Improving additional adversarial robustness for classification

Michael Guo
Washington University in St. Louis

Follow this and additional works at: https://openscholarship.wustl.edu/eng_etds

Part of the Artificial Intelligence and Robotics Commons, Data Science Commons, Engineering Commons, Information Security Commons, and the Theory and Algorithms Commons

Recommended Citation
Guo, Michael, "Improving additional adversarial robustness for classification" (2021). McKelvey School of Engineering Theses & Dissertations. 572.
https://openscholarship.wustl.edu/eng_etds/572

This Thesis is brought to you for free and open access by the McKelvey School of Engineering at Washington University Open Scholarship. It has been accepted for inclusion in McKelvey School of Engineering Theses & Dissertations by an authorized administrator of Washington University Open Scholarship. For more information, please contact digital@wumail.wustl.edu.
Improving Additional Adversarial Robustness for Classification

By

Michael Guo

A thesis presented to the McKelvey School of Engineering of Washington University in St. Louis in partial fulfillment of the requirements for the degree of Master of Science

May 2021

St. Louis, Missouri
# Contents

Acknowledgments ................................................. iv  
Abstract ......................................................... vi  

1 Introduction ................................................... 1  

2 Background ..................................................... 3  
  2.1 General Goal of Classification ............................. 3  
  2.2 Adversarial Examples ........................................ 4  
    2.2.1 Continuous Space Adversarial Examples ............... 4  
    2.2.2 Discrete Space Adversarial Examples ....... 5  
  2.3 Robust Learning ............................................... 6  
    2.3.1 Adversarial Training .................................. 6  
    2.3.2 Randomized Smoothing ................................ 7  
    2.3.3 Fourier Stabilization .................................. 7  
  2.4 Regularization .............................................. 8  

3 Generalized Adversarial Training with Different Malicious Example .... 9  
  3.1 Dual-PGD: Generalized Attack for Adversarial Training in Image Classification ... 9  
    3.1.1 Motivation .............................................. 9  
    3.1.2 Dual-Perturbation Attacks ............................... 11  
    3.1.3 Identifying Foreground and Background .............. 13  
  3.2 Defense against Dual-Perturbation Attacks .................. 14  
    3.2.1 Adversarial Training with Dual-perturbation .......... 15  
    3.2.2 MixTrain with Dual-perturbation ...................... 15  
  3.3 Experiment on Image Classification ....................... 16  
    3.3.1 Experimental Setup ................................... 16  
    3.3.2 Dual-perturbation Attacks on Robust Classifiers .... 17  
    3.3.3 Generalizability of Defense .......................... 21  
  3.4 Experiment on Binary Inputs ................................ 23  
    3.4.1 Dataset and Computing Infrastructure ................. 23  
    3.4.2 Attacks and Defense Methods .......................... 24  
    3.4.3 $l_1$ BB Attacks ...................................... 25  

4 Increasing Robustness Using Gradient Free Regularization .......... 27
4.1 Motivation .................................................. 27
4.2 Robust SVM and Layer-Wise Regularization .................. 28
   4.2.1 Robust SVM .......................................... 28
   4.2.2 First Layer Regularization ......................... 29
   4.2.3 Choice of Loss Function ............................ 30
   4.2.4 Generalized Layer-wise Regularization ............... 31
4.3 Baseline Method ........................................... 31
   4.3.1 Adversarial Training ................................ 32
   4.3.2 FGSM .................................................. 32
4.4 Experiment .................................................. 33
   4.4.1 Experiment Setup .................................... 33
   4.4.2 Experiment on MNIST ................................ 34
   4.4.3 Experiment on Cifar10 .............................. 35
   4.4.4 Loss function Comparasion .......................... 37
4.5 Conclusion .................................................. 38

5 Conclusion and future work ................................ 39

References ......................................................... 41
Acknowledgments

A special thanks goes to the many graduate students and distinguished faculty within my department who have reviewed this thesis and helped support the related research.

Michael Guo

Washington University in Saint Louis
May 2021
Dedicated to my parents.
ABSTRACT OF THE THESIS

Improving Additional Adversarial Robustness for Classification

by

Michael Guo

Master of Science in Computer Science

Washington University in St. Louis, May 2021

Research Advisor: Professor Yevgeniy Vorobeychik

Although neural networks have achieved remarkable success on classification, adversarial robustness is still a significant concern. There are now a series of approaches for designing adversarial examples and methods to defending against them. This paper consists of two projects. In our first work, we propose an approach by leveraging cognitive salience to enhance additional robustness on top of these methods. Specifically, for image classification, we split an image into the foreground (salient region) and background (the rest) and allow significantly larger adversarial perturbations in the background to produce stronger attacks. Furthermore, we show that adversarial training with dual-perturbation attacks yield classifiers that are more robust to these than state-of-the-art robust learning approaches and comparable in robustness to conventional attacks. We also incorporate a stabilization process for binary inputs after the regular defense method to increase robustness.

1 This is a project I worked with Liang Tong with his major contribution
2 This is a project I worked with Aidan Kelly with his major contribution
In the second part of our work\textsuperscript{3}, we introduce a naive method that requires much less computation than other state-of-the-art methods, which adds regularization to the first layer of the neural networks. We also provide a generalized version that could apply to more complicated neural networks and empirically prove that our method has comparable robustness with baseline methods and is much faster.

\textsuperscript{3}This is a project with my only contribution
Chapter 1

Introduction

Neural networks model has set the capability of machine learning to a new stage. To reach better performance, the structure of neural networks have been more and more complicated. The delicacy of models comes at the expense of vulnerability. An observation by Szegedy [40] found that they can fabricate an adversarial image from a normal image. Those type of example is hard for people to distinguish between the two, but it will lead to a wrong prediction by the neural network with high probability and is called adversarial example. A series of studies have been done regarding how to find those adversarial examples [13, 6, 47]. Correspondingly, computer scientists have been developing new mechanisms to prevent those malicious perturbations. [32, 25].

Decision-time attack, defined by manipulating the data to fool the classifier at its prediction time, can be generally separated into two types, those need computing gradient[40] and those do not[6]. The reason behind this is due to the divergence of how to make those noises invisible to humans. In general, the constraint of gradient based-methods is some small $\ell_p$ norm which has a large chance to be neglected by human eyes. Those non-gradient-based methods usually do not modify images at pixel level but preserve the good semantic meaning of the image, for example, rotating or stretch objects in the image.

This project consists of two parts. In the first part, we propose a simple formalization of an important aspect of what makes adversarial perturbations unsuspicious. Specifically, we make a distinction between image foreground and background, allowing significantly more noise in the background than the foreground while maintaining the strong semantic meaning of the image. Further more, we used these dual-perturbation attacks for Adversarial Training.
as a new defense method that outperforms the state-of-the-art methods. Our first contribution is a formal model of such dual-perturbation attacks, which is a generalization of the $l_p$-norm-bounded attack models using object detection and the second using segmentation. Then, we present two methods for defending against dual-perturbation attacks, both based on the adversarial training framework [26]. We present an extensive experimental study that demonstrates that (a) the proposed attacks are significantly stronger than PGD, successfully defeating all state-of-the-art defenses, (b) proposed defenses using our attack model significantly outperform state-of-the-art alternatives, with relatively small performance degradation on non-adversarial instances. Finally, we incorporate stabilization when the inputs are binary to enhance additional robustness.

We proposed another gradient-free method in our second work, which is adding regularization to the model and use hinge loss as the loss function for the classifier. Experiments show that this method can make the model robust when the neural network is shallow and number of class is small. Under these circumstances, regularization is able to defend the attacks method almost 100% of the time and with hardly sacrifice in the accuracy on clean data. For the more general settings, we also provide a generic way to improve the robustness using additional regularization.
Chapter 2

Background

In this section, we provide background information of adversarial machine learning starting with the general goal of classification. Subsequently, notations of adversarial examples will be given and the heuristics to compute them will be presented under when the sample space are both continuous and discrete. We would then state the objective of robust learning, and the corresponding defense method. Finally, we will introduce regularization and how to use it to improve robustness.

2.1 General Goal of Classification

Neural networks has been used for image classification for more than decades [19], and a classifier can be defined as following. A model $h_\theta$ parameterized by the hyperparameter $\theta$ which is usually the weight of the model, some observations $(x_i, y_i) \in D$ where $x_i$ is the data point and $y_i$ is the corresponding true label. Given a loss function, $\mathcal{L}(h_\theta(x_i), y_i)$, measuring the deviation of the output of the model from the true label, a natural goal would be to find the best hyperparameter $\theta$ that minimized the averaged loss on the whole dataset $D$ with size of $|D|$. Mathematically, the goal is

$$
\min_\theta \frac{1}{|D|} \sum_{(x_i, y_i) \in D} \mathcal{L}(h_\theta(x_i, y_i))
$$

which is to find the population mean minimizer.
2.2 Adversarial Examples

Adversarial examples in image classification, images that look benign to human eyes but can make machine learning model mis-classify, has been explored and studied since Szegedy et al. [40] in 2013. Since then, a series of approaches has been developed to construct adversarial examples [14, 40, 5]. To craft adversarial examples that looks benign to human eyes, one common trick is to make perturbation small so that is invisible to human eyes. Naturally, we can model adversarial example as maximizing the loss with given constraint, which in general setting is certain $\ell_p$ norm. Formally, the objective of a adversarial example is:

$$\max_{\delta \in \Delta} \mathcal{L}(h_\theta(x_i + \delta), y_i)$$ (2.2)

where $\Delta = \{ \delta : \|\delta\|_p \leq \epsilon \}$ and is bounded by the constraint $\epsilon$ so that when epsilon is not large, the perturbation can be invisible to human eyes. Therefore attackers' objective on the whole dataset are the following:

$$\max_{\delta_i \in \Delta} \frac{1}{|\mathcal{D}|} \sum_{(x_i, y_i) \in \mathcal{D}} \mathcal{L}(h_\theta(x_i + \delta_i), y_i)$$ (2.3)

To solve this, and a number of methods has been proposed [14] and we will introduce them in following subsections.

2.2.1 Continuous Space Adversarial Examples

Most of the adversarial attacks are designed relying on the assumption that the sample space of the observations is continuous, though some of them still function well when the sample space is discrete. we focus one of the state-of-the-art approach is $p$Projected Gradient Descent [27] with $\ell_{\infty}$ and $\ell_2$ as the distance metrics.

In the PGD approach, we solve the problem in Eq.2.2 by iteratively updating the adversarial perturbation as follows:
\[ \delta^{(k+1)} = \mathcal{P}_\epsilon(\delta^{(k)} + \alpha \mathcal{G}(\nabla_{\delta^{(k)}} \mathcal{L}(h_\theta(x + \delta^{(k)}), y))) \]  

(2.4)

where \( \delta^{(k)} \) is the perturbation at the \( k \)-th iteration, \( \alpha \) is the step size in each iteration, \( \mathcal{P}_\epsilon \) is a projection to ensure that \( \delta^{(k)} \) is feasible, for example, \( \|\delta\|_p \leq \epsilon \) and \( x + \delta^{(k)} \) is clipped to feasible pixel values; and \( \mathcal{G} \) returns the update corresponding to the \textit{normalized steepest descent} given the gradient. Note that both the realization of \( \mathcal{P}_\epsilon \) and \( \mathcal{G} \) depend on the \( \ell_p \) distance used in the PGD attack. Specifically, for the \( \ell_\infty \) PGD attack,

\[
\begin{align*}
\mathcal{G}(m) &= \text{sign}(m) \\
\mathcal{P}_\epsilon(n) &= \text{clip}(n, [-\epsilon \mathbf{1}, +\epsilon \mathbf{1}] \cap [-x, 1 - x])
\end{align*}
\]

(2.5)

where \( \text{sign} \) returns the sign matrix of an input, \( \mathbf{1} \) is an all-ones matrix that has the same dimensions as \( \delta^{(k)} \), and \( \text{clip} \) is an operator that clip the each element of a given input in an element-wise manner. When it comes to the \( \ell_2 \) PGD attack,

\[
\begin{align*}
\mathcal{G}(m) &= \frac{m}{\|m\|_2} \\
\mathcal{P}_\epsilon(n) &= \text{clip}(\frac{en}{\max\{e, \|n\|_2\}}, [-x, 1 - x])
\end{align*}
\]

(2.6)

Since the adversarial optimization problem is non-convex, PGD is typically augmented by using a number of random starting points for adversarial perturbations that are close to the original input \( x \).[27]

### 2.2.2 Discrete Space Adversarial Examples

Specifically, we care about the situation that the feature space is binary if it is discrete, and focus on state-of-the-art \textit{Jacobian-based Saliency Map Attack} (JSMA) [30] under this condition. Although it was not originally designed for binary feature space only, we can easily modify the iterative update routine to make JSMA attack work in binary feature space. Initially, the problem is formulated as following. Given a model \( h_\theta \) and a input \( x \)

---

4Generally, images are preprocessed such that pixels are divided by 255 for computational convenience in training and testing. Consequently, a feasible pixel value should lie in \([0,1]\).
with label $y$, the adversarial saliency value of the $i$th feature in $x$, $S(x, y)[i]$, can be computed by:

$$S(x, y)[i] = \begin{cases} 0 & \text{if } \frac{\partial F_y}{\partial x_i}(x) < 0 \text{ or } \sum_{j \neq y} \left| \frac{\partial F_j}{\partial x_i}(x) \right| > 0 \\ \frac{\partial F_y}{\partial x_i} \sum_{j \neq y} \left| \frac{\partial F_j}{\partial x_i}(x) \right| & \text{otherwise} \end{cases}$$

(2.7)

where $J_F = [\frac{\partial F_i}{\partial x_j}]_{ij}$ is the model’s Jacobian matrix. Input components $i$ are added to some perturbation in the order of decreasing adversarial saliency value $S(x, y)[i]$. And in binary input space, we change this update routine to flipping the input feature with the highest adversarial saliency.

### 2.3 Robust Learning

The robust learning problem, which takes the adversarial perturbation into account, is formulated by the following:

$$\min_{\theta} \frac{1}{|D|} \max_{\delta_i \in \Delta} \sum_{(x_i, y_i) \in D} \mathcal{L}(h_\theta(x_i + \delta_i), y_i)$$

(2.8)

This in general can be extremely computational expensive. Many heuristics has been proposed and proved empirically to achieve this goal. Among them the state-of-the-art methods are adversarial training[27] and FGSM[46], which are the two baseline models we compare with.

#### 2.3.1 Adversarial Training

The basic idea of adversarial training is to produce adversarial examples and incorporate these into the training process. Formally, adversarial training aims to solve the following robust learning problem:

$$\min_{\theta} \frac{1}{|D|} \sum_{x, y \in D} \max_{\|\delta\|_p \leq \epsilon} \mathcal{L}(h_\theta(x + \delta), y)$$

(2.9)

where $D$ is the training dataset.
In practice, this problem is commonly solved by iteratively using the following two steps [27]: 1) use a PGD (or other) attack to produce adversarial examples of the training data; 2) use any optimizer to minimize the loss of those adversarial examples. It has been shown that adversarial training can significantly boost the adversarial robustness of a classifier against $\ell_p$ attacks, and it can be scaled to neural networks with complex architectures.

### 2.3.2 Randomized Smoothing

A next methods for robust learning considers adding random perturbations to inputs at both training and test time. The basic idea is to construct a new smoothed classifier $g_\theta(\cdot)$ from a base classifier $h_\theta(\cdot)$ as follows: first, the base classifier $h_\theta(\cdot)$ is trained with Gaussian data augmentation with variance $\sigma^2$; then, for any input $x$ at test time, the smoothed classifier $g_\theta(\cdot)$ returns the class that has the highest probability measure for the base classifier $h_\theta(\cdot)$ when inputs are perturbed with isotropic Gaussian noise:

$$g_\theta(x) = \arg \max_c P(h_\theta(x + \eta) = c)$$

where $\eta \sim \mathcal{N}(0, \sigma^2 I)$.  

(2.10)

It has been shown that randomized smoothing can provide certified robustness to adversarial perturbations for $\ell_2$-norm-bounded attacks [9, 21].

### 2.3.3 Fourier Stabilization

*Fourier Stabilization* [34] is a defense technique works only when the input features are binary which based on Fourier analysis of boolean functions [29]. Unlike most of methods focusing on optimizing the weights of the model as a whole, *Fourier Stabilization* treat each neuron seperately as a linear classifier. For a classifier model $h_\theta$ with inputs $x$, the robustness of the classifier., is defined by standard [11] as following:

$$\mathbb{E}_x \inf \{r : \exists x' \in Ball^p_r(x), h_\theta(x') \neq h_\theta(x') \}$$

(2.11)
where $Ball_p^r(x)$ is the set of all elements that are of $l_p$-distance at most $r$ from $x$. Notice that, when the classifier is linear, the robustness of the classifier equals the $l_p$ distance to the decision hyperplane. Now suppose a neuron is a linear classifier $h(x) = \text{sign}(xw^T - b)$, we wish to transform the neuron parametrized by $(w, b)$ in order to maximize its robustness on the dataset, which is the average distance to the hyperplane. To achieve this goal, we choose a new weights and bias $(v, \mu)$, for this neuron. Thus, the problem can be formulated as following:

$$\max_v \mathbb{E}_x (xv^T - \mu)h(x)$$
$$S.t. \text{ if } p > 1 \|v\|_q^p = 1$$
$$\text{ if } p = 1 \|v\|_\infty = 1$$

(2.12)

2.4 Regularization

Regularization has been studied since last century invented by Andrey Nikolayevich Tikhonov. This method was first developed to find preferable solutions of a inverse problems. Namely, suppose we have a known matrix $A$, a known vector $b$ and wish to find a vector $x$ such that:

$$Ax = b$$

(2.13)

With regularization, we can formulate this problem as an optimization problem as following:

$$\min_x \|Ax - b\|^2_2 + \|\Gamma x\|^2_2$$

(2.14)

where $\|\|^2$ is the standard Euclidian norm. By adding additional constraints, we will be able to find the solutions with desired properties. One special case of regularization is ridge regression which has the form of:

$$\min_x \|Ax - b\|^2_2 + \|x\|^2_2$$

(2.15)

Notice that, it has a geometric interpretation of this formulation, which is projecting the solution $x$ to the $l_2Ball$.
Chapter 3

Generalized Adversarial Training with Different Malicious Example

3.1 Dual-PGD: Generalized Attack for Adversarial Training in Image Classification

In this section, we introduce a novel threat model, the dual-perturbation attack [41], against state-of-the-art adversarially robust image classification models.

Below, we first describe the motivation of the proposed attack. Then, we turn to technical details.

3.1.1 Motivation

Our threat model is motivated by the feature integration theory [42] in cognitive science: regions that have features that are different from their surroundings are more likely to catch a viewer’s gaze. Such regions are called salient regions, or foreground, while the others are called background. Accordingly, for a given image, the semantics of the object of interest is more likely to be preserved in the foreground, as it catches more visual attention of a viewer compared to the background. If the foreground of an image is corrupted, then the semantics of the object of interest is broken. In contrast, the same extent of corruption in the background nevertheless preserves the overall semantic meaning of the scene captured (see, e.g.,
Figure 3.1: Distinction between foreground and background. Left: Original image of a train. Middle: Adversarial example with $\ell_\infty$ bounded perturbations ($\epsilon = 60/255$) on the background, the semantic meaning (train) is preserved. Right: Adversarial example with $\ell_\infty$ bounded perturbations ($\epsilon = 60/255$) on the foreground, the semantics are broken.

Figure 3.1). Indeed, detection of salient regions, as well as the segmentation of foreground and background, have been extensively studied in computer vision [2]. These approaches either predict human fixations, which are sparse bubble-like salient regions sampled from a distribution [18], or salient objects that contain smooth connected areas in an image [16].

Despite this important cognitive distinction between foreground and background, essentially all of the attacks on deep neural networks for image classification make no such distinction, even though a number of other semantic factors have been considered [1, 28]. Rather, much of the focus has been on adversarial perturbations that are not noticeable to a human, but which are applied equally to the entire image. However, in security applications, the important issue is not merely that an attack cannot be noticed, but that whatever is observed is not suspicious. This is, indeed, the frame of reference for many high-profile physical attacks on image classification, which are clearly visible, but not suspicious because they hide in the “human psyche”, that is, are easily ignored [36, 12].

The main goal of the threat model we introduce next is therefore to capture more precisely the notion that an adversarial example is not suspicious by leveraging the cognitive distinction between foreground and background of an image.
3.1.2 Dual-Perturbation Attacks

At the high level, our proposed threat model involves producing small (imperceptible) adversarial perturbations in the foreground of an image, and larger perturbations in the background. This can be done by incorporating state-of-the-art attacks into our method: we can use one attack with small $\epsilon$ in the foreground, and another with a large $\epsilon$ in the background. Consequently, we term our approach dual-perturbation attacks. Note that these clearly generalize the standard small-norm (e.g., PGD) attacks, since we can set the $\epsilon$ to be identical in both the foreground and background.

Formally, the dual-perturbation attack solves the following optimization problem:

$$\max_{\|\delta \circ \mathcal{F}(x)\|_p \leq \epsilon_F, \|\delta \circ \mathcal{B}(x)\|_p \leq \epsilon_B} \mathcal{L}(h_\theta(x + \delta), y)$$

(3.1)

where $\mathcal{F}$ returns the mask matrix constraining the area of the perturbation in the foreground, and $\mathcal{B}$ returns the mask matrix restricting the area of the perturbation in the background, for an input image $x$. $\mathcal{F}(x)$ and $\mathcal{B}(x)$ have the same dimension as $x$ and contain 1 in the area which can be perturbed and 0 elsewhere. $\circ$ denotes element-wise multiplication for matrices. Hence, we have $x = \mathcal{F}(x) + \mathcal{B}(x)$ which indicates that any input image can be decomposed into two independent images: one containing just the foreground, and the other containing the background.

A natural approach for solving the optimization problem shown in Eq.(3.1) is to apply an iterative method, such as the PGD attack, as shown in (??). However, the use of this approach poses two challenges in our setting. First, as in the PGD attack, the problem is non-convex, and PGD only converges to a local optimum. We can address this issue by using random starts, i.e. by randomly initializing the starting point of the adversarial perturbations, as in [26]. Second, and unlike PGD, the optimization problem in Eq.(3.1) involves two hard constraints $\|\delta \circ \mathcal{F}(x)\|_p \leq \epsilon_F$ and $\|\delta \circ \mathcal{B}(x)\|_p \leq \epsilon_B$. Thus, the feasible region of the adversarial perturbation $\delta$ is not an $\ell_p$ ball, which makes computing the projection $\mathcal{P}_\epsilon$ computationally challenging in high-dimensional settings. To address this challenge, we split the dual-perturbation attack into two individual processes in each iteration, one for the adversarial perturbation in the foreground and the other for the background, and then
merge these two perturbations when computing the gradients, like a standard PGD attack. Specifically, we use the following steps to solve the optimization problem in Eq.(3.1):

1. **Initialization.** Start with a random initial starting point $\delta^{(0)}$. To do this, randomly sample a data point $\delta^{(0)}_F$ in $\ell_p$ ball $\Delta(\epsilon_F)$ and $\delta^{(0)}_B$ in $\Delta(\epsilon_B)$. Then, $\delta^{(0)}$ can be obtained by using $\delta^{(0)} = \delta^{(0)}_F \circ F(x) + \delta^{(0)}_B \circ B(x)$. This ensures that the initial perturbation is feasible in both foreground and background.

2. **Split.** At the $k$-th iteration, split the perturbation $\delta^{(k)}$ into $\delta^{(k)}_F$ for foreground and $\delta^{(k)}_B$ for background:

   \[
   \begin{align*}
   \delta^{(k)}_F &= \delta^{(k)} \circ F(x) \\
   \delta^{(k)}_B &= \delta^{(k)} \circ B(x)
   \end{align*}
   \] (3.2)

   Then update the foreground and background perturbations using the following rules accordingly:

   \[
   \begin{align*}
   \delta^{(k+1)}_F &= \mathcal{P}_\epsilon(\delta^{(k)}_F + \alpha_F \cdot g_F) \\
   \delta^{(k+1)}_B &= \mathcal{P}_\epsilon(\delta^{(k)}_B + \alpha_B \cdot g_B)
   \end{align*}
   \] (3.3)

   where $g_F$ is the update that corresponds to the normalized steepest descent constrained in the foreground, and $g_B$ for the background. Specifically,

   \[
   \begin{align*}
   g_F &= \mathcal{G}(F(x) \circ \nabla_{\delta^{(k)}} \mathcal{L}(h_\theta(x + \delta^{(k)}), y)) \\
   g_B &= \mathcal{G}(B(x) \circ \nabla_{\delta^{(k)}} \mathcal{L}(h_\theta(x + \delta^{(k)}), y))
   \end{align*}
   \] (3.4)

   where $\alpha_F$ s the stepsize for foreground, and $\alpha_B$ is the stepsize for background.

3. **Merge.** At the end of the $k$-th iteration, merge the perturbations obtained in the last step by using

   \[
   \delta^{(k+1)} = \delta^{(k+1)}_F + \delta^{(k+1)}_B.
   \] (3.5)

   $\delta^{(k+1)}$ is further used to derive the update for the normalized steepest descent at the next iteration.

4. Return to step 2 or terminate after either a fixed number of iterations.
In addition to deterministic classifiers that make a deterministic prediction for a test sample, our proposed attack can be adapted to stochastic classifiers that apply randomization at training and prediction time. For example, for classifiers using randomized smoothing, we can refine Eq.(3.1) as follows:

\[
\max_{\|\delta_0F(x)\|_p \leq \epsilon_F, \|\delta_0B(x)\|_p \leq \epsilon_B} \mathbb{E}_{\eta \sim \mathcal{N}(0, \sigma^2 I)} [\mathcal{L}(h_\theta(x + \delta + \eta), y)] \tag{3.6}
\]

where \(\sigma^2\) is the variance of the Gaussian data augmentation in randomized smoothing. The optimization problem in Eq.(3.6) can be solved by the same approach used for deterministic classifiers, with the following modification on Eq.(3.4) at the second step:

\[
\begin{align*}
\begin{cases}
g_F = \mathcal{G}(F(x) \circ \nabla_{\delta^{(k)}\eta} \mathbb{E}_{\eta}[\mathcal{L}(h_\theta(x + \delta^{(k)} + \eta), y)]) \\
g_B = \mathcal{G}(B(x) \circ \nabla_{\delta^{(k)}\eta} \mathbb{E}_{\eta}[\mathcal{L}(h_\theta(x + \delta^{(k)} + \eta), y)])
\end{cases}
\end{align*}
\tag{3.7}
\]

Now, the question that remains is how to partition an input image \(x\) into foreground, \(F(x)\), and background, \(B(x)\), which we address this next.

### 3.1.3 Identifying Foreground and Background

![Figure 3.2: An illustration of dual-perturbation attacks. The top uses semantic segmentation to identify foreground and background. The bottom uses object bounding box.](image)

---

5Note that the Gaussian perturbations are only used to compute the expectation of loss and are not in the resulting adversarial examples.
A natural question is how to identify foreground and background and we now turn to our
approach to compute the foreground and background masks for an input image: Given an
input $x$, we aim to compute $F(x)$, the foreground mask and $B(x)$, the background mask.
We use two approaches for this: object detection and segmentation.

Our first approach leverages the object detection approaches [35] to obtain the bounding
box of the object of interest as the foreground. A bounding box is a rectangular box that
can be determined by the axis coordinates in the upper-left corner, and the axis coordinates
in the lower-right corner of the rectangle. Once we obtained all the bounding boxes of
a given image, we can filter out those that are irrelevant to the object of interest and
combine the remaining boxes such that the area inside the union of the boxes is identified as
foreground, and the area outside is identified as background. Afterward, the corresponding
foreground and background masks can be obtained in a similar way to the one with semantic
segmentation.

Our second approach is to make use of semantic segmentation to provide a partition of the
foreground and background in pixel level. This can be done in two steps: First, we use state-
of-the-art paradigms for semantic segmentation (e.g., [24]) to identify pixels that belong to
each corresponding object, as there might be multiple objects in an image. Next, we identify
the pixels that belong to the object of interest as the foreground pixels, and the others
as background pixels. Afterward, we obtain the foreground mask by setting corresponding
pixels with value 1s, and others with 0s. The background mask can be obtained using the
reverse way: the foreground pixels are set to be 0s, while pixels in the background are set to
be 1s. We use both of the above approaches in dual-perturbation attacks when evaluating
the robustness of classifiers, as well as designing robust models. More details are available
in the following section.

3.2 Defense against Dual-Perturbation Attacks

Once we are able to compute the dual-perturbation attack, we can incorporate it into con-
tentional adversarial training paradigms for defense, as it has been demonstrated that ad-
versarial training is highly effective in designing classification models that are robust to a
given attack. Specifically, we propose two defense approaches that we describe next.
3.2.1 Adversarial Training with Dual-perturbation

We replace the PGD attack in the adversarial training framework proposed by [26], with the proposed dual-perturbation attack. We term this approach \textit{AT-Dual}, which aims to solve the following optimization problem:

\[
\min_{\theta} \frac{1}{|D|} \sum_{x,y \in D} \max_{\|\delta_{F}(x)\|_p \leq \epsilon_{F}, \|\delta_{B}(x)\|_p \leq \epsilon_{B}} \mathcal{L}(h_{\theta}(x + \delta), y)
\]  

(3.8)

3.2.2 MixTrain with Dual-perturbation

As pointed out by [45], conventional adversarial training can at times fail to converge for very strong attacks, and often significantly degrades clean data accuracy. Conventional adversarial training can be ineffective to achieve robustness as it may not converge well if a strong attack model is applied. In addition, using adversarial training may decrease accuracy on clean data [26, 45].

To address these concerns, we incorporate dual-perturbation attack with MixTrain [45], a variant of adversarial training. The resulting approach, \textit{Mix-Dual}, is formulated as follows:

\[
\min_{\theta} \frac{1}{|D|} \sum_{x,y \in D} \max_{\|\delta_{F}(x)\|_p \leq \epsilon_{F}, \|\delta_{B}(x)\|_p \leq \epsilon_{B}} \beta \mathcal{L}_{1} + (1 - \beta) \mathcal{L}_{2}
\]  

(3.9)

where $\mathcal{L}_{1}$ is the loss on adversarial examples such that

\[
\mathcal{L}_{1} = \mathcal{L}(h_{\theta}(x + \delta), y),
\]  

(3.10)

and $\mathcal{L}_{2}$ is the loss on clean examples.

\[
\mathcal{L}_{2} = \mathcal{L}(h_{\theta}(x), y).
\]  

(3.11)

$\beta \in [0, 1]$ is a parameter that trades off between $\mathcal{L}_{1}$ and $\mathcal{L}_{2}$. When $\beta = 1$, \textit{Mix-Dual} is reduced to \textit{AT-Dual}. 

15
Note that both AT-Dual and Mix-Dual need to apply object detection method to identify background and foreground for any input when solving the inner maximization problems in Eq. (3.8) and Eq. (3.9) at training time. At prediction time, our approaches classify test samples like any standard classifiers, which is independent of the semantic partitions so as to close the backdoors to attacks on object detection approaches [48]. We evaluate the effectiveness of our approaches in the following chapter.

### 3.3 Experiment on Image Classification

In the section, we evaluate the effectiveness of the proposed dual-perturbation attack and defense approaches (AT-Dual and Mix-Dual) on a variety of datasets. We will start with a broad introduction to the experimental setup and then proceed to describe the experimental method and results in detail.

#### 3.3.1 Experimental Setup

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Image size</th>
<th>Training set size</th>
<th>Testing set size</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment-6</td>
<td>32 × 32 × 3</td>
<td>18,000</td>
<td>1,200</td>
<td>6</td>
</tr>
<tr>
<td>STL-7</td>
<td>96 × 96 × 3</td>
<td>1,246</td>
<td>500</td>
<td>7</td>
</tr>
<tr>
<td>ImageNet-10</td>
<td>224 × 224 × 3</td>
<td>3,327</td>
<td>344</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 3.1: Evaluation datasets.

**Datasets**

We conducted the experiments on the following three datasets, listed in Table 3.1. The first is Segment-6, obtained by pre-processing the Microsoft COCO dataset [23] to make it compatible with image classification tasks. We used semantic segmentation to obtain the foreground masks on this dataset (see the Supplement for details). We include data pre-processing and foreground identification in Appendix. Our second dataset is STL-7, a subset of the STL-10 dataset [8]. We chose this subset by filtering out those the objects of interest of which were not correctly detected by YOLOv3 [35], the state-of-the-art object
detection method. For each image in the dataset, we used YOLOv3 to identify the bounding box of the object of interest as the foreground.

Our third dataset is ImageNet-10, a 10-class subset of the ImageNet dataset [10]. We chose this subset of images and obtained the corresponding foreground masks by using the same approach as the STL-7 dataset.

**Baseline**

The performance of the proposed approach was compared with two alternative paradigms that have been proved to boost adversarial robustness: *Adversarial Training with PGD Attacks* (henceforth, *AT-PGD*) proposed in [26], the state-of-the-art adversarially robust training scheme widely used in practice; *Randomized Smoothing* (henceforth, *RS*) in [9], which achieves provably robust classification against $\ell_2$ attacks. Additionally, we compared our approach with the classifier trained on non-adversarial data (henceforth, *Clean*).

**Evaluation Method.** For a given classification model, we evaluated its performance in two ways: 1) Evaluated the accuracy of prediction on clean test data where no adversarial attacks were attempted. 2) Evaluated its adversarial robustness against different attacks by using the accuracy of prediction on corresponding adversarial examples.

We used both $\ell_2$ and $\ell_\infty$ norms as the distance metrics to implement attacks, as well as the robust classification models that are based on adversarial training when doing evaluation. The implementations of attacks and defense approaches are detailed in the Appendix. Due to space limitations, we only present experimental results of the *Clean* model and classification models that are trained to be robust to $\ell_2$ norm attacks. The results for $\ell_\infty$ norm are similar and deferred to the Supplement.

**3.3.2 Dual-perturbation Attacks on Robust Classifiers**

We evaluate the effectiveness of dual-perturbation attacks against state-of-the-art robust learning methods, as well as the effectiveness of adversarial training variants that use dual-perturbation attacks for generating adversarial examples. For the purpose of comparison between our attack and the traditional PGD attack, we applied both attacks to evaluate the
adversarial robustness of our defense approaches and the baselines. Specifically, for a given classification model, we first evaluated its adversarial robustness to white-box PGD and dual-perturbation attacks; we then proceeded to study its robustness to transferred adversarial examples, that is, its accuracy on adversarial examples produced by white-box attacks on other classification models. We begin by considering white-box attacks, and subsequently evaluate transferability.

Figure 3.3: Robustness to white-box $\ell_2$ attacks with a variety of distortions on Segment-6 (top row), STL-7 (middle row), and ImageNet-10 (bottom row). AT-PGD, AT-Dual and Mix-Dual are trained by using $\ell_2$-norm attacks with hyper-parameters listed in Table ?? in the Supplement. RS uses $\sigma = 0.5$. Left: dual-perturbation attacks. Right: PGD.

The results for white-box attacks are presented in Figure 3.3. First, consider the dual-perturbation attacks (left plots). Note that in all cases these attacks are highly successful against both baseline robust classifiers (AT-PGD and RS); indeed, even relatively small
levels of foreground noise yield near-zero accuracy when accompanied by sufficiently large background perturbations. For example, when the perturbation to the foreground is $\epsilon_F = 1.0$ and background perturbation is $\epsilon_B = 15.0$ on STL-7 data, $RS$ achieves robust accuracy below 10%, while $AT-PGD$ only has 5% robust accuracy. In contrast, both of the defense approaches discussed, AT-Dual and Mix-Dual remain significantly more robust, with an improvement of up to 40% compared to the baselines, without sacrificing much in accuracy on clean data. Second, consider the standard PGD attacks (right plots). It can be observed that all of the robust models are successful against the $\ell_2$ PGD attacks. However, in the cases of the Segment-6 and STL-7 datasets, our defenses exhibit moderately higher robustness than the baselines under large distortions of PGD attacks. For example, when the perturbation of the $\ell_2$ PGD attack is $\epsilon = 2.0$ on the ImageNet-10 data, $AT-Dual$ and $Mix-Dual$ can achieve 10% more robust accuracy.

It can be seen that both the baselines and the proposed approaches can significantly improve robustness to $\ell_2$ PGD attacks compared to the $Clean$ model. Specifically, our approach is more robust compared to $RS$, with an up to 20% higher robustness on Segment-6 and up to 10% higher robustness on STL-7. Compared to $AT-PGD$, our approaches exhibit comparable robustness under small distortions, and slightly higher robustness under large distortion of attacks (e.g., when $\epsilon = 1$ on Segment-6, as shown in Figure 3.3 (left)). Moreover, when comparing $AT-Dual$ and $Mix-Dual$, one can observe that the latter has up to $\sim 5\%$ improvement of accuracy on clean data, which is comparable to $AT-PGD$, while the former can achieve slightly higher robustness. Overall, classification models that are trained to be robust to dual-perturbation attacks are comparable to, or better than the baselines against standard PGD attacks, with relatively small performance degradation on clean test data.

Next, we measure the transferability of adversarial examples among different classification models. To do this, we first produced adversarial examples by using $\ell_2$ PGD attack or dual-perturbation attack on a source model. Then, we used these examples to evaluate the performance of an independent target model, where a higher prediction accuracy means weaker transferability.

The results are presented in Figure 3.4. The first observation is that dual-perturbation attacks exhibit significantly better transferability than the conventional PGD attacks (transferability is up to 40% better for dual-perturbation attacks). Second, we can observe that
Figure 3.4: Transferability of adversarial examples on Segment-6 (top row), STL-7 (middle row), and ImageNet-10 (bottom row). AT-PGD, AT-Dual and Mix-Dual are trained by using corresponding $\ell_2$-norm attacks with hyper-parameters listed in Table ?? in the Supplement. RS uses $\sigma = 0.5$. Left: dual-perturbation attacks. Right: PGD.

When either $AT$-Dual or Mix-Dual are used as the target (i.e., defending by adversarial training with dual-perturbation examples), these are typically resistant to adversarial examples generated against either the clean model, or against RS and AT-PGD. This observation obtains even when we use PGD to generate adversarial examples.
Figure 3.5: Robustness to additional white-box attacks on Segment-6 (top row), STL-7 (middle row), and ImageNet-10 (bottom row). AT-PGD, AT-Dual and Mix-Dual are trained by using corresponding $\ell_2$-norm attacks with hyper-parameters listed in Table ?? in Appendix. Left: $\ell_\infty$ PGD attacks. Middle: $\ell_\infty$ dual-perturbation attacks. Right: $\ell_0$ JSMA attacks.

3.3.3 Generalizability of Defense

It has been observed that robustness against $l_p$-norm-bounded attacks for one value of $p$ can be fragile when facing attacks with a different norm $l_q$ [37].

Our final goal is to present evidence that the approaches for defense based on dual-perturbation attacks remain relatively robust even when faced with attacks generated using different norms. Here, we show this when our models are trained using the $l_2$-bounded attacks, and evaluated against other attacks using other norms. Data for models trained using $l_1$-bounded attacks is provided in the Supplement.

The results are in Figure 3.5. We present the results for three datasets: Segment-6 (where foreground is determined using segmentation), top row, STL-7, middle row, and ImageNet-10, bottom row. We consider three alternative attacks: 1) PGD using the $l_\infty$-bounded perturbations, as in Madry et al. [26] (left column in Figure 3.5) 2) dual-perturbation attacks with $l_\infty$-norm bounds (middle column in Figure 3.5), and 3) JSMA, a $l_0$-bounded attack [31]...
(right column in Figure 3.5). We additionally considered $l_2$ attacks, per Carlini and Wagner [7], but find that all of the robust models, whether based on PGD or dual-perturbation attacks, are successful against these.

Our first observation is that both AT-Dual and Mix-Dual are significantly more robust to $l_\infty$-bounded PGD attacks than the adversarial training approach in which adversarial examples are generated using $l_2$-bounded PGD attacks. Moreover, the magnitude of the advantage appears to increase with more complex datasets: the gap is larger for STL-7 and ImageNet-10 compared to Segment-6. Consequently, training with dual-perturbation attacks already exhibits better ability to generalize to other attacks compared to conventional adversarial training.

The gap between dual-perturbation-based adversarial training and standard adversarial training is even more significant when we consider $l_\infty$ dual-perturbation attacks (middle column). Here, we see that robustness of PGD-based adversarially trained model is only marginally better than that of a clean model, whereas both AT-Dual and Mix-Dual remain relatively robust.

Finally, considering JSMA attacks, we can observe that on two of the three datasets (Segment-6 and ImageNet), conventionally adversarially trained model remains relatively robust. However, STL-7 exhibits a case where this model is indeed fragile to $l_0$-bounded attacks. In contrast, in all of the cases, the model made robust using dual-perturbation attacks remains quite robust even as we evaluate against a different attack, using a different norm.
3.4 Experiment on Binary Inputs

When inputs have binary feature, we can incorporate *Fourier Stabilization* as a complementary defense method besides *Adversarial Training* [26] to further enhance the robustness of neural network. Since *Fourier Stabilization* is much faster than *Adversarial Training*, it’s worthy to enhance robustness additionally with sacrifice of negligible time.

3.4.1 Dataset and Computing Infrastructure

We evaluated the proposed approach using three security-related datasets: PDFRate, Hidost, and Hate Speech. The PDFRate dataset [39] is a PDF malware dataset which extracts features based on PDF file metadata and content. The metadata features include the size of a file, author name, and creation date, while content-based features include position and counts of specific keywords. This dataset includes 135 total features, which are then binarized if not already binary. The Hidost dataset[43] is a PDF malware dataset which extracts features based on the logical structure of a PDF document. Specifically, each binary feature corresponds to the presence of a particular structural path, which is a sequence of edges in the reduced (tree) logical structure, starting from the catalog dictionary and ending at this object (i.e., the shortest reference path to a PDF object). This dataset is comprised of 658,763 PDF files and 961 features. The Hate Speech dataset [33], collected from Gab, is comprised of conversation segments, with hate speech labels collected from Amazon Mechanical Turk workers. This dataset contains 33,776 posts, and we used a bag-of-words binary representation with 200 most commonly occurring words (not including stop words).

All datasets were divided into training, validation, and test subsets; the former two were used for training and parameter tuning, while all the results below are using the test data. For each dataset, we learned a two-layer sigmoidal fully connected neural network as a baseline. Experiments were run on a research computer cluster with over 2,500 CPUs and 60 GPUs.
3.4.2 Attacks and Defense Methods

The robustness-accuracy tradeoff is quantified by the success rate of two state-of-the-art attacks, JSMA and $l_1$BB, under limited budget. *Jacobian-based Saliency Map Attack (JSMA)*[30] (naturally adapted to the $\{\pm1\}$ domain rather than $\{0,1\}$), employs a greedy heuristic by which the bit with the highest impact is flipped. $l_1$ Brendel & Bethge [3] is an attack that allows non-binary perturbations. It is radically different from JSMA in the sense that it requires an already adversarial starting point which is then optimized. Given a clean point to attack, we select the adversarial starting point as the closest to it in $l_1$-distance, among all points in the training set.

In addition to the conventional baseline above, we also evaluated the use of neural network stabilization after *Adversarial Training*, which is still a state-of-the-art general purpose approach for defense against adversarial example attacks. We performed AT with the JSMA attack ($l_1$-norm $\epsilon = 20$), which we adapted as follows: instead of minimizing the number of perturbed features to cause misclassification, we maximize loss subject to a constraint that we change at most $\epsilon$ features, still choosing which features to flip in the sorted order produced by JSMA.
Figure 3.6: Robustness of neural network after AT and their stabilized variants. Top row: PDFRate dataset, after AT with 1, 5, 10 epoch(s) from left to right. Bottom row: Hidost dataset after 1(left) and 4(right) epoch(s) of AT. The $x$-axis represents varying level of $l_1$ perturbation bound $\epsilon$ for the BB attack. $\ell_0$ JSMA attacks.

### 3.4.3 $l_1$ BB Attacks

In addition to demonstrating the value of stabilization for regularly trained neural networks (for example, when adversarial training is not an option, such as when datasets on which the original model was trained are sensitive), we now show that the approach also effectively composes with adversarial training (AT). Figure 3.6) presents the results of stabilization performed after several epochs of AT. In all cases we see some improvement, and in a number of them the improvement over AT is considerable. For example, on the Hidost dataset after 4 epochs of AT, robust accuracy is considerably improved by AT compared to the original model in but then further improved significantly by the proposed stabilization approach. For example, for $\epsilon = 24$, robust accuracy increases from approximately 20% to 80%
Figure 3.7: Robustness of neural network after AT and their stabilized variants. Top row: PDFRate dataset, after AT with 1, 10 epoch(s) from left to right. Bottom row: Hidost dataset after 1(left) and 4(right) epoch(s) of AT. The x-axis represents varying level of $l_1$ perturbation bound $\epsilon$ for the JSMA attack.
Chapter 4

Increasing Robustness Using Gradient Free Regularization

4.1 Motivation

As people have rising attention to the robustness of deep neural networks since 2013 [40], a series of attack and defend methods have been proposed [27, 46]. Although they have different assumptions and constraints, these attacks are generally very computation expensive. The most popular gradient based attack is called the \textit{PGD} (Projected gradient descent) attack, which needs the gradient of the loss function to the input multiple times for each image. Gradient computation is very slow in general, especially when network architecture is complicated. The state-of-the-art defense method against \textit{PGD} attack is called \textit{Adversarial training} [27] which needs to generate the PGD adversarial examples for the whole training dataset at each training epoch. Other methods, such as \textit{FGSM} (Fast Gradient Sign Method) [46], was developed to accelerate the \textit{AT} process and maintain the same level of robustness but still require gradient computation which might be ineffective.

We proposed a method that originated from \textit{Robust Linear SVM} [22], and we generalize the idea to DNN, and provides extensive experiments to show that regularization could have comparable robustness with nearly no loss of performance on clean data under additional constraints.
4.2 Robust SVM and Layer-Wise Regularization

4.2.1 Robust SVM

Before introducing our method, we first recap the Robust SVM classifier that inspires our method. Consider a standard binary classification problem with linear SVM, let \( \mathcal{D} = \{x_i, y_i\}_{i=1}^m \) be the dataset with \( m \) samples and our goal is to find a linear classifier parameterized by \((w, b)\) such that \( h(x) = \text{Sign}(wx + b) \). The standard regularized SVM can be formulated as following:

\[
\min_{w,b} \|w\|_p + \sum_{i=1}^m \xi_i \\
\text{s.t. } \xi_i \geq [1 - y_i(wx_i + b)] \\
\xi_i \geq 0
\]  

(4.1)

Notice that this formulation equals the following:

\[
\min_{w,b} \|w\|_p + \sum_{i=1}^m \max[0, 1 - y_i(x_iw + b)]
\]  

(4.2)

Now, suppose an adversary can modify the sample with in \( \epsilon - \text{Ball} \) around the benign sample. Thus, the objective of robust learning for linear SVM is:

\[
\min_{w,b} \max \sum_{i=1}^m \max[0, 1 - y_i((x_i + \delta_i)w + b)]
\]

\[
\Rightarrow \min_{w,b} \sum_{i=1}^m \max[0, 1 - y_i((x_i + \delta_i)w + b)] + \epsilon\|w\|_q
\]  

(4.3)

where \( \frac{1}{p} + \frac{1}{q} = 1 \)

The prove of equivalence for two formulations above will not be included in this work, but is thoroughly documented in [44] and the intuition behind this is that the maximum perturbation is bounded by \( \epsilon \) times a factor that related to the weight \( w \).
4.2.2 First Layer Regularization

Recall that in Chapter 2, we define a classifier as $h_\theta(x)$. One interpretation for this is that it can be viewed as a function that maps from input space $x$ to a discrete class $\{1, 2, 3, 4, ... N\}$. Similarly, we can view a neural network of $k$ layers as a composition of functions where a given layer can be viewed as a function that map from the output of previous layer to the input of its next layer, namely,

$$h_\theta(x) = h_{\theta_k}(...h_{\theta_2}(h_{\theta_1}(x)))$$

$$= h_{\theta_k} \circ h_{\theta_{k-1}} \circ ... \circ h_{\theta_2} \circ h_{\theta_1}(x)$$ (4.4)

![Figure 4.1: A naive example of layer-wise interpretation of neural network. In this example, we can treat each layer as a separate function, and the whole network can be interpreted as a composition of 4 functions](image)

An example is also given in Figure 4.2.2 in which we could easily see that the final output can be viewed as $h_{\theta_3} \circ h_{\theta_2} \circ h_{\theta_1}(x)$ of input $x$. We can then separate any neural network into two parts, the input layer and the rest. An input layer can be viewed as a linear classifier, and the rest layers can be viewed as a linear classifier with *kernel trick.*

29
Our proposed defense method is naive but could be effective, we also provide extensive experiments later to support our results. The defense method is simply add the dual norm of regularization to the first layer of neural networks, and switching the loss function as multi-class hinge loss. Namely, if the constrain of adversarial perturbation is \( \|\delta\|_\infty \leq \epsilon \) then we add \( l_1 \) regularization; if the constrain is based on \( l_2 \) norm, then we add \( l_2 \) regularization to the first layer.

Consider a extreme case where there is only one neuron with linear activation function for the layers other than the first, then the whole model is essentially a linear classifier. Given the fact that \textit{Rosbust Linear SVM} will not be affected by any perturbation with in the \( \epsilon – Ball \), a neural network will preserve some robustness if the rest layers and the loss function are more 'linear'.

Also, if the first layer is robust, then the output of the first layer will be unlikely to be affected by the perturbation adversary added to the inputs. In other words, the adversarial noise can be 'filtered' by regularized layers. In reality, we find out that adding regularization to the first layer can not solve the problem entirely. Empirically, this method can preserve robustness and clean accuracy when the number of total class is small, with a much faster speed.

4.2.3 Choice of Loss Function

Unlike most classifiers nowadays use \textit{Cross Entropy Loss}, our method is based on \textit{Multi Margin Loss}, which is a generalized hinge loss for multi-class classification. The reason behind the specific choice of loss function is that we want to reduce the probability incurred by using a non-linear loss function. One can also interpret our method as a multi-class SVM with neural network as \textit{kernel trick}. If we use \textit{Cross Entropy Loss} for our loss function, then we know that for a given output of the model \( y \in \mathbb{R}^{1 \times m} \) with target class index \( t \in \{0, 1, 2, ..., m - 1\} \) where \( m \) is the number of class, the cross entropy loss can be written as the:

\[
\mathcal{L}(y, t) = -y[t] + \log(\sum_{j \neq t} \exp(x[j])) \tag{4.5}
\]
When we use *Multi Margin Loss*, the loss function is:

\[
\mathcal{L}(y, t) = \frac{\sum_{i \neq t} \max\{0, 1 - y[t] + y[j]\}}{m}
\]  \hspace{1cm} (4.6)

With the fact that *Linear SVM* with regularization can defend any \( \epsilon \) perturbation, we want our model could be as similar as a *Linear SVM* and that’s part of the reason why we switch to another loss function for our model.

### 4.2.4 Generalized Layer-wise Regularization

An inherent idea of generalization of the first layer regularization method is adding regularization to the succeeding layers one by one. Theoretically, this should increase the robustness by generating a more accurate term on the training objective. From the filtering view we mentioned in the previous section, adding layers of regularization will reduce the effect of the adversarial noise in the output. Since we know that strong regularization can lead to a random model, we could implement the idea similar to curriculum adversarial training \[4\] that gradually increase the strength of regularization for the hidden layers.

The regularization method can be implemented with deep convolutional neural network \[15, 38\] as well, where we add regularization to the first fully connected layer instead of the first layer of the entire network, and treat the convolutional layers as the feature extraction process. One can also freeze the weights of convolutional layers and then do transfer learning on custom dataset.

### 4.3 Baseline Method

The two baseline methods we compare with are *AT* and *FGSM*. Both methods requires retraining on top of a clean model. We have mentioned *AT* in previous chapter, and will present the algorithm in this section.
4.3.1 Adversarial Training

The procedure of AT is to generate adversarial examples for each mini-batch of inputs, and use only adversarial examples to re-train the clean model. Suppose we have a $N$ steps PGD adversarial training for $T$ epochs and number of batches of data as $M$, given norm bound as $\epsilon$, the step size as $\alpha$ for a clean model $h_\theta$.

**Algorithm 1: Adversarial training**

```
for $t=1:T$ do
    for $j=1:M$ do
        $\delta = 0$
        for $j=1:N$ do
            $\delta = \delta + \alpha \times sign(\nabla_\delta \mathcal{L}(h_\theta((X_i + \delta), Y_i)))$
            $\delta = Proj(\delta, \epsilon)$
        end
        $\theta = \theta - \nabla_\theta \mathcal{L}(h_\theta((X_i + \delta), Y_i))$
    end
end
```

Namely, AT generates adversarial example using PGD attacks in this case and using them as the training data.

4.3.2 FGSM

A similar FGSM training with training for $T$ epochs and number of batches of data as $M$, given norm bound as $\epsilon$, the step size as $\alpha$ for a clean model $h_\theta$. 

32
Algorithm 2: FGSM

\[
\begin{align*}
&\text{for } t=1:T \text{ do} \\
&\quad \text{for } i=1:M \text{ do} \\
&\quad\quad \delta = \text{sign}(\nabla_\delta \mathcal{L}(h_\theta(X_i, Y_i))) \\
&\quad\quad \theta = \theta - \nabla_\theta \mathcal{L}(h_\theta((X_i + \delta), Y_i)) \\
&\quad \text{end} \\
&\text{end}
\end{align*}
\]

The only difference is the method to generate adversarial examples; for FGSM, we only need to compute gradient once with respect to the input. By reducing the number of gradient computations, this method could be much faster than AT with PGD attacks.

4.4 Experiment

In this section, we will introduce our experiment results on different datasets. In addition to compare our method with those two state-of-the-art defense methods mentioned above, we also tested the results using regularization on models trained with Multi Margin Loss and Cross Entropy Loss to back up our hypothesis mentioned in the previous sections.

4.4.1 Experiment Setup

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Image size</th>
<th>Training set size</th>
<th>Testing set size</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>28 × 28 × 3</td>
<td>50,000</td>
<td>10,000</td>
<td>10</td>
</tr>
<tr>
<td>Cifar-10</td>
<td>32 × 32 × 3</td>
<td>50,000</td>
<td>10,000</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 4.1: Evaluation datasets.

Experiments were conducted on the following dataset, MNIST and Cifar-10 (as shown in the Table 4.1). The MNIST dataset [20] consists of black white image data of handwritten digits. The dataset contains 50,000 training data and 10,000 testing samples evenly distributed across 10 classes. Each data in the experiment contains 28 × 28 feature flattened the image to 1 × 784. To perform the proposed method as well as other experiments, we developed a two layers fully connected neural network with ReLu activation function from scratch.
The Cifar-10 dataset [17] is the dataset of color image with size of $32 \times 32 \times 3$ for each data. The whole dataset consists of 50,000 training samples and 10,000 testing samples almost evenly distributed across 10 classes. We used a custom convolutional neural network on this dataset and trained a model from scratch.

4.4.2 Experiment on MNIST

The experiment results are shown in Figure 4.2. We include the results on clean test data by setting $\delta = 0$. We compared our defense method with the two state-of-the-art defense methods we mentioned above, along with a model trained on clean data. Our intuition is to replicate a $RobustLinearSVM$ using neural networks first and then try to generalize the
results into multi-class classification tasks. To this end, we randomly selected 2 to 9 classes and evaluated the robustness as number of classes increases.

As shown in Figure 4.2, when the number of class is small, especially when there are only 2 or 3 classes, our regularization methods performed extremely well that it could successfully defend almost all attacks with little sacrifice on the performance of clean data. With the number of classes increasing, we shall see a sharp decrease in the performance of our proposed method. The penalty of regularization starts to undermine the performance of the clean model, and thus our method becomes ineffective.

4.4.3 Experiment on Cifar10

Figure 4.3: Robustness of neural network after AT, FGSM, and regularization. Robustness is proportion prediction correct when inputs are adversarially perturbed. The underlying model is custom convolutional neural network. Test accuracy on clean data is given as $\epsilon = 0$

We evaluate the defense method with $\|\delta\|_{\infty} = \{0, 2/255, 4/255, 8/255, 16/255, 32/255\}$. The factor of $l_1$ regularization here is $4/255$ and is applied to the first fully connected layer

35
To test the results of generalized regularization, we also tested our method with a more complicated neural network architecture, which is a customized convolutional neural network. As we proposed in the previous section, we added regularization to the first fully connected layer of the model. Similarly, we randomly selected 2 to 9 classes and evaluated the robustness as number of classes increases. A similar pattern can be observed in Fig 4.3 that as we include more classes in the classification, regularization becomes less effective. Notice that our method has comparable performance with baselines methods when it degenerates to a binary classification problem.
4.4.4 Loss function Comparison

Figure 4.4: Robustness of neural network after AT, FGSM, and regularization. Left column are the results using Multi Margin Loss, right column are the results using Cross Entropy Loss. Robustness is proportion prediction correct when inputs are adversarially perturbed. The underlying model is custom convolutional neural network. Test accuracy on clean data is given as $\epsilon = 0$. We evaluate the defense method with $\|\delta\|_{\infty} = \{0, 2/255, 4/255, 8/255, 16/255, 32/255\}$. The factor of $l_1$ regularization here is $4/255$ and is applied to the first fully connected layer.

In addition, we also evaluate the robustness using regularization with both Multi Margin Loss and Cross Entropy Loss. Figure 4.4 shows the value of switching to Multi Margin Loss as it consistently performs better as we increase the number of classes. In all cases, we can see some improvements and the improvements are especially considerable when the classes are fewer.


4.5 Conclusion

In this work, we proposed a naive approach by simply adding regularization to the first layer of fully connected layers to enhance the robustness of a neural network. The major limitation of this method is that it only performs well when the number of class is small. As we can see in the results, the performance downgrade as the number of classes increases. Also, usually the optimal optimal factor regularization term, $\epsilon^*$, can be very hard to find, and training with regularization can be extremely unstable — a strong regularization usually incurs a random model, and a week regularization can hardly improve robustness.

Even with the right norm within the feasible range, the relationship between robustness and strength of regularizers is not simply linear. In some cases, a weaker regularizer can make the model even more robust. Also, when the neural network is deep, add regularizer to the first layer can only increase limited robustness. Regularize later layers requires additional processes to determine the optimal factors. One potential improvement might be using Bayesian optimization in the future to find the optimal factors.

However, this project sheds light on how we should think of robustness for neural networks and its relationship with regularization. One of the main reasons why adversarial examples can found easily is that the underlying machine learning model is complicated such that they are local-nonlinear. We have shown in this project that: 1) It is possible to do multi-class classification task using multi-margin loss, which is a generalized version of hinge loss that alleviate the local-non linear, and 2) regularization can indeed increase robustness without any gradient computation.

To construct a robust classifier based on a neural network, one can start with binary classification and then extend to the multi-class scenario. Essentially, if regularization works for binary classification, then what we can do is to generalize it to multi-class by doing binary classification multiple times. To further improve the results on a convolutional neural network, one can add regularization to the fully connected layer.
Chapter 5

Conclusion and future work

In this work, we first proposed the dual-perturbation attack, a novel threat model that produces by leveraging the cognitive distinction between image foreground and background. As we have shown, our attack can defeat all state-of-the-art defenses. By contrast, the proposed defense approaches using our attack model can significantly improve robustness against unsuspicious adversarial examples, with relatively small performance degradation on non-adversarial data. In addition, our defense approaches can achieve comparable to or better robustness than the alternatives in the face of traditional attacks. We the inputs are binary, and where dual-perturbation is not working, we incorporate a stabilization process after AT to further enhance robustness.

We also propose a gradient-free regularization method in chapter 4 and provide extensive experimental results to show that our method is faster and has comparable results with the baseline methods when the number of the class is few, and the network architecture is relatively simple. We also proposed a generalized version of the regularization method in which we view the regularized layer as a filter against adversarial noises.

However, there are still some research questions that are raised by have not been resolved. The first is whether there are more effective methods to identify foreground of images and can we further improve robustness to dual-perturbation attacks. Besides, we have measure $l_1$ the distance the weights difference between the model before and after stabilized for both clean and adversarially trained model, and find a interesting observation that stabilization has a larger impact on the model that been adversarially trained. This rules out the assumption that adversarial training and stabilization is shaping the weights of neural network to a similar direction. Thus, the exact impact of stabilization on AT is still remaining unrevealed.
Finally, it’s possible that one can generalize stabilization method to real value inputs using harmonic analysis.

As for gradient-free regularization method, the major problem so far is that it performs poorly as the number of class increase. One potential solution to this is that designing a multi-class classification problem as a multi-round binary classification problem such that all the binary classifiers will preserve robustness and accuracy at the same time. Our future works will focus on these fields, and hopefully, this project can inspire future relevant research topics.
References


