a}}{2(|\mathcal{S}|-1)}} \tag{5.10}$$

Proof. Assume $\epsilon_i \sim \mathcal{N}(0,\sigma)$. Then, $\|\boldsymbol{\epsilon}\|_2^2$ follows a Chi-square distribution. By Lemma.1 in (Laurent and Massart, 2000), following inequality holds for every non-negative t:

$$\mathrm{P}(\|\boldsymbol{\epsilon}\|_2^2 - k\sigma^2 \geq 2\sigma^2\sqrt{kt} + 2t\sigma^2) \leq exp(-t) \tag{5.11}$$

Without loss of generality, let $\rho_1 \leq \rho_2$. Then, with probability at least $1-1/\sqrt{\alpha n}$:

$$\|\boldsymbol{\epsilon}\|_2^2 \leq \sigma^2(2\log\sqrt{\alpha n} + k + 2\sqrt{k\log\sqrt{\alpha n}}) \leq 2\sigma^2 k\left(1+\sqrt{\frac{\log\sqrt{\alpha n}}{k}}\right)^2 \leq \rho_1^2 \tag{5.12}$$

$$\leq 2\sigma^2 k\left(1+\sqrt{\frac{\log\sqrt{n}}{k}}\right)^2 \leq \rho_2^2 \tag{5.13}$$

Given a hyper-parameter λ, with probability at least $1-\frac{1}{\sqrt{\alpha}}exp\left(-\left(\frac{\rho_2}{\sqrt{2}\sigma}-\sqrt{k}\right)^2\right)$,

$$\mathbb{E}_{\epsilon_i\sim N(0,\sigma)}\left[L_{\mathcal{S}}(\boldsymbol{w}+\boldsymbol{\epsilon})\right] = \lambda\mathbb{E}_{\epsilon_i\sim N(0,\sigma)}\left[L_{\mathcal{S}}(\boldsymbol{w}+\boldsymbol{\epsilon})\right] + (1-\lambda)\mathbb{E}_{\epsilon_i\sim N(0,\sigma)}\left[L_{\mathcal{S}}(\boldsymbol{w}+\boldsymbol{\epsilon})\right] \tag{5.14}$$

$$\leq \lambda\max_{\|\boldsymbol{\epsilon}\|\leq\rho_1}\left[L_{\mathcal{S}}(\boldsymbol{w}+\boldsymbol{\epsilon})\right] + (1-\lambda)\max_{\|\boldsymbol{\epsilon}\|\leq\rho_2}\left[L_{\mathcal{S}}(\boldsymbol{w}+\boldsymbol{\epsilon})\right] \tag{5.15}$$

□

Corollary 5.2 implies that minimizing both losses of perturbed weights with different ρ_1 and ρ_2 also provides an upper bound on generalization.

5.4 Experiments

5.4.1 Experimental Setup

We use CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009), and ImageNet (Deng et al., 2009). For CIFAR-10 and CIFAR-100, we trained ResNet18 (He et al., 2016a) and Wide-ResNet-28-10 (WRN-28-10) (Zagoruyko and Komodakis, 2016b), respectively. We basically follow the settings in (Du et al., 2022; Foret et al., 2020), but we find that manipulating some training parameters can boost the performance of vanilla training and SAM. For CIFAR-10, we use a batch size of 128, weight decay 5e-4, and SGD with a learning rate of 0.1 with a momentum of 0.9. For CIFAR-100, we use a batch size of 256, weight decay 1e-3, and SGD with a learning rate of 0.1 with a momentum of 0.9. For both datasets, cutout (DeVries and Taylor, 2017) and cosine learning rate decay (Loshchilov and Hutter, 2016) are used with 200 epochs. Finally, in the radius ρ for SAM, we observe that CIFAR-10 and CIFAR-100 at $\rho = 0.1$ and 0.5 show the best performance, respectively. For ImageNet, we followed the settings used in (Du et al., 2022). We trained ResNet-50 with 224×224 resized images. The number of epochs is set to 90 with a cosine learning rate schedule. The maximum learning rate is fixed to 0.2 with the weight decay 1×10^{-4}. Inception-style data augmentation, and 512 batch size are used. For both SAM and MSAM, $\rho = 0.05$ is used. We train all the models using PyTorch-based libraries (Paszke et al., 2019; Kim, 2020; Yao et al., 2020) on NVIDIA TITAN V and RTX 3090.

Table 5.2: Experiment on CIFAR. The bold and underlined number denote the best and the second best results, respectively.

Method	Acc(A)	Acc(B)	Reported	Best
CIFAR-10 (ResNet18)				
SGD	96.26±0.03	-	-	96.26±0.03
SAM	96.54±0.07	96.38±0.14	96.52±0.13	96.54±0.07
ESAM	<u>96.56±0.05</u>	96.43±0.11	96.56±0.08	<u>96.56±0.05</u>
GSAM	96.05±0.05	96.19±0.03	-	96.19±0.03
ASAM	95.19±0.08	<u>96.53±0.09</u>	-	96.53±0.09
MSAM	**96.88±0.03**	**96.88±0.03**†	-	**96.88±0.03**
CIFAR-100 (WideResNet-28-10)				
SGD	82.94±0.13	-	-	82.94±0.13
SAM	<u>85.44±0.07</u>	84.05±0.17	85.10±0.20	<u>85.44±0.07</u>
ESAM	78.87±0.20	84.07±0.15	84.51±0.01	84.51±0.01
GSAM	83.36±0.39	82.76±0.20	-	83.36±0.39
ASAM	84.09±0.08	<u>84.36±0.15</u>	83.68±0.12	84.36±0.15
MSAM	**85.87±0.38**	**86.06±0.07**	-	**86.06±0.07**

†MSAM shows the best performance under the setting where SAM shows the best performance.

5.4.2 Generalization Performance

Here, we conduct experiments with additional methods in other studies (Kwon et al., 2021; Zhuang et al., 2021; Du et al., 2022).

For the CIFAR datasets, we find that ρ used in comparison methods (Foret et al., 2020; Kwon et al., 2021; Zhuang et al., 2021; Du et al., 2022) are highly varying than ImageNet. Thus, to provide a fair comparison, we report three accuracy values for each method on CIFAR: (1) Accuracy under the same ρ and training setting where SAM achieves the best performance, denoted as *Acc(A)*; (2) Accuracy under the radius ρ proposed by authors but other settings remain the same, denoted as *Acc(B)*; (3) Reported accuracy in their original paper, denoted as *Reported*. We report the average accuracy and standard deviation with three different random seeds.

Table 5.3: Experiments on SAM → MSAM on various switching epochs. Switching epochs of 200 indicates the model only trained with SAM.

Epochs	100	125	150	175	200
Acc (%)	96.71±0.08	96.71±0.01	96.65±0.11	96.63±0.04	96.54±0.07

The results are summarized in Table 5.2. Although SAM achieves higher accuracy than those reported in (Foret et al., 2020; Du et al., 2022) for both datasets under the optimized training settings, MSAM shows better performance than any other methods for both CIFAR-10 and CIFAR-100. Note that MSAM also outperforms SAM with $N \geq 2$, which shown in Table 5.1.

5.4.3 Escaping Local Minima

MSAM requires the same forward/backward computation to multi-step ascent SAM. However, when we compare it to single-step ascent SAM, it requires 1.5× computation. This is obviously the limitation of the proposed method. To detour the computational burden, we here to show the usefulness of MSAM by training pre-trained models. Specifically, we use MSAM for models obtained from different epochs during training with SAM. As shown in Table 5.3, MSAM improves the generalization performance of the model trained with SAM regardless of switching epochs. Thus, to reduce the training time, we recommend to use MSAM after training a model with SAM. To further improve the time computation, researches on combining efficient variants of SAM (such as ESAM) and using various gradient information remains as our future work.

5.5 Chapter Summary

In this chapter, we aimed to understand how the number of ascent steps affects on SAM. We also discovered the effect of using different gradients obtained during multi-step ascent. Based on these empirical observation, we proposed a new method utilizing all gradient information from multiple perturbed weights and achieve a better performance than comparison methods.

Chapter 6

Conclusion

6.1 Contributions

This dissertation provided a deeper understanding of adversarial robustness and generalization performance of deep learning models by analyzing their loss landscape. We empirically showed that catastrophic overfitting is closely related to decision boundary distortion, and proposed a new simple method to determine the appropriate magnitude of the perturbation for each image. We evaluated the robustness of the proposed method against various adversarial attacks and showed sufficient robustness using single-step adversarial training without the occurrence of any catastrophic overfitting. Additionally, we investigated and analyzed the adversarial robustness of deep learning models in various domains, and improved the performance of both adversarial attacks and adversarial defenses. We also analyzed the behavior of SAM, which revealed the presence of convergence instability near a saddle point, and demonstrated the importance of training tricks, such as momentum and batch size, in relation to the diffusion theory. Furthermore, we proposed a new method utilizing all gradient information from multiple perturbed weights and achieved a better performance than comparison methods.

6.2 Future Work

In this dissertation, we discovered that analyzing the loss landscape can be beneficial for improving adversarial robustness and generalization performance. However, although our observations and proposed methods enhance the stability and performance of deep learning models, robustness against possible diverse perturbations and generalization for possible transformations should be further studied and much progress is still needed to ensure security and provide more consistent accuracy in real-world applications. We leave additional research challenges in the pursuit of ideal robustness and generalization performance over various tasks and datasets. Furthermore, given the close relationship between these two problems in terms of the loss landscape, further analysis is needed across various tasks and datasets. Specifically, since both objectives are deeply connected to the shape of the loss landscape, there may be a method that simultaneously improves adversarial robustness and generalization performance. We hope our work will motivate researchers to develop optimal deep learning models and training algorithms.

Bibliography

Alayrac, J.B., Uesato, J., Huang, P.S., Fawzi, A., Stanforth, R., Kohli, P., 2019. Are labels required for improving adversarial robustness?, in: Advances in Neural Information Processing Systems, pp. 12214–12223.

Alzantot, M., Balaji, B., Srivastava, M., 2018. Did you hear that? adversarial examples against automatic speech recognition. arXiv preprint arXiv:1801.00554 .

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., et al., 2016. Deep speech 2: End-to-end speech recognition in english and mandarin, in: International conference on machine learning, pp. 173–182.

Anand, D., Patil, R., Agrawal, U., Rahul, V., Ravishankar, H., Sudhakar, P., 2022. Towards generalization of medical imaging ai models: Sharpness-aware minimizers and beyond, in: 2022 IEEE 19th International Symposium on Biomedical Imaging (ISBI), IEEE. pp. 1–5.

Andriushchenko, M., Croce, F., Flammarion, N., Hein, M., 2020. Square attack: a query-efficient black-box adversarial attack via random search, in: European Conference on Computer Vision, Springer. pp. 484–501.

Andriushchenko, M., Flammarion, N., 2020. Understanding and improving fast adversarial training. Advances in Neural Information Processing Systems 33.

Andriushchenko, M., Flammarion, N., 2022. Towards understanding sharpness-aware minimization, in: International Conference on Machine Learning, PMLR. pp. 639–668.

Anil, C., Lucas, J., Grosse, R., 2019. Sorting out lipschitz function approximation, in: International Conference on Machine Learning, PMLR. pp. 291–301.

Athalye, A., Carlini, N., Wagner, D., 2018a. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. arXiv preprint arXiv:1802.00420 , 274–283.

Athalye, A., Engstrom, L., Ilyas, A., Kwok, K., 2018b. Synthesizing robust adversarial examples, in: International conference on machine learning, PMLR. pp. 284–293.

Baevski, A., Schneider, S., Auli, M., 2019. vq-wav2vec: Self-supervised learning of discrete speech representations. arXiv preprint arXiv:1910.05453 .

Baevski, A., Zhou, Y., Mohamed, A., Auli, M., 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. Advances in Neural Information Processing Systems 33.

Balaji, Y., Goldstein, T., Hoffman, J., 2019. Instance adaptive adversarial training: Improved accuracy tradeoffs in neural nets. arXiv preprint arXiv:1910.08051 .

Barrett, D., Dherin, B., 2020. Implicit gradient regularization. International Conference on Learning Representations .

Bartlett, P.L., Jordan, M.I., McAuliffe, J.D., 2006. Convexity, classification, and risk bounds. Journal of the American Statistical Association 101, 138–156.

Belkin, M., Hsu, D., Ma, S., Mandal, S., 2019. Reconciling modern machine-learning practice and the classical bias–variance trade-off. Proceedings of the National Academy of Sciences 116, 15849–15854.

Bengio, Y., Louradour, J., Collobert, R., Weston, J., 2009. Curriculum learning, in: Proceedings of the 26th annual international conference on machine learning, pp. 41–48.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., Roli, F., 2013. Evasion attacks against machine learning at test time, in: Joint European conference on machine learning and knowledge discovery in databases, Springer. pp. 387–402.

Biggio, B., Roli, F., 2018. Wild patterns: Ten years after the rise of adversarial machine learning. Pattern Recognition 84, 317–331.

Boyd, S.P., Vandenberghe, L., 2004. Convex optimization. Cambridge university press.

Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al., 2020. Language models are few-shot learners. arXiv preprint arXiv:2005.14165 .

Cai, Q.Z., Liu, C., Song, D., 2018. Curriculum adversarial training, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, pp. 3740–3747.

Carlini, N., Wagner, D., 2017a. Adversarial examples are not easily detected: Bypassing ten detection methods, in: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, pp. 3–14.

Carlini, N., Wagner, D., 2017b. Towards evaluating the robustness of neural networks, in: 2017 ieee symposium on security and privacy (sp), IEEE. pp. 39–57.

Carlini, N., Wagner, D., 2018. Audio adversarial examples: Targeted attacks on speech-to-text, in: 2018 IEEE Security and Privacy Workshops (SPW), IEEE. pp. 1–7.

Carmon, Y., Raghunathan, A., Schmidt, L., Duchi, J.C., Liang, P.S., 2019. Unlabeled data improves adversarial robustness, in: Advances in Neural Information Processing Systems, pp. 11192–11203.

Chaudhari, P., Choromanska, A., Soatto, S., LeCun, Y., Baldassi, C., Borgs, C., Chayes, J., Sagun, L., Zecchina, R., 2019. Entropy-sgd: Biasing gradient descent into wide valleys. Journal of Statistical Mechanics: Theory and Experiment 2019, 124018.

Chen, J., Cheng, Y., Gan, Z., Gu, Q., Liu, J., 2020. Efficient robust training via backward smoothing. arXiv preprint arXiv:2010.01278 .

Chen, P.Y., Zhang, H., Sharma, Y., Yi, J., Hsieh, C.J., 2017. 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.

Chen, X., Hsieh, C.J., Gong, B., 2022. When vision transformers outperform resnets without pre-training or strong data augmentations. International Conference on Learning Representations URL: https://openreview.net/forum?id=LtKcMgGOeLt.

Cheng, M., Lei, Q., Chen, P.Y., Dhillon, I., Hsieh, C.J., 2020. Cat: Customized adversarial training for improved robustness. arXiv preprint arXiv:2002.06789 .

Cheng, S., Dong, Y., Pang, T., Su, H., Zhu, J., 2019. Improving black-box adversarial attacks with a transfer-based prior, in: Advances in Neural Information Processing Systems, pp. 10934–10944.

Choi, K., Fazekas, G., Sandler, M., Cho, K., 2017. Convolutional recurrent neural networks for music classification, in: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE. pp. 2392–2396.

Chung, J.S., Nagrani, A., Zisserman, A., 2018. Voxceleb2: Deep speaker recognition. arXiv preprint arXiv:1806.05622 .

Cisse, M., Adi, Y., Neverova, N., Keshet, J., 2017a. Houdini: Fooling deep structured prediction models. Advances in Neural Information Processing Systems 30.

Cisse, M., Bojanowski, P., Grave, E., Dauphin, Y., Usunier, N., 2017b. Parseval networks: Improving robustness to adversarial examples, in: International Conference on Machine Learning, PMLR. pp. 854–863.

Cohen, J., Kaur, S., Li, Y., Kolter, J.Z., Talwalkar, A., 2021. Gradient descent on neural networks typically occurs at the edge of stability. International Con-

ference on Learning Representations URL: https://openreview.net/forum?id=jh-rTtvkGeM.

Cohen, J.M., Rosenfeld, E., Kolter, J.Z., 2019. Certified adversarial robustness via randomized smoothing. arXiv preprint arXiv:1902.02918 , 1310–1320.

Croce, F., Andriushchenko, M., Sehwag, V., Debenedetti, E., Flammarion, N., Chiang, M., Mittal, P., Hein, M., 2020. Robustbench: a standardized adversarial robustness benchmark. arXiv preprint arXiv:2010.09670 .

Croce, F., Hein, M., 2019. Minimally distorted adversarial examples with a fast adaptive boundary attack. arXiv preprint arXiv:1907.02044 .

Croce, F., Hein, M., 2020. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. arXiv preprint arXiv:2003.01690 , 2206–2216.

Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V., 2019. Autoaugment: Learning augmentation strategies from data, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 113–123.

Demontis, A., Melis, M., Pintor, M., Jagielski, M., Biggio, B., Oprea, A., Nita-Rotaru, C., Roli, F., 2019. Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks, in: 28th {USENIX} Security Symposium ({USENIX} Security 19), pp. 321–338.

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L., 2009. Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. doi:10.1109/CVPR.2009.5206848.

Deng, L., Li, J., Huang, J.T., Yao, K., Yu, D., Seide, F., Seltzer, M., Zweig, G., He, X., Williams, J., et al., 2013. Recent advances in deep learning for speech research at microsoft, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE. pp. 8604–8608.

Deng, Y., Zheng, X., Zhang, T., Chen, C., Lou, G., Kim, M., 2020. An analysis of adversarial attacks and defenses on autonomous driving models, in: 2020 IEEE International Conference on Pervasive Computing and Communications (PerCom), IEEE. pp. 1–10.

Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 .

DeVries, T., Taylor, G.W., 2017. Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 .

Dewi, C., Chen, R.C., Liu, Y.T., et al., 2019. Similar music instrument detection via deep convolution yolo-generative adversarial network, in: 2019 IEEE 10th international conference on awareness science and technology (iCAST), IEEE. pp. 1–6.

Dhillon, G.S., Azizzadenesheli, K., Lipton, Z.C., Bernstein, J., Kossaifi, J., Khanna, A., Anandkumar, A., 2018. Stochastic activation pruning for robust adversarial defense. arXiv preprint arXiv:1803.01442 .

Dietterich, T.G., 2000. Ensemble methods in machine learning, in: International workshop on multiple classifier systems, Springer. pp. 1–15.

Ding, G.W., Sharma, Y., Lui, K.Y.C., Huang, R., 2019. Mma training: Direct input space margin maximization through adversarial training. International Conference on Learning Representations .

Dinh, L., Pascanu, R., Bengio, S., Bengio, Y., 2017. Sharp minima can generalize for deep nets, in: International Conference on Machine Learning, PMLR. pp. 1019–1028.

Dong, M., 2018. Convolutional neural network achieves human-level accuracy in music genre classification. arXiv preprint arXiv:1802.09697 .

Dong, Y., Pang, T., Su, H., Zhu, J., 2019. Evading defenses to transferable adversarial examples by translation-invariant attacks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4312–4321.

Dong, Y., Xu, K., Yang, X., Pang, T., Deng, Z., Su, H., Zhu, J., 2021. Exploring memorization in adversarial training. arXiv preprint arXiv:2106.01606 .

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al., 2020. An image is worth 16x16 words: Transformers for image recognition at scale. International Conference on Learning Representations .

Du, J., Yan, H., Feng, J., Zhou, J.T., Zhen, L., Goh, R.S.M., Tan, V., 2022. Efficient sharpness-aware minimization for improved training of neural networks. International Conference on Learning Representations URL: https://openreview.net/forum?id=n0OeTdNRG0Q.

Du, S.S., Jin, C., Lee, J.D., Jordan, M.I., Singh, A., Poczos, B., 2017. Gradient descent can take exponential time to escape saddle points. Advances in neural information processing systems 30.

Du, X., Pun, C.M., Zhang, Z., 2020. A unified framework for detecting audio adversarial examples, in: Proceedings of the 28th ACM International Conference on Multimedia, pp. 3986–3994.

Duchi, J., Hazan, E., Singer, Y., 2011. Adaptive subgradient methods for online learning and stochastic optimization. Journal of machine learning research 12.

Dziugaite, G.K., Roy, D.M., 2017. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. arXiv preprint arXiv:1703.11008 .

Eddy, S.R., 1996. Hidden markov models. Current opinion in structural biology 6, 361–365.

Elsayed, G., Krishnan, D., Mobahi, H., Regan, K., Bengio, S., 2018. Large margin deep networks for classification. Advances in Neural Information Processing Systems 31, 842–852.

Engstrom, L., Ilyas, A., Athalye, A., 2018. Evaluating and understanding the robustness of adversarial logit pairing. arXiv preprint arXiv:1807.10272 .

Esmaeilpour, M., Cardinal, P., Koerich, A.L., 2019. A robust approach for securing audio classification against adversarial attacks. IEEE Transactions on Information Forensics and Security 15, 2147–2159.

Fazlyab, M., Robey, A., Hassani, H., Morari, M., Pappas, G., 2019. Efficient and accurate estimation of lipschitz constants for deep neural networks. Advances in Neural Information Processing Systems 32, 11427–11438.

Feinman, R., Curtin, R.R., Shintre, S., Gardner, A.B., 2017. Detecting adversarial samples from artifacts. arXiv preprint arXiv:1703.00410 .

Foggia, P., Petkov, N., Saggese, A., Strisciuglio, N., Vento, M., 2015. Reliable detection of audio events in highly noisy environments. Pattern Recognition Letters 65, 22–28.

Foret, P., Kleiner, A., Mobahi, H., Neyshabur, B., 2020. Sharpness-aware minimization for efficiently improving generalization. International Conference on Learning Representations .

Garipov, T., Izmailov, P., Podoprikhin, D., Vetrov, D.P., Wilson, A.G., 2018. Loss surfaces, mode connectivity, and fast ensembling of dnns. Advances in neural information processing systems 31.

Garofolo, J., Graff, D., Paul, D., Pallett, D., 1993. Csr-i (wsj0) complete ldc93s6a. Web Download. Philadelphia: Linguistic Data Consortium 83.

Geiger, A., Lenz, P., Urtasun, R., 2012. Are we ready for autonomous driving? the kitti vision benchmark suite, in: 2012 IEEE conference on computer vision and pattern recognition, IEEE. pp. 3354–3361.

Goldberg, Y., Levy, O., 2014. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. arXiv preprint arXiv:1402.3722 .

Gong, Y., Poellabauer, C., 2017. Crafting adversarial examples for speech paralinguistics applications. arXiv preprint arXiv:1711.03280 .

Goodfellow, I.J., Shlens, J., Szegedy, C., 2014. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 .

Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T., Kohli, P., 2018. On the effectiveness of interval bound propagation for training verifiably robust models. arXiv preprint arXiv:1810.12715 .

Gowal, S., Qin, C., Uesato, J., Mann, T., Kohli, P., 2020. Uncovering the limits of adversarial training against norm-bounded adversarial examples. arXiv preprint arXiv:2010.03593 .

Graves, A., Fernández, S., Gomez, F., Schmidhuber, J., 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks, in: Proceedings of the 23rd international conference on Machine learning, pp. 369–376.

Graves, A., Mohamed, A.r., Hinton, G., 2013. Speech recognition with deep recurrent neural networks, in: 2013 IEEE international conference on acoustics, speech and signal processing, Ieee. pp. 6645–6649.

Grosse, K., Manoharan, P., Papernot, N., Backes, M., McDaniel, P., 2017. On the (statistical) detection of adversarial examples. arXiv preprint arXiv:1702.06280 .

Guckenheimer, J., Holmes, P., 2013. Nonlinear oscillations, dynamical systems, and bifurcations of vector fields. volume 42. Springer Science & Business Media.

Guo, C., Rana, M., Cisse, M., Van Der Maaten, L., 2017. Countering adversarial images using input transformations. arXiv preprint arXiv:1711.00117 .

Guo, Y., Li, Q., Chen, H., 2020. Backpropagating linearly improves transferability of adversarial examples. arXiv preprint arXiv:2012.03528 .

Han, K., Xia, B., Li, Y., 2022. 2: Adversarial domain adaptation to defense with adversarial perturbation removal. Pattern Recognition 122, 108303.

Hang, J., Han, K., Chen, H., Li, Y., 2020. Ensemble adversarial black-box attacks against deep learning systems. Pattern Recognition 101, 107184.

Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A., et al., 2014. Deep speech: Scaling up end-to-end speech recognition. arXiv preprint arXiv:1412.5567 .

He, H., Huang, G., Yuan, Y., 2019a. Asymmetric valleys: Beyond sharp and flat local minima. Advances in neural information processing systems 32.

He, K., Zhang, X., Ren, S., Sun, J., 2016a. Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778.

He, K., Zhang, X., Ren, S., Sun, J., 2016b. Identity mappings in deep residual networks, in: European conference on computer vision, Springer. pp. 630–645.

He, Z., Rakin, A.S., Fan, D., 2019b. Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 588–597.

Hein, M., Andriushchenko, M., 2017. Formal guarantees on the robustness of a classifier against adversarial manipulation, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 2263–2273.

Hendrycks, D., Dietterich, T., 2018. Benchmarking neural network robustness to common corruptions and perturbations. International Conference on Learning Representations .

Hendrycks, D., Lee, K., Mazeika, M., 2019. Using pre-training can improve model robustness and uncertainty, in: International Conference on Machine Learning, PMLR. pp. 2712–2721.

Hochreiter, S., Schmidhuber, J., 1994. Simplifying neural nets by discovering flat minima. Advances in neural information processing systems 7.

Hoffer, E., Hubara, I., Soudry, D., 2017. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. Advances in neural information processing systems 30.

Hu, S., Shang, X., Qin, Z., Li, M., Wang, Q., Wang, C., 2019. Adversarial examples for automatic speech recognition: attacks and countermeasures. IEEE Communications Magazine 57, 120–126.

Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q., 2017. Densely connected convolutional networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700–4708.

Huang, L., Liu, X., Lang, B., Yu, A.W., Wang, Y., Li, B., 2018. Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds

in deep neural networks. Thirty-Second AAAI Conference on Artificial Intelligence .

Ilyas, A., Engstrom, L., Athalye, A., Lin, J., 2018. Black-box adversarial attacks with limited queries and information. arXiv preprint arXiv:1804.08598 .

Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: International conference on machine learning, PMLR. pp. 448–456.

Ishida, T., Yamane, I., Sakai, T., Niu, G., Sugiyama, M., 2020. Do we need zero training loss after achieving zero training error?, in: International Conference on Machine Learning, PMLR. pp. 4604–4614.

Iter, D., Huang, J., Jermann, M., 2017. Generating adversarial examples for speech recognition. Stanford Technical Report .

Izmailov, P., Wilson, A., Podoprikhin, D., Vetrov, D., Garipov, T., 2018. Averaging weights leads to wider optima and better generalization, in: 34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018, pp. 876–885.

Jastrzkebski, S., Kenton, Z., Arpit, D., Ballas, N., Fischer, A., Bengio, Y., Storkey, A., 2017. Three factors influencing minima in sgd. arXiv preprint arXiv:1711.04623 .

Jati, A., Hsu, C.C., Pal, M., Peri, R., AbdAlmageed, W., Narayanan, S., 2021. Adversarial attack and defense strategies for deep speaker recognition systems. Computer Speech & Language 68, 101199.

Jayashankar, T., Roux, J.L., Moulin, P., 2020. Detecting audio attacks on asr systems with dropout uncertainty. arXiv preprint arXiv:2006.01906 .

Jelassi, S., Li, Y., 2022. Towards understanding how momentum improves generalization in deep learning, in: International Conference on Machine Learning, PMLR. pp. 9965–10040.

Jiang, Y., Neyshabur, B., Mobahi, H., Krishnan, D., Bengio, S., 2019. Fantastic generalization measures and where to find them. International Conference on Learning Representations .

Kannan, H., Kurakin, A., Goodfellow, I., 2018. Adversarial logit pairing. arXiv preprint arXiv:1803.06373 .

Kariyappa, S., Qureshi, M.K., 2019. Improving adversarial robustness of ensembles with diversity training. arXiv preprint arXiv:1901.09981 .

Katz, G., Barrett, C., Dill, D.L., Julian, K., Kochenderfer, M.J., 2017. Reluplex: An efficient smt solver for verifying deep neural networks, in: International Conference on Computer Aided Verification, Springer. pp. 97–117.

Kereliuk, C., Sturm, B.L., Larsen, J., 2015. Deep learning and music adversaries. IEEE Transactions on Multimedia 17, 2059–2071.

Keskar, N.S., Nocedal, J., Tang, P.T.P., Mudigere, D., Smelyanskiy, M., 2017. On large-batch training for deep learning: Generalization gap and sharp minima. 5th International Conference on Learning Representations, ICLR 2017 .

Kim, H., 2020. Torchattacks: A pytorch repository for adversarial attacks. arXiv preprint arXiv:2010.01950 .

Kim, H., Lee, W., Lee, J., 2021a. Understanding catastrophic overfitting in single-step adversarial training, in: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 8119–8127.

Kim, H., Lee, W., Lee, S., Lee, J., 2021b. Bridged adversarial training. arXiv preprint arXiv:2108.11135 .

Kim, H., Park, J., Choi, Y., Lee, J., 2023. Stability analysis of sharpness-aware minimization. arXiv preprint arXiv:2301.06308 .

Kim, H., Park, J., Lee, J., 2022a. Comment on transferability and input transformation with additive noise. arXiv preprint arXiv:2206.09075 .

Kim, H., Park, J., Lee, J., 2022b. Generating transferable adversarial examples for speech classification. Pattern Recognition , 109286URL: https://www.sciencedirect.com/science/article/pii/S0031320322007658, doi:https://doi.org/10.1016/j.patcog.2022.109286.

Kim, J.W., Yoon, H., Jung, H.Y., 2021c. Linguistic-coupled age-to-age voice translation to improve speech recognition performance in real environments. IEEE Access 9, 136476–136486. doi:10.1109/ACCESS.2021.3115608.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 .

Kleinberg, B., Li, Y., Yuan, Y., 2018a. An alternative view: When does sgd escape local minima?, in: International Conference on Machine Learning, PMLR. pp. 2698–2707.

Kleinberg, B., Li, Y., Yuan, Y., 2018b. An alternative view: When does SGD escape local minima?, in: Dy, J., Krause, A. (Eds.), Proceedings of the 35th International Conference on Machine Learning, PMLR. PMLR. pp. 2698–2707.

Kreuk, F., Adi, Y., Cisse, M., Keshet, J., 2018. Fooling end-to-end speaker verification with adversarial examples, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE. pp. 1962–1966.

Krizhevsky, A., Hinton, G., et al., 2009. Learning multiple layers of features from tiny images.(2009). Citeseer .

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems 25, 1097–1105.

Krogh, A., Hertz, J., 1991. A simple weight decay can improve generalization. Advances in neural information processing systems 4.

Kurakin, A., Goodfellow, I., Bengio, S., 2017. Adversarial machine learning at scale. International Conference on Learning Representations .

Kurakin, A., Goodfellow, I., Bengio, S., et al., 2016. Adversarial examples in the physical world.

Kwon, H., Kim, Y., Yoon, H., Choi, D., 2019a. Selective audio adversarial example in evasion attack on speech recognition system. IEEE Transactions on Information Forensics and Security 15, 526–538.

Kwon, H., Yoon, H., Park, K.W., 2019b. Poster: Detecting audio adversarial example

through audio modification, in: Proceedings of the 2019 ACM SIGSAC conference on computer and communications security, pp. 2521–2523.

Kwon, J., Kim, J., Park, H., Choi, I.K., 2021. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks, in: International Conference on Machine Learning, PMLR. pp. 5905–5914.

Lamb, A., Verma, V., Kannala, J., Bengio, Y., 2019. Interpolated adversarial training: Achieving robust neural networks without sacrificing too much accuracy, in: Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security, pp. 95–103.

Laurent, B., Massart, P., 2000. Adaptive estimation of a quadratic functional by model selection. Annals of Statistics , 1302–1338.

Le, Y., Yang, X., 2015. Tiny imagenet visual recognition challenge. CS 231N 7, 7.

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. nature 521, 436–444.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE 86, 2278–2324.

LeCun, Y., Cortes, C., Burges, C., 2010. Mnist handwritten digit database. ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist 2.

Lecuyer, M., Atlidakis, V., Geambasu, R., Hsu, D., Jana, S., 2019. Certified robustness to adversarial examples with differential privacy, in: 2019 IEEE Symposium on Security and Privacy (SP), IEEE. pp. 656–672.

Lee, H., Pham, P., Largman, Y., Ng, A., 2009. Unsupervised feature learning for audio classification using convolutional deep belief networks. Advances in neural information processing systems 22, 1096–1104.

Lee, J., Kim, T., Park, J., Nam, J., 2017. Raw waveform-based audio classification using sample-level cnn architectures. arXiv preprint arXiv:1712.00866 .

Lee, J.D., Simchowitz, M., Jordan, M.I., Recht, B., 2016. Gradient descent only converges to minimizers, in: Conference on learning theory, PMLR. pp. 1246–1257.

Lee, S., Kim, H., Lee, J., 2022a. Graddiv: Adversarial robustness of randomized neural networks via gradient diversity regularization. IEEE Transactions on Pattern Analysis and Machine Intelligence , 1–1doi:10.1109/TPAMI.2022.3169217.

Lee, S., Lee, J., Park, S., 2020. Lipschitz-certifiable training with a tight outer bound. Advances in Neural Information Processing Systems 33, 16891–16902. URL: https://proceedings.neurips.cc/paper/2020/file/c46482dd5d39742f0bfd417b492d0e8e-Paper.pdf.

Lee, S., Lee, W., Park, J., Lee, J., 2021a. Towards better understanding of training certifiably robust models against adversarial examples. Advances in Neural Information Processing Systems 34.

Lee, W., Kim, H., Lee, J., 2021b. Compact class-conditional domain invariant learning for multi-class domain adaptation. Pattern Recognition 112, 107763.

Lee, W., Lee, S., Byun, J., Kim, H., Lee, J., 2022b. Variational cycle-consistent

imputation adversarial networks for general missing patterns. Pattern Recognition 129, 108720.

Levinson, J., Askeland, J., Becker, J., Dolson, J., Held, D., Kammel, S., Kolter, J.Z., Langer, D., Pink, O., Pratt, V., et al., 2011. Towards fully autonomous driving: Systems and algorithms, in: 2011 IEEE intelligent vehicles symposium (IV), IEEE. pp. 163–168.

Lewkowycz, A., Bahri, Y., Dyer, E., Sohl-Dickstein, J., Gur-Ari, G., 2020. The large learning rate phase of deep learning: the catapult mechanism. arXiv preprint arXiv:2003.02218 .

Li, B., Chen, C., Wang, W., Carin, L., 2019. Certified adversarial robustness with additive noise. Advances in Neural Information Processing Systems 32, 9464–9474.

Li, B., Wang, S., Jana, S., Carin, L., 2020a. Towards understanding fast adversarial training. arXiv preprint arXiv:2006.03089 .

Li, D., Zhang, J., Huang, K., 2021. Universal adversarial perturbations against object detection. Pattern Recognition 110, 107584.

Li, H., Xu, Z., Taylor, G., Studer, C., Goldstein, T., 2018. Visualizing the loss landscape of neural nets. Advances in neural information processing systems 31.

Li, Z., Wu, Y., Liu, J., Chen, Y., Yuan, B., 2020b. Advpulse: Universal, synchronization-free, and targeted audio adversarial attacks via subsecond perturbations, in: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, pp. 1121–1134.

Lin, Y.Y., Zheng, W.Z., Chu, W.C., Han, J.Y., Hung, Y.H., Ho, G.M., Chang, C.Y., Lai, Y.H., 2021. A speech command control-based recognition system for dysarthric patients based on deep learning technology. Applied Sciences 11, 2477.

Liu, C., Salzmann, M., Lin, T., Tomioka, R., Süsstrunk, S., 2020. On the loss landscape of adversarial training: Identifying challenges and how to overcome them. arXiv preprint arXiv:2006.08403 33.

Liu, F., Han, B., Liu, T., Gong, C., Niu, G., Zhou, M., Sugiyama, M., et al., 2021a. Probabilistic margins for instance reweighting in adversarial training. Advances in Neural Information Processing Systems 34.

Liu, G., Khalil, I., Khreishah, A., 2021b. Using single-step adversarial training to defend iterative adversarial examples, in: Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy, pp. 17–27.

Liu, H., Ditzler, G., 2020. Detecting adversarial audio via activation quantization error, in: 2020 International Joint Conference on Neural Networks (IJCNN), IEEE. pp. 1–7.

Liu, J., Zhang, W., Zhang, Y., Hou, D., Liu, Y., Zha, H., Yu, N., 2019a. Detection based defense against adversarial examples from the steganalysis point of view, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4825–4834.

Liu, K., Ziyin, L., Ueda, M., 2021c. Noise and fluctuation of finite learning rate stochastic gradient descent, in: International Conference on Machine Learning, PMLR. pp. 7045–7056.

Liu, X., Cheng, M., Zhang, H., Hsieh, C.J., 2018a. Towards robust neural networks via random self-ensemble, in: Proceedings of the European Conference on Computer Vision (ECCV), pp. 369–385.

Liu, X., Hsieh, C.J., 2019. Rob-gan: Generator, discriminator, and adversarial attacker, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11234–11243.

Liu, X., Li, Y., Wu, C., Hsieh, C.J., 2018b. Adv-bnn: Improved adversarial defense through robust bayesian neural network. arXiv preprint arXiv:1810.01279 .

Liu, X., Zhang, X., Wan, K., Zhu, Q., Ding, Y., 2019b. Towards weighted-sampling audio adversarial example attack. arXiv preprint arXiv:1901.10300 .

Liu, Y., Chen, X., Liu, C., Song, D., 2017. Delving into transferable adversarial examples and black-box attacks. International Conference on Learning Representations .

Liu, Y., Mai, S., Cheng, M., Chen, X., Hsieh, C.J., You, Y., 2022. Random sharpness-aware minimization. Advances in Neural Information Processing Systems URL: https://openreview.net/forum?id=htUvh7xPoa.

Long, P., Ben-David, S., et al., 1999. Proceedings of the Twelfth Annual Conference on Computational Learning Theory. Assn for Computing Machinery.

Loshchilov, I., Hutter, F., 2016. Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983 .

Loshchilov, I., Hutter, F., 2018. Decoupled weight decay regularization. International Conference on Learning Representations .

von Luxburg, U., Bousquet, O., 2004. Distance-based classification with lipschitz functions. J. Mach. Learn. Res. 5, 669–695.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A., 2018. Towards deep learning models resistant to adversarial attacks. International Conference on Learning Representations .

Mandt, S., Hoffman, M.D., Blei, D.M., 2017. Stochastic gradient descent as approximate bayesian inference. arXiv preprint arXiv:1704.04289 .

Masters, D., Luschi, C., 2018. Revisiting small batch training for deep neural networks. arXiv preprint arXiv:1804.07612 .

McAllester, D.A., 1999. Pac-bayesian model averaging, in: Proceedings of the twelfth annual conference on Computational learning theory, pp. 164–170.

Mendes, E., Hogan, K., 2020. Defending against imperceptible audio adversarial examples using proportional additive gaussian noise. Massachusetts Institute of Technology Department of Mathematics .

Metzen, J.H., Genewein, T., Fischer, V., Bischoff, B., 2017. On detecting adversarial perturbations. arXiv preprint arXiv:1702.04267 .

Modas, A., Moosavi-Dezfooli, S.M., Frossard, P., 2019. Sparsefool: a few pixels make a big difference, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 9087–9096.

Moosavi-Dezfooli, S.M., Fawzi, A., Frossard, P., 2016. 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.

Nagarajan, V., Kolter, J.Z., 2019. Generalization in deep networks: The role of distance from initialization. arXiv preprint arXiv:1901.01672 .

Najafi, A., Maeda, S.i., Koyama, M., Miyato, T., 2019. Robustness to adversarial perturbations in learning from incomplete data, in: Advances in Neural Information Processing Systems, pp. 5541–5551.

Neekhara, P., Hussain, S., Pandey, P., Dubnov, S., McAuley, J., Koushanfar, F., 2019. Universal adversarial perturbations for speech recognition systems. arXiv preprint arXiv:1905.03828 .

Neyshabur, B., Bhojanapalli, S., McAllester, D., Srebro, N., 2017. Exploring generalization in deep learning, in: Advances in neural information processing systems, pp. 5947–5956.

Oord, A.v.d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., Kavukcuoglu, K., 2016. Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499 .

Palis, J.J., De Melo, W., 2012. Geometric theory of dynamical systems: an introduction. Springer Science & Business Media.

Panayotov, V., Chen, G., Povey, D., Khudanpur, S., 2015. Librispeech: an asr corpus based on public domain audio books, in: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE. pp. 5206–5210.

Pang, T., Xu, K., Du, C., Chen, N., Zhu, J., 2019a. Improving adversarial robustness via promoting ensemble diversity, in: International Conference on Machine Learning, PMLR. pp. 4970–4979.

Pang, T., Xu, K., Zhu, J., 2019b. Mixup inference: Better exploiting mixup to defend adversarial attacks. arXiv preprint arXiv:1909.11515 .

Pang, T., Yang, X., Dong, Y., Su, H., Zhu, J., 2020. Bag of tricks for adversarial training. arXiv preprint arXiv:2010.00467 .

Papernot, N., McDaniel, P., Goodfellow, I., 2016a. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv preprint arXiv:1605.07277 .

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B., Swami, A., 2017. Practical black-box attacks against machine learning, in: Proceedings of the 2017 ACM on Asia conference on computer and communications security, pp. 506–519.

Papernot, N., McDaniel, P., Sinha, A., Wellman, M., 2016b. Towards the science of security and privacy in machine learning. arXiv preprint arXiv:1611.03814 .

Park, N., Ji, S., Kim, J., 2021. Detecting audio adversarial examples with logit noising, in: Annual Computer Security Applications Conference, pp. 586–595.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al., 2019. Pytorch: An imperative style, high-performance deep learning library, in: Advances in neural information processing systems, pp. 8026–8037.

Pawitan, Y., 2001. In all likelihood: statistical modelling and inference using likelihood. Oxford University Press.

Qian, X., Klabjan, D., 2020. The impact of the mini-batch size on the variance of gradients in stochastic gradient descent. arXiv preprint arXiv:2004.13146 .

Qin, C., Martens, J., Gowal, S., Krishnan, D., Dvijotham, K., Fawzi, A., De, S., Stanforth, R., Kohli, P., 2019a. Adversarial robustness through local linearization, in: Advances in Neural Information Processing Systems, pp. 13847–13856.

Qin, Y., Carlini, N., Cottrell, G., Goodfellow, I., Raffel, C., 2019b. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition, in: International conference on machine learning, PMLR. pp. 5231–5240.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., 2019. Language models are unsupervised multitask learners. OpenAI blog 1, 9.

Raff, E., Sylvester, J., Forsyth, S., McLean, M., 2019. Barrage of random transforms for adversarially robust defense, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6528–6537.

Raghunathan, A., Steinhardt, J., Liang, P., 2018. Certified defenses against adversarial examples. arXiv preprint arXiv:1801.09344 .

Raghunathan, A., Xie, S.M., Yang, F., Duchi, J., Liang, P., 2020. Understanding and mitigating the tradeoff between robustness and accuracy. Proceedings of Machine Learning Research .

Raghunathan, A., Xie, S.M., Yang, F., Duchi, J.C., Liang, P., 2019. Adversarial training can hurt generalization. arXiv preprint arXiv:1906.06032 .

Rajaratnam, K., Kalita, J., 2018. Noise flooding for detecting audio adversarial examples against automatic speech recognition, in: 2018 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), IEEE. pp. 197–201.

Ravanelli, M., Parcollet, T., Bengio, Y., 2019. The pytorch-kaldi speech recognition toolkit, in: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE. pp. 6465–6469.

Rebuffi, S.A., Gowal, S., Calian, D.A., Stimberg, F., Wiles, O., Mann, T., 2021. Fixing data augmentation to improve adversarial robustness. arXiv preprint arXiv:2103.01946 .

Reddi, S., Zaheer, M., Sra, S., Poczos, B., Bach, F., Salakhutdinov, R., Smola, A., 2018. A generic approach for escaping saddle points, in: International conference on artificial intelligence and statistics, PMLR. pp. 1233–1242.

Rice, L., Wong, E., Kolter, J.Z., 2020. Overfitting in adversarially robust deep learning. arXiv preprint arXiv:2002.11569 , 8093–8104.

Risken, H., 1996. Fokker-planck equation, in: The Fokker-Planck Equation. Springer, pp. 63–95.

Ross, A.S., Doshi-Velez, F., 2018. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. Thirty-second AAAI conference on artificial intelligence .

Sainath, T.N., Parada, C., 2017. Convolutional neural networks for small-footprint keyword spotting, in: INTERSPEECH, ISCA. pp. 1606–1610.

Salman, H., Li, J., Razenshteyn, I., Zhang, P., Zhang, H., Bubeck, S., Yang, G., 2019. Provably robust deep learning via adversarially trained smoothed classifiers, in: Advances in Neural Information Processing Systems, pp. 11292–11303.

Sanyal, A., Dokania, P.K., Kanade, V., Torr, P., 2020. How benign is benign overfitting? International Conference on Learning Representations .

Sato, I., Nakagawa, H., 2014. Approximation analysis of stochastic gradient langevin dynamics by using fokker-planck equation and ito process, in: International Conference on Machine Learning, PMLR. pp. 982–990.

Schmidt, L., Santurkar, S., Tsipras, D., Talwar, K., Madry, A., 2018. Adversarially robust generalization requires more data, in: Advances in Neural Information Processing Systems, pp. 5014–5026.

Schneider, S., Baevski, A., Collobert, R., Auli, M., 2019. wav2vec: Unsupervised pre-training for speech recognition, in: INTERSPEECH, ISCA. pp. 3465–3469.

Sehwag, V., Wang, S., Mittal, P., Jana, S., 2020. Hydra: Pruning adversarially robust neural networks. Advances in Neural Information Processing Systems 33, 19655–19666.

Shaeiri, A., Nobahari, R., Rohban, M.H., 2020. Towards deep learning models resistant to large perturbations. arXiv preprint arXiv:2003.13370 .

Shafahi, A., Najibi, M., Ghiasi, M.A., Xu, Z., Dickerson, J., Studer, C., Davis, L.S., Taylor, G., Goldstein, T., 2019. Adversarial training for free!, in: Advances in Neural Information Processing Systems, pp. 3358–3369.

Simon-Gabriel, C.J., Ollivier, Y., Bottou, L., Schölkopf, B., Lopez-Paz, D., 2019. First-order adversarial vulnerability of neural networks and input dimension, in: International Conference on Machine Learning, PMLR. pp. 5809–5817.

Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 .

Sitawarin, C., Chakraborty, S., Wagner, D., 2020. Improving adversarial robustness through progressive hardening. arXiv preprint arXiv:2003.09347 .

Smith, L.N., 2017. Cyclical learning rates for training neural networks, in: 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE. pp. 464–472.

Smith, L.N., Topin, N., 2019. Super-convergence: Very fast training of neural networks using large learning rates, in: Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications, International Society for Optics and Photonics. p. 1100612.

Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., Khudanpur, S., 2018. X-vectors: Robust dnn embeddings for speaker recognition, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE. pp. 5329–5333.

Sokolić, J., Giryes, R., Sapiro, G., Rodrigues, M.R., 2017. Robust large margin deep neural networks. IEEE Transactions on Signal Processing 65, 4265–4280.

Springer, J.M., Mitchell, M., Kenyon, G.T., 2021. Uncovering universal features: How adversarial training improves adversarial transferability. ICML 2021 Workshop on Adversarial Machine Learning .

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014.

Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research 15, 1929–1958.

Strand, O.N., 1974. Theory and methods related to the singular-function expansion and landweber's iteration for integral equations of the first kind. SIAM Journal on Numerical Analysis 11, 798–825.

Su, J., Vargas, D.V., Sakurai, K., 2019. One pixel attack for fooling deep neural networks. IEEE Transactions on Evolutionary Computation 23, 828–841.

Sutskever, I., Martens, J., Dahl, G., Hinton, G., 2013a. On the importance of initialization and momentum in deep learning, in: Dasgupta, S., McAllester, D. (Eds.), Proceedings of the 30th International Conference on Machine Learning, PMLR, Atlanta, Georgia, USA. pp. 1139–1147.

Sutskever, I., Martens, J., Dahl, G., Hinton, G., 2013b. On the importance of initialization and momentum in deep learning, in: International conference on machine learning, PMLR. pp. 1139–1147.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R., 2013. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 .

Takashima, Y., Takashima, R., Takiguchi, T., Ariki, Y., 2019. Knowledge transferability between the speech data of persons with dysarthria speaking different languages for dysarthric speech recognition. IEEE Access 7, 164320–164326. doi:10.1109/ACCESS.2019.2951856.

Taori, R., Kamsetty, A., Chu, B., Vemuri, N., 2019. Targeted adversarial examples for black box audio systems, in: 2019 IEEE Security and Privacy Workshops (SPW), IEEE. pp. 15–20.

Torralba, A., Fergus, R., Freeman, W.T., 2008. 80 million tiny images: A large data set for nonparametric object and scene recognition. IEEE transactions on pattern analysis and machine intelligence 30, 1958–1970.

Tramer, F., Carlini, N., Brendel, W., Madry, A., 2020. On adaptive attacks to adversarial example defenses. Advances in Neural Information Processing Systems 33.

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., McDaniel, P., 2017a. Ensemble adversarial training: Attacks and defenses. arXiv preprint arXiv:1705.07204 .

Tramèr, F., Papernot, N., Goodfellow, I., Boneh, D., McDaniel, P., 2017b. The space of transferable adversarial examples. arXiv preprint arXiv:1704.03453 .

Trigeorgis, G., Ringeval, F., Brueckner, R., Marchi, E., Nicolaou, M.A., Schuller, B., Zafeiriou, S., 2016. Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network, in: 2016 IEEE international conference on acoustics, speech and signal processing (ICASSP), IEEE. pp. 5200–5204.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., Madry, A., 2018. Robustness may be at odds with accuracy. arXiv preprint arXiv:1805.12152 .

Tsuzuku, Y., Sato, I., Sugiyama, M., 2018. Lipschitz-margin training: Scalable cer-

tification of perturbation invariance for deep neural networks, in: Advances in neural information processing systems, pp. 6541–6550.

Uesato, J., Alayrac, J.B., Huang, P.S., Stanforth, R., Fawzi, A., Kohli, P., 2019. Are labels required for improving adversarial robustness? arXiv preprint arXiv:1905.13725 .

Uesato, J., O'Donoghue, B., Oord, A.v.d., Kohli, P., 2018. Adversarial risk and the dangers of evaluating against weak attacks. arXiv preprint arXiv:1802.05666 .

Vadillo, J., Santana, R., 2019. Universal adversarial examples in speech command classification. arXiv preprint arXiv:1911.10182 .

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. Advances in neural information processing systems 30, 5998–6008.

Veaux, C., Yamagishi, J., MacDonald, K., 2017. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit. University of Edinburgh .

Vivek, B., Babu, R.V., 2020. Single-step adversarial training with dropout scheduling, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE. pp. 947–956.

Wan, T., Zhou, Y., Ma, Y., Liu, H., 2019. Noise robust sound event detection using deep learning and audio enhancement, in: 2019 IEEE international symposium on signal processing and information technology (ISSPIT), IEEE. pp. 1–5.

Wang, D., Dong, L., Wang, R., Yan, D., Wang, J., 2020a. Targeted speech adversarial

example generation with generative adversarial network. IEEE Access 8, 124503–124513. doi:10.1109/ACCESS.2020.3006130.

Wang, L., Cho, W., Yoon, K.J., 2020b. Deceiving image-to-image translation networks for autonomous driving with adversarial perturbations. IEEE Robotics and Automation Letters 5, 1421–1428.

Wang, Q., Zheng, B., Li, Q., Shen, C., Ba, Z., 2020c. Towards query-efficient adversarial attacks against automatic speech recognition systems. IEEE Transactions on Information Forensics and Security 16, 896–908.

Wang, Y., Ma, X., Bailey, J., Yi, J., Zhou, B., Gu, Q., 2019a. On the convergence and robustness of adversarial training., in: ICML, p. 2.

Wang, Y., Wang, J., Yin, Z., Gong, R., Wang, J., Liu, A., Liu, X., 2022. Generating transferable adversarial examples against vision transformers, in: Proceedings of the 30th ACM International Conference on Multimedia, pp. 5181–5190.

Wang, Y., Zou, D., Yi, J., Bailey, J., Ma, X., Gu, Q., 2019b. Improving adversarial robustness requires revisiting misclassified examples. International Conference on Learning Representations .

Warden, P., 2018. Speech commands: A dataset for limited-vocabulary speech recognition. arXiv preprint arXiv:1804.03209 .

Wong, E., Kolter, Z., 2018. Provable defenses against adversarial examples via the convex outer adversarial polytope, in: International Conference on Machine Learning, pp. 5286–5295.

Wong, E., Rice, L., Kolter, J.Z., 2020. Fast is better than free: Revisiting adversarial training. arXiv preprint arXiv:2001.03994 .

Woodland, P.C., Odell, J.J., Valtchev, V., Young, S.J., 1994. Large vocabulary continuous speech recognition using htk, in: Proceedings of ICASSP'94. IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE. pp. II–125.

Wu, D., Wang, Y., Xia, S.T., Bailey, J., Ma, X., 2020a. Skip connections matter: On the transferability of adversarial examples generated with resnets. arXiv preprint arXiv:2002.05990 .

Wu, D., Xia, S.T., Wang, Y., 2020b. Adversarial weight perturbation helps robust generalization. Advances in Neural Information Processing Systems 33, 2958–2969.

Wu, H., Zheng, B., Li, X., Wu, X., Lee, H.y., Meng, H., 2022. Characterizing the adversarial vulnerability of speech self-supervised learning, in: ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE. pp. 3164–3168.

Wu, J., Chen, B., Luo, W., Fang, Y., 2020c. Audio steganography based on iterative adversarial attacks against convolutional neural networks. IEEE Transactions on Information Forensics and Security 15, 2282–2294.

Wu, W., Su, Y., Chen, X., Zhao, S., King, I., Lyu, M.R., Tai, Y.W., 2020d. Boosting the transferability of adversarial samples via attention, in: Proceedings of the

IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1161–1170.

Xie, C., Wang, J., Zhang, Z., Ren, Z., Yuille, A., 2017. Mitigating adversarial effects through randomization. arXiv preprint arXiv:1711.01991 .

Xie, C., Zhang, Z., Zhou, Y., Bai, S., Wang, J., Ren, Z., Yuille, A.L., 2019. Improving transferability of adversarial examples with input diversity, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2730–2739.

Xie, Z., Sato, I., Sugiyama, M., 2020. A diffusion theory for deep learning dynamics: Stochastic gradient descent exponentially favors flat minima. International Conference on Learning Representations .

Xie, Z., Wang, X., Zhang, H., Sato, I., Sugiyama, M., 2022. Adaptive inertia: Disentangling the effects of adaptive learning rate and momentum, in: International Conference on Machine Learning, PMLR. pp. 24430–24459.

Xu, H., Caramanis, C., Mannor, S., 2009. Robustness and regularization of support vector machines. Journal of machine learning research 10.

Xu, L., Du, Z., Mao, R., Zhang, F., Liu, R., 2020. Gsam: A deep neural network model for extracting computational representations of chinese addresses fused with geospatial feature. Computers, Environment and Urban Systems 81, 101473.

Yakura, H., Sakuma, J., 2019. Robust audio adversarial example for a physical attack, in: IJCAI, pp. 5334–5341.

Yamagishi, J., Veaux, C., MacDonald, K., et al., 2019. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit (version 0.92). University of Edinburgh. The Centre for Speech Technology Research (CSTR) .

Yang, Y., Khanna, R., Yu, Y., Gholami, A., Keutzer, K., Gonzalez, J.E., Ramchandran, K., Mahoney, M.W., 2020a. Boundary thickness and robustness in learning models. arXiv preprint arXiv:2007.05086 .

Yang, Y.Y., Rashtchian, C., Zhang, H., Salakhutdinov, R., Chaudhuri, K., 2020b. A closer look at accuracy vs. robustness. Advances in Neural Information Processing Systems .

Yang, Z., Li, B., Chen, P.Y., Song, D., 2019. Characterizing audio adversarial examples using temporal dependency. International Conference on Learning Representations .

Yao, Z., Gholami, A., Keutzer, K., Mahoney, M.W., 2020. Pyhessian: Neural networks through the lens of the hessian, in: 2020 IEEE international conference on big data (Big data), IEEE. pp. 581–590.

Yin, D., Kannan, R., Bartlett, P., 2019. Rademacher complexity for adversarially robust generalization, in: International Conference on Machine Learning, PMLR. pp. 7085–7094.

You, Z., Ye, J., Li, K., Xu, Z., Wang, P., 2019. Adversarial noise layer: Regularize neural network by adding noise, in: 2019 IEEE International Conference on Image Processing (ICIP), IEEE. pp. 909–913.

Yuan, X., Chen, Y., Zhao, Y., Long, Y., Liu, X., Chen, K., Zhang, S., Huang, H., Wang, X., Gunter, C.A., 2018. Commandersong: A systematic approach for practical adversarial voice recognition, in: 27th {USENIX} Security Symposium ({USENIX} Security 18), pp. 49–64.

Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y., 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 6023–6032.

Zagoruyko, S., Komodakis, N., 2016a. Wide residual networks. arXiv preprint arXiv:1605.07146 .

Zagoruyko, S., Komodakis, N., 2016b. Wide residual networks, in: Richard C. Wilson, E.R.H., Smith, W.A.P. (Eds.), Proceedings of the British Machine Vision Conference (BMVC), BMVA Press. pp. 87.1–87.12. URL: https://dx.doi.org/10.5244/C.30.87, doi:10.5244/C.30.87.

Zantedeschi, V., Nicolae, M.I., Rawat, A., 2017. Efficient defenses against adversarial attacks, in: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, pp. 39–49.

Zavriev, S., Kostyuk, F., 1993. Heavy-ball method in nonconvex optimization problems. Computational Mathematics and Modeling 4, 336–341.

Zhai, R., Cai, T., He, D., Dan, C., He, K., Hopcroft, J., Wang, L., 2019. Adversarially robust generalization just requires more unlabeled data. arXiv preprint arXiv:1906.00555 .

Zhang, C., Bengio, S., Hardt, M., Recht, B., Vinyals, O., 2016. Understanding deep learning requires rethinking generalization. arXiv preprint arXiv:1611.03530 .

Zhang, C., Bengio, S., Hardt, M., Recht, B., Vinyals, O., 2021. Understanding deep learning (still) requires rethinking generalization. Communications of the ACM 64, 107–115.

Zhang, G., Yan, C., Ji, X., Zhang, T., Zhang, T., Xu, W., 2017a. Dolphinattack. Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security URL: http://dx.doi.org/10.1145/3133956.3134052, doi:10.1145/3133956.3134052.

Zhang, H., Chen, H., Xiao, C., Gowal, S., Stanforth, R., Li, B., Boning, D., Hsieh, C.J., 2019a. Towards stable and efficient training of verifiably robust neural networks. arXiv preprint arXiv:1906.06316 .

Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D., 2017b. mixup: Beyond empirical risk minimization. arXiv preprint arXiv:1710.09412 .

Zhang, H., Yu, Y., Jiao, J., Xing, E.P., Ghaoui, L.E., Jordan, M.I., 2019b. Theoretically principled trade-off between robustness and accuracy. arXiv preprint arXiv:1901.08573 , 7472–7482.

Zhang, J., Xu, X., Han, B., Niu, G., Cui, L., Sugiyama, M., Kankanhalli, M., 2020a. Attacks which do not kill training make adversarial learning stronger, in: International Conference on Machine Learning, PMLR. pp. 11278–11287.

Zhang, M., Lucas, J., Ba, J., Hinton, G.E., 2019c. Lookahead optimizer: k steps forward, 1 step back. Advances in Neural Information Processing Systems 32.

Zhang, X., Zhou, Y., Pei, S., Zhuge, J., Chen, J., 2020b. Adversarial examples detection for xss attacks based on generative adversarial networks. IEEE Access 8, 10989–10996. doi:10.1109/ACCESS.2020.2965184.

Zhao, H., Wang, C., Guo, R., Rong, X., Guo, J., Yang, Q., Yang, L., Zhao, Y., Li, Y., 2022. Autonomous live working robot navigation with real-time detection and motion planning system on distribution line. High Voltage .

Zheng, S., Song, Y., Leung, T., Goodfellow, I., 2016. Improving the robustness of deep neural networks via stability training, in: Proceedings of the ieee conference on computer vision and pattern recognition, pp. 4480–4488.

Zheng, Y., Zhang, R., Mao, Y., 2021. Regularizing neural networks via adversarial model perturbation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8156–8165.

Zhou, W., Hou, X., Chen, Y., Tang, M., Huang, X., Gan, X., Yang, Y., 2018. Transferable adversarial perturbations, in: Proceedings of the European Conference on Computer Vision (ECCV), pp. 452–467.

Zhu, Z., Wu, J., Yu, B., Wu, L., Ma, J., 2018. The anisotropic noise in stochastic gradient descent: Its behavior of escaping from sharp minima and regularization effects. arXiv preprint arXiv:1803.00195 .

Zhuang, J., Gong, B., Yuan, L., Cui, Y., Adam, H., Dvornek, N.C., s Duncan, J., Liu, T., et al., 2021. Surrogate gap minimization improves sharpness-aware training. International Conference on Learning Representations .

Zimmermann, R.S., 2019. Comment on" adv-bnn: Improved adversarial defense through robust bayesian neural network". arXiv preprint arXiv:1907.00895 .

Ziyin, L., Li, B., Simon, J.B., Ueda, M., 2022. SGD can converge to local maxima. International Conference on Learning Representations URL: https://openreview.net/forum?id=9XhPLAjjRB.

국문초록

딥러닝은 다양한 분야에서 뛰어난 성능향상을 보이며, 음성 인식, 자율주행 및 의료 산업 등 많은 분야에 활용되고 있다. 딥러닝 모델은 수많은 가중치를 기반으로, 주어진 학습 데이터에 대한 손실함수를 줄이도록 학습된다. 그러나, 최근 학습 데이터에 대한 맹목적인 손실함수의 최소화는 크게 두 가지의 논의점이 있음이 밝혀졌다.

첫 번째 논의점은 딥러닝 모델의 강건성이다. 강건성이란 딥러닝 모델의 적대적 공격에 대한 방어 능력을 말한다. 적대적 공격은 학습된 딥러닝 모델의 가중치와 기울기 정보 등을 활용하여 비정상적인 데이터를 만들어내는 방법으로, 딥러닝 모델의 성능을 현저하게 저하시킨다. 현재까지 밝혀진 바로는 아주 작은 크기의 섭동도 비정상 데이터를 생성하기에 충분하여, 사람에게는 정상 데이터로 인식되나 딥러닝 모델은 치명적으로 오작동하는 적대적 예제를 쉽게 만들 수 있다. 따라서 딥러닝 모델의 안전한 상용화를 위해 강건성은 필수적으로 연구되어야 할 요소이다.

두 번째 논의점은 딥러닝 모델의 일반화이다. 일반화란 딥러닝 모델의 학습 데이터에 대한 성능과 평가 데이터에 대한 성능의 차이를 의미한다. 차이가 작을수록 일반화 성능이 높으며, 이는 곧 딥러닝 모델의 높은 상용화 가능성을 내포한다. 그러나 학습 데이터에 대한 손실함수만을 줄이는 학습 방법은 학습 데이터에 대한 과적합 현상을 불러오며, 이는 곧 평가 데이터에 대한 성능 감소로 이어짐이 여러 선행 연구에 의해 밝혀진 바 있다. 딥러닝 모델의 성능 향상은 학습 데이터가 아닌 평가 데이터에 대해 판단되므로, 일반화 성능의 달성은 모든 딥러닝 모델의 궁극적인 목표라고 할 수 있다.

본 연구에서는 손실함수평면의 탐색을 통해 두 논의점에 대한 분석과 각 논의점에 대응하는 지표를 향상시킬 수 있는 학습 방법을 제안한다. 우선, 강건성의 이해와 향상을 위해 입력값에 대한 손실함수를 분석한다. 적대적 공격은 입력값에 대해 손실함수를

최대화하는 섭동을 생성하므로, 비정상적인 섭동이 더해진 입력값에 대해서 손실함수를 최소화할 수 있는 방어 방법에 대해 연구한다. 그 시작으로, 적대적 방어 기법의 하나인 단일 단계 적대적 학습에서 손실함수평면이 쉽게 뒤틀릴 수 있음을 밝혀낸다. 제안된 연구에서 뒤틀린 손실함수평면이 모델의 강건성을 심각하게 손상할 수 있음을 보이고, 이를 기반으로 매끄러운 손실함수를 갖는 것의 중요성을 증명한다. 손실함수평면의 특성을 기반으로 다양한 영역에서의 적대적 공격과 방어 기법에 대한 분석과 성능 향상을 연구한다. 첫 번째로, 구조나 가중치가 상이한 모델에서 적대적 예제를 생성하여 대상 모델로 공격하는 전이 공격의 세기가 손실함수평면과 깊이 관련이 있음을 증명한다. 이를 기반으로 강력한 적대적 소리 예제를 생성하고, 딥러닝 모델의 신뢰할 수 있는 강건성 수준을 제안한다. 이어 적대적 학습의 특징과 학습된 모델의 손실함수평면을 탐색한다. 입력값에 대한 손실함수평면을 부드럽게 만들기 위하여, 적대적 학습에 중앙점을 고려한 손실함수를 도입하여 모델의 강건성을 높인다.

다음으로, 일반화의 이해와 향상을 위해 가중치에 대한 손실함수를 분석한다. 최근 일련의 연구에서는 딥러닝 모델의 일반화 성능은 손실함수평면의 평평함과 긴밀하게 연결되어 있음이 증명된 바 있다. 이를 기반으로 제안된 첨예 기반 학습은 첨예한 최적점을 기피하고 평평한 최적점을 찾음으로써 높은 일반화 성능을 달성한다. 본 연구에서는 첨예 기반 학습 방법의 손실함수평면에 대한 분석을 진행한다. 우선 첨예 기반 학습이 손실함수평면에 안장점이 존재할 경우 수렴이 불안정함을 밝힌다. 불안정한 수렴 때문에 최적점이 아닌 안장점에 갇히는 경우가 발생하며, 이는 첨예 기반 학습의 성능을 저해함을 보인다. 불안정한 수렴을 개선하고 더 높은 일반화 성능을 달성하기 위해, 가중치 공간에서의 섭동을 구하는 단계에서 도출되는 모든 중앙점의 기울기 정보를 활용하는 방법을 제안한다.

본 연구는 손실함수평면에 대한 탐색과 고찰을 바탕으로 강건성과 일반화에 대한 더 깊은 이해를 제시하고, 이를 통해서 각 지표의 향상을 위한 새로운 적대적 공격 방법,

적대적 방어 방법, 첨예 기반 학습 방법을 제안하였다. 연구 결과는 향후 딥러닝 모델의 실현을 위한 추후 연구에 확장성 있는 모델이며, 강건성과 일반화에 있어 손실함수평면에 대한 심도 있는 분석이 선행되어야 한다는 함의점을 제공한다.

주요어: 딥러닝, 강건성, 일반화, 손실함수평면

학번: 2018-23641