

© 2020 Siwakorn Srisakaokul

MULTI-MODEL-BASED DEFENSE AGAINST ADVERSARIAL EXAMPLES FOR  
NEURAL NETWORKS

BY

SIWAKORN SRISAKAOKUL

THESIS

Submitted in partial fulfillment of the requirements  
for the degree of Master of Science in Computer Science  
in the Graduate College of the  
University of Illinois at Urbana-Champaign, 2020

Urbana, Illinois

Adviser:

Professor Tao Xie  
Assistant Professor Bo Li

## ABSTRACT

Neural networks recently have been used to solve many real-world tasks such as image recognition and can achieve high effectiveness on these tasks. Despite being popularly used in many applications, neural network models have been found to be vulnerable to *adversarial examples*, i.e., carefully crafted examples aiming to mislead machine learning models. Adversarial examples can pose potential risks on safety/security-critical applications. Existing defense approaches are still vulnerable to emerging attacks, especially in a white-box attack scenario. In this thesis, we focus on mitigating the adversarial attacks by improving machine learning models to be more robust against those attacks.

In particular, we propose a new defense approach, named MULDEF, based on *robustness diversity*. Our approach consists of (1) a general defense framework based on diverse models and (2) a technique for generating diverse models to achieve high defense capability. Our framework generates multiple models (constructed from the target model) to form a model family. The model family is designed to achieve robustness diversity (i.e., an adversarial example crafted to attack one model may not succeed in attacking other models in the family). At runtime, a model is randomly selected from the family to process each input example. Our evaluation results show that MULDEF (with only up to 5 models in the family) can substantially improve the target model’s robustness against adversarial examples by 19–78% in a white-box attack scenario among MNIST, CIFAR-10, and Tiny ImageNet datasets, while maintaining similar accuracy on legitimate examples.

Our general framework can also inspire rich future research to construct a desirable model family achieving higher robustness diversity.

*To my parents, for their love and support.*

## ACKNOWLEDGMENTS

During my graduate school, I have studied computer science in the University of Illinois at Urbana-Champaign (UIUC). My research journey started in the Illinois Automated Software Engineering Research (ASE) group at UIUC, working with my advisers, Professor Tao Xie and Assistant Professor Bo Li, along with support from many people.

I first would like to show my gratitude to Professor Tao Xie for his invaluable support since I joined the graduate school. He is knowledgeable and always provides useful guidance for students in developing their research career path. I learn so much from his great insight and ability in research. In addition, he often shares his personal experience outside research, which is very inspiring.

I would like to thank Assistant Professor Bo Li for her guidance in improving my research work. Her great critical thinking and ability in doing research really help me not only polishing my work but also broadening my ideas for future work.

I also would like to thank all the members of the ASE group for their help and collaboration. I learn so much from our discussion, and they always inspire me throughout my graduate school.

Lastly, I wish to acknowledge the love and support of my friends and family. Especially, my parents always encourage me and are with me through all my critical time.

## TABLE OF CONTENTS

CHAPTER 1	INTRODUCTION . . . . .	1
CHAPTER 2	ADVERSARIAL ATTACKS AND DEFENSES . . . . .	4
2.1	Legitimate vs. Adversarial Examples . . . . .	4
2.2	Attack Approaches . . . . .	4
2.3	Defense Approaches . . . . .	5
CHAPTER 3	APPROACH OF MULDEF . . . . .	7
3.1	Diversity Measurement . . . . .	8
3.2	General Defense Framework and Model-Generation Technique . . . . .	8
3.3	Empirical Exploration on Design Choices . . . . .	11
CHAPTER 4	EVALUATION . . . . .	14
4.1	Experiment Setup . . . . .	14
4.2	Results and Analysis . . . . .	16
CHAPTER 5	DISCUSSION . . . . .	27
CHAPTER 6	RELATED WORK . . . . .	28
6.1	Existing Defenses . . . . .	28
6.2	Improving Machine Learning Models . . . . .	29
CHAPTER 7	CONCLUSION AND FUTURE WORK . . . . .	30
REFERENCES	. . . . .	31

## CHAPTER 1: INTRODUCTION

Neural networks recently have been used to solve many real-world tasks such as image recognition and can achieve high effectiveness on these tasks [1]. However, given a legitimate input that the model can produce correct results, previous research [2, 3, 4, 5] proposed various attack approaches to perturb the input by applying imperceptible modification on the input to fool the model, i.e., causing the model to produce incorrect result for the perturbed input. We refer to such perturbed input an *adversarial example* and the model being attacked the *target model*. These attack approaches can be used in two attack scenarios: (1) a white-box attack scenario where the attackers have complete knowledge about the target model (and also its defense approaches), and (2) a black-box attack scenario where the attackers do not know anything about the target model (or its defense approaches), but know the output produced by the model, given an arbitrary example. We focus on improving the target model against the white-box attack scenario, which is known to be harder to defend against.

With various effective attack approaches being invented for the two attack scenarios, a number of defense approaches such as adversarial training [6, 2] and defensive distillation [7] were proposed. However, these existing approaches are not robust against various attacks, especially in the white-box attack scenario, facing three main limitations. (1) *Ineffectiveness against re-attack*. The improved target model resulted from some defense approaches such as adversarial training is still vulnerable to adversarial examples generated by reapplying the same attack approach on the improved model in the white-box attack scenario. These attack approaches rely on computing the model’s gradient. Even after the defense approach of adversarial training improves the model with additional adversarial examples in the training set, the attack approaches can still compute the gradient of the improved model and generate new adversarial examples. Our preliminary results (Figure 3.4a) also suggest that adversarial training alone is ineffective in the white-box attack scenario, where attackers have complete knowledge about the model. One potentially effective variation of adversarial training includes modification of the loss function used to optimize the model parameters [5], but that variation requires manually changing the target model’s implementation for different attacks. (2) *Ineffectiveness against transferable attack*. Adversarial examples have the *transferability* property: adversarial examples can be used to transfer attacks across models [8]. So attackers can train a substitute model (on which white-box attack can be applied) for the target model and generate adversarial examples for the substitute model to indirectly attack the target model [2]. This transferability property can also be used to attack the tar-

get model in a black-box attack scenario [9]. (3) *Bypassable distillation*. Even after defensive distillation, attackers can still compute the gradient of the inputs to the pre-softmax layer and reduce the magnitude of the inputs to the softmax layer [10].

To address these three main limitations, we propose a new defense approach, named MULDEF, based on robustness diversity. Our approach consists of (1) a general defense framework based on multiple models and (2) a technique for generating these models to achieve high defense capability. Our defense framework includes two components: the model generator and runtime model selector. The design of the general defense framework is based on the design principle of security through *diversity*. Such diversity among multiple models introduces uncertainty in the target model [11], making it harder to attack. We design our general framework based on our main insight that existing attack approaches attack a single-model machine learning system by computing the target model’s gradient based on its loss function.

In particular, the model generator constructs a model family to assure that an adversarial example generated for one model in the family cannot fool other models in the family. The model family addresses the limitations of ineffectiveness against re-attack and transferable attack. To address the limitation of bypassable distillation, the runtime model selector uses a low-cost random strategy to select a model in a model family to be applied on a given example such that the attackers do not know beforehand which model to compute the gradient even when distillation can be bypassed. Such random strategy increases difficulty for the reverse engineering efforts by attack approaches. Note that generally (deterministic) runtime analysis (e.g., multi-model majority voting [12, 13]) can be conducted on a given example and all models in the family to pick the model that is robust to the example. However, such runtime analysis is costly and unscalable.

In our general defense framework, the runtime model selector has to select a model (in the family of diverse models) that is robust to any given example with a high chance. Thus there are two main desirable properties of the model family. (1) *Legitimate-behavior preservation*. The models in the family shall preserve the same accuracy on legitimate examples as the target model. (2) *Robustness diversity*. Given an adversarial example, the majority of the models in the family are robust to the example; in this way, even when the adversarial example is carefully constructed to attack one model in the family successfully (e.g., in a white-box attack scenario), there is at most  $\frac{N-1}{N}$  chance for randomly selecting from the family another model robust to the adversarial example, where  $N$  is the family size (i.e., the number of the models in the family). We introduce a new metric named *diversity measurement* to quantify the robustness diversity of a model family.

In addition, while aiming to satisfy the preceding two properties,  $N$ , the number of models



generated for the family, shall aim to be as high as possible in order to increase the defense capability. The reason is that the chance for randomly selecting the model based on which the adversarial example is carefully constructed (and thus likely successful in attacking) is at least  $1/N$ . The larger the value  $N$  is, the higher the chance of selecting a robust model for a given example is. However, increasing  $N$  makes it more challenging to satisfy the preceding two properties. Even for large  $N$ , if many models are similar to the target model, the chance of selecting a robust model can still be low.

To demonstrate the benefits of our general framework, our approach includes a technique for generating these multiple models to achieve robustness diversity. In particular, for the first property (i.e., legitimate-behavior preservation), our technique constructs each additional model by using the same architecture and parameter configuration as the target model, and the majority of the training examples are from the original training set to train the target model. For the second property (i.e., robustness diversity), the models in the family should be diverse and complementary. To accomplish so, our technique trains later-constructed models with some adversarial examples for the earlier-constructed models.

In summary, this thesis makes the following main contributions:

- A general defense framework based on multiple models constructed from the target model.
- A novel technique for generating these multiple models to achieve high robustness diversity.
- Comprehensive evaluation on three attack approaches (Fast Gradient Sign Method [5], Carlini & Wagner attack of type  $L_2$  [2], Projected Gradient Descent [14]) for showing substantial benefits of our general framework instantiated with the technique for improving defense capability. Moreover, we include the analysis for strong adaptive attacks that are specifically designed to break our defense approach.

We choose the random strategy for the runtime model selector in the target model; however, there are other strategies, which we can explore in future work, such as using majority vote across all the models in the family. The rest of the thesis is organized as follows. Chapter 2 provides some background related to adversarial attacks and defenses. Chapter 3 explains our MULDEF approach in more detail. Chapter 4 discusses our experiments and evaluation. Chapter 5 includes more discussion related to our approach. Chapter 6 presents related work. Lastly, Chapter 7 concludes this thesis.

## CHAPTER 2: ADVERSARIAL ATTACKS AND DEFENSES

In this chapter, we illustrate the terminology and basic attacks and defense approaches in previous/related work.

### 2.1 LEGITIMATE VS. ADVERSARIAL EXAMPLES

We focus on addressing adversarial example attacks on neural network models for classification tasks. A *legitimate example*  $x$  is an example that occurs naturally [15] for the classification task. For example, if a classification task is to classify digits, legitimate examples can be images of real digits without other elements. An *adversarial example* [8]  $x'$  is an example similar to a legitimate example with imperceptible changes (of the legitimate example) that can change the target model's prediction on the example.

### 2.2 ATTACK APPROACHES

An attack approach takes a legitimate example and then tries to generate an adversarial example similar to the legitimate example. Thus in our experiment setup (Chapter 4), we control each attack configuration to generate only an adversarial example within a certain distance from the given legitimate example in order to avoid human effort to label the adversarial example. There are two types of adversarial attack: *targeted* and *untargeted* attack. An untargeted attack finds an adversarial example  $x'$  that changes the target model's prediction to any other class label. A targeted attack also has a specific class label and finds an adversarial examples  $x'$  that changes the target model's prediction to the specific class label. In this thesis, we consider only untargeted attacks, because our goal is to make the target model more robust. If an adversarial example can change the target model's prediction to any other class label, our defense fails to protect against that adversarial example.

We evaluate our proposed defense approach against major existing attack approaches as described below.

**Fast Gradient Sign Method (FGSM).** FGSM [6] generates adversarial examples iteratively. Let  $l$  be the ground-truth label of  $x$ . Let  $J(x, l)$  be the loss function of classifying  $x$  as label  $l$ . For each pixel, FGSM updates the pixel according to the sign of the gradient of the loss function at the pixel. Formally, FGSM iteratively modifies  $x$  as follows:

$$x' = x + \epsilon \cdot \text{sign}(\nabla_x J(x, l)) \quad (2.1)$$

**Carlini & Wagner attack (C&W).** C&W [2] generates adversarial examples with small perturbation  $\delta$  through the following optimization

$$\delta = \arg \min_{\delta'} D(x, x + \delta') + c \cdot f(x + \delta') \quad (2.2)$$

where  $D$  is a distance metric,  $c$  is a constant to balance constraints, and a logit-based objective function  $f(\cdot)$  is designed in such a way that  $f(x') \leq 0$  if and only if the classifier misclassifies  $x'$ , indicating that the attack succeeds. Such logic-based objective function enables C&W to generate adversarial examples robust against the defensive distillation [7].

**Projected Gradient Descent (PGD).** PGD [14] is similar to FGSM, but more powerful. PGD is an iterative variant of FGSM. In each iteration, PGD updates the example as follows:

$$x'_{i+1} = \pi\{FGSM(x'_i)\} \quad (2.3)$$

where  $\pi$  is a clip function to keep  $x'_{i+1}$  within a defined perturbation range.

FGSM is fast, simple, and can be powerful. C&W and PGD are state-of-the-art attack approaches. There are different threat levels of adversarial attack, categorized by the amount of attackers' knowledge and their goal [16]. In a white-box attack scenario, the attackers have complete knowledge of the target model. The attackers' goal is *misclassification*, altering the output classification to any class label different from the original one. A black-box attack scenario has the same goal, but less knowledge. The attackers have access to only the *oracle*. We focus on the white-box attack scenario; however, our evaluation also includes the black-box attack scenario.

## 2.3 DEFENSE APPROACHES

We next describe two main existing defense approaches proposed in previous work.

**Adversarial training.** The idea of adversarial training [6, 2] is to make the target model more general and have some exposure to adversarial examples. A straightforward way is to augment the training set by replacing some training samples with the corresponding adversarial examples generated by an attack approach [17]. Also, one can train the model using an adversarial objective function to improve the robustness and generality of a classifier [14]. Adversarial examples can also be crafted from pre-trained models (e.g., ones from Ensemble Adversarial Training [18]).

**Defensive distillation.** Some attack approaches rely on optimizing an objective function by computing the gradient of the target model. Thus it would be useful for a defender to

hide the gradient of the target model. Defensive distillation [7] trains a classifier to cause a rapid reduction of its gradient over an input, resulting in that an attacker can hardly perform an attack requiring computing gradients. Defensive distillation hides the gradient between the pre-softmax layer and softmax outputs by using distillation training.

Some other existing defense approaches address the problem in different ways. First, they detect whether an example is adversarial by training a model detector [19]. Second, they reform the detected adversarial example to a legitimate example [15] or classify the adversarial example by using a highly nonlinear model [20].

In general, existing defense approaches extend the target model in various ways. Some approaches improve the target model against adversarial examples by modifying the weights and parameters of the target model. Some approaches [15] do not modify the target model at all, but detect and modify adversarial examples before passing them to the target model. Our proposed MULDEF approach does not modify the target model or adversarial examples. We evaluate our MULDEF approach on the following two metrics.

- **Test accuracy (TestAcc):** the accuracy of MULDEF on legitimate examples in the test set.
- **Robustness accuracy:** the accuracy of MULDEF on adversarial examples generated by an attack approach. The label of an adversarial example is the label of the legitimate example used to generate that adversarial example. In other words, this metric is for untargeted attack.

### CHAPTER 3: APPROACH OF MULDEF

We design our MULDEF approach for improving a neural network model to invalidate the reverse engineering efforts made by attack approaches. The design of the MULDEF approach is based on the design principle of diversity. Diversity achieved by randomly selecting a model from a family of models introduces uncertainty in the target under defense.

The MULDEF approach includes a general defense framework as shown in Figure 3.1. The framework consists of two components: (1) the model generator and (2) the runtime model selector. The two components work together to improve the target model’s effectiveness on adversarial examples. As described in Chapter 1, the model generator aims to produce a model family for achieving the two properties: (1) legitimate-behavior preservation and (2) robustness diversity. In addition, we aim to have as many models as possible to increase the chance that the runtime model selector ends up selecting a robust model for any given example.

First, let us formulate the problem. Given a target model  $T$ , we construct a model family (consisting of  $T, M_1, M_2, \dots, M_p$ ) that achieves the following objectives:

1. The difference between the lowest test accuracy of the models in the family and the test accuracy of  $T$  as formulated below shall be minimized:

$$|\min\{\mathbf{TestAcc}(M_1), \dots, \mathbf{TestAcc}(M_p)\} - \mathbf{TestAcc}(T)| \quad (3.1)$$

2. For a given adversarial example  $x$ , the total number of models (in the family) successfully attacked by  $x$  shall be minimized.
3. The family size  $p$  shall be maximized.

By trying to achieve all three objectives together, we may confront a situation similar to the space-time tradeoff. When we generate many models (increasing  $p$ ), those models should be diverse enough such that the majority of them are not successfully attacked by the same adversarial example. Moreover, they cannot be too diverse as we need to achieve the legitimate-behavior preservation. Our idea exploration in solving this problem suggests that we may focus only on achieving the first two objectives and then we can keep increasing more models as long as we do not compromise (much) the first two objectives.

In the rest of this chapter, we first introduce a new metric, *diversity measurement*, for the second objective, to measure the majority-model robustness. Then we explain the two components in our general framework along with empirical exploration on design choices for our model-generation technique.

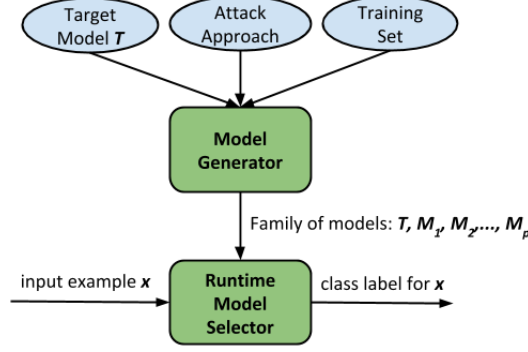


Figure 3.1: Our general defense framework. Model generator constructs  $p$  similar models to  $T$ :  $M_1, M_2, \dots, M_p$ . Given an input example  $x$ , MULDEF randomly selects one model to compute the class label for  $x$ .

### 3.1 DIVERSITY MEASUREMENT

We measure the diversity of the model family based on the robustness of majority of models against adversarial examples generated targeting on one of the models. The detailed steps are as follows: (1) generate adversarial examples for each model in the family; (2) for each generated example  $i$ , count the number of models that the example can successfully attack, denoted as  $s_i$ ; (3) plot the distribution of  $s_i$ , i.e., the number of models that each example successfully attacks. The diversity of the model family can be calculated as

$$1 - \frac{\sum_i s_i}{a \times n} \quad (3.2)$$

where  $a$  is the number of adversarial examples and  $n$  is the size of the model family. Informally if most examples successfully attack fewer models in the family (low  $s_i$ ), the model family is more diverse (close to 1).

### 3.2 GENERAL DEFENSE FRAMEWORK AND MODEL-GENERATION TECHNIQUE

According to Figure 3.1, given a target model, the model generator constructs a family of models. Models in the family (except the target model) are trained with additional adversarial examples from the specified attack approach. Note that the specified attack approach used to train the adversarial examples do not need to be the same as the attack approach that we are defending against.

### 3.2.1 Model-Generation Technique

To generate other additional models in the family, we initially start with the simple idea of adversarial training [6, 2]. In particular, given a target model  $T$ , we initially construct only one additional model  $M_1$  that has the same architecture and parameter setting as  $T$ . However,  $M_1$  is trained with an augmented dataset (the training set plus some adversarial examples for  $T$ ). Note that the adversarial examples are generated in the white-box attack scenario. Let  $Adv_{M_i}$  denote the set of adversarial examples for  $M_i$  and  $Adv_T$  denote the set of adversarial examples (generated by the specified attack approach in the white-box attack scenario) for the target model  $T$ . We notice that  $M_1$  performs better on the adversarial examples for  $T$  ( $Adv_T$ ), but  $M_1$  still performs worse on its own adversarial examples ( $Adv_{M_1}$ ). In fact, augmenting the training set cannot really improve the model against adversarial attacks. Next, we construct more models,  $M_1, M_2, M_3, \dots, M_p$ , where  $p$  denotes the number of additional models. The next question is how to train  $M_2, M_3, \dots, M_p$ .

For our model-generation technique, we devise the following two mechanisms to create the training set for each model:

- **Solution 1.** The training set of  $M_i$  is constructed as the union of the original training set and the adversarial examples generated for the previously constructed model  $Adv_{M_{i-1}}$ .
- **Solution 2.** The training set of  $M_i$  is constructed as the union of the original training set and the adversarial examples generated for each of all the previously constructed models  $Adv_T, Adv_{M_1}, Adv_{M_2}, \dots, Adv_{M_{i-1}}$ .

To compare the two solutions, we measure the diversity of two families of models (with the family size chosen as 5), each of which is constructed by using the two proposed solutions. We find that a model family constructed by **Solution 2** performs better than that constructed by **Solution 1**. Figure 3.2 shows the diversity measurement of the two model families. In **Solution 1**, the majority of the adversarial examples successfully attack 2 to 3 models. However, the majority of the adversarial examples successfully attack at most one model in **Solution 2**. Our experimental results also confirm that the model family constructed by **Solution 2** has a higher accuracy.

One observation to explain such results is that in the first solution,  $Adv_{M_2}$  may not be representative for  $Adv_{M_1}$ . So  $M_3$  (trained with the union of the original training set and  $Adv_{M_2}$ ) can still be vulnerable to  $Adv_{M_1}$ . In the second solution, we train  $M_3$  with the union of the original training set and  $Adv_T, Adv_{M_1}, Adv_{M_2}$  to make  $M_3$  more robust against all the previously constructed models' adversarial examples. Thus, we implement MULDEF

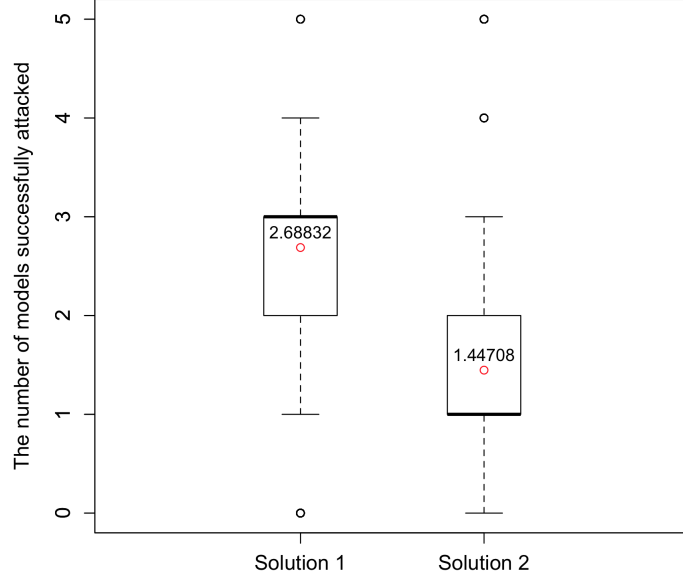


Figure 3.2: Diversity measurement of two model families constructed with **Solution 1** and **Solution 2**. Each red dot represents the mean.

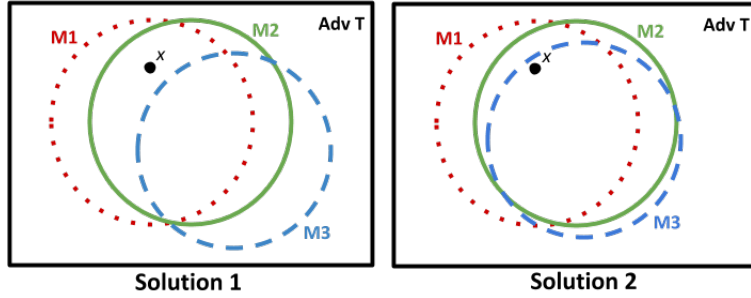


Figure 3.3: Comparison of the two solutions of augmenting the training set for constructing additional models.

by following the second solution. It is worth noting that the last model, which is trained with all the other models’ adversarial examples, seems to be more robust than other models. However, the reason that we still include other models in our defense is that the last model is vulnerable to its own adversarial set ( $Adv_{M_3}$ ), and all the models in the family could be complementary to each other.

Figure 3.3 illustrates the idea of having multiple models and why **Solution 2** performs better than **Solution 1**. Let the rectangle in each solution represent the set of all adversarial examples for the target model ( $Adv_T \subset \mathbb{S}$ ) under a given attack approach. This set can be infinite. MULDEF incrementally constructs additional models one by one. First, MULDEF constructs  $M_1$  aiming to defend against a subset of  $Adv_T$  by training  $M_1$  with some adversarial examples for  $T$ . The circle for  $M_1$  covers the subset of  $Adv_T$  that  $M_1$  can defend.



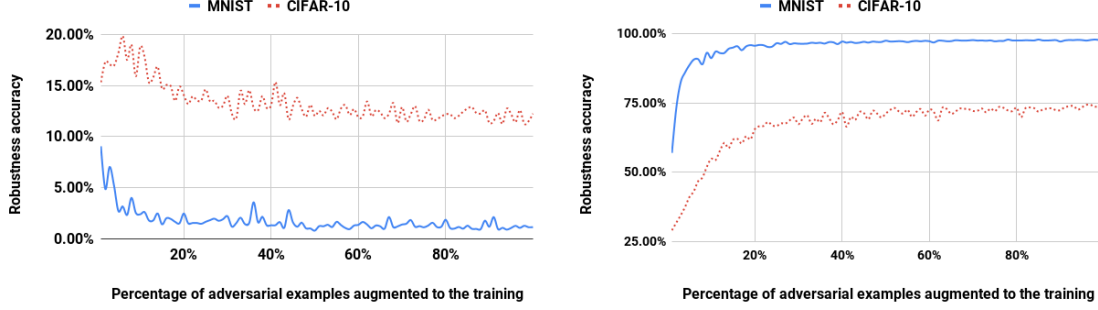
MULDEF keeps constructing more models to cover more space in the rectangle, indicating that MULDEF is getting more robust to adversarial examples. Intuitively, MULDEF performs well when many of the constructed models cover a large portion of the rectangle. The difference between **Solution 1** and **Solution 2** is that MULDEF trains  $M_3$  with the union of the original training set and  $Adv_T, Adv_{M_2}$  in **Solution 1**, but trains  $M_3$  with the union of the original training set and  $Adv_T, Adv_{M_1}, Adv_{M_2}$  in **Solution 2**. Thus,  $M_3$  in **Solution 2** is likely able to defend against adversarial examples for  $M_1$ , resulting in having a higher chance that a given adversarial example can be defended by all the three models ( $M_1, M_2, M_3$ ). According to Figure 3.3, a given adversarial example  $x$  can be defended by only  $M_1$  and  $M_2$  in **Solution 1**. However,  $x$  can be defended by all the three models in **Solution 2**. A higher number of additional models that are robust to  $x$  result in a higher chance that MULDEF selects a right model at runtime.

### 3.2.2 Runtime Model Selector

To combine multiple models together, MULDEF randomly selects a model from the family of models  $T, M_1, M_2, M_3, \dots, M_p$  to compute the class label for each given input example. The intuition of this strategy is to be able to introduce uncertainty in the target model (by the design principle of diversity) within the family of models so that it is hard for the attackers to generate adversarial examples that can attack all or most of the models. Moreover, this runtime model selector acts as a wall to hide the gradient of a single model, because the attackers do not know in advance which model MULDEF ends up selecting at runtime. This random strategy shares the same spirit of defensive distillation [7], which attempts to hide the gradient of the target model.

## 3.3 EMPIRICAL EXPLORATION ON DESIGN CHOICES

Before settling down on our approach, we also conduct some experiments to see whether using only adversarial training can make the target model more robust against adversarial examples. We create two convolutional neural network models that can achieve about 99.13%/80.4% test accuracy (on the test set) for both MNIST and CIFAR-10. Then we use FGSM to attack the model in the white-box attack scenario. Without adversarial training, the model has about 8.87%/13.79% robustness accuracy against FGSM for MNIST/CIFAR-10. Then we try augmenting the training set with adversarial examples (generated by FGSM) to see whether the model is more robust. Note that we run the experiment three times and report the average accuracy. Figure 3.4a shows that augmenting adversarial examples for both



(a) Robustness accuracy of target model  $T$  (b) Robustness accuracy of model  $D$  (on when its training set is augmented with different quantities of adversarial examples. with different quantities of  $Adv'_T$ .

Figure 3.4: Results of using adversarial training against FGSM white-box attack for MNIST and CIFAR-10 datasets.

datasets can even worsen the model: the more adversarial examples we augment, the lower robustness accuracy the model achieves. According to Figure 3.4a, the robustness accuracy of target model  $T$  for MNIST/CIFAR-10 surprisingly goes down to be under 2.50%/13.00% when we augment the training set with more than 50% of the original training set’s size.

How about constructing another model  $D$  that is robust to the target model’s adversarial examples ( $Adv_T$ )? Thus, we try to construct a new model  $D$  with the same architecture as  $T$ , and train  $D$  with the original training set augmented with a different set of adversarial examples for  $T$  ( $Adv'_T$ ) to see how  $D$  performs on  $Adv_T$ . The results in Figure 3.4b show that the robustness accuracy of model  $D$  on  $Adv_T$  is higher when we augment more adversarial examples especially in the beginning for both datasets. Then the robustness accuracy converges to around 97%/74% for MNIST/CIFAR-10. Notice that the robustness accuracy does not significantly change when we augment more than around 15-20% of adversarial examples for MNIST and CIFAR-10. So we decide that in our MULDEF approach, we construct other models by using 15% of adversarial examples to augment each training set, because having too many adversarial examples in the training set can decrease the test accuracy. For example, if we use the augment rate of 20%, the test accuracy of the fifth model in the family goes down by 4% for CIFAR-10. There is a tradeoff between robustness accuracy and test accuracy.

One may wonder why we decide to augment the training set with adversarial examples. We can also use only adversarial examples for  $T$  to train other additional models. If we use only adversarial examples to train other additional models, the additional models will perform worse on legitimate examples (in the test set), not being desirable.

According to the preceding observation, simply retraining the target model with adver-

sarial examples cannot significantly make the model more robust against future adversarial examples as the attack approach also knows everything about the retrained target model. However, having multiple models helps as an adversarial example for one model may not be able to fool another model. The runtime model selector helps combine multiple models together. We use the random selection strategy because it is simple, low-cost, and provides some probabilistic guarantee that MULDEF will not likely select a model that is vulnerable to a given adversarial example when we have many models. Our evaluation includes experiments to investigate on how many additional models our model-generation technique can generate while achieving legitimate-behavior preservation and robust diversity.

## CHAPTER 4: EVALUATION

In this chapter, we evaluate the effectiveness of our MULDEF defense approach in various settings. In addition, we compare MULDEF to other existing defense approaches, including other analysis on how our approach works.

### 4.1 EXPERIMENT SETUP

We discuss the datasets and models used in our evaluation. We run all the evaluations three times and report the average accuracy to reduce chance of accidental observations.

#### 4.1.1 Datasets

We evaluate our approach on three popular public datasets as listed below.

- **MNIST (MNIST)** is a dataset of handwritten digits, consisting of ten labels for the ten digits. We select 60,000 examples for the training set and 10,000 examples for the test set. Each image is a  $28 \times 28$  black and white image.
- **CIFAR-10 (CIFAR)** is a widely used dataset consisting of 10 labels. We select 50,000 examples for the training set and 10,000 examples for the test set. So there are 6,000 images per class. Each image is a  $32 \times 32$  color image.
- **Tiny ImageNet (ImageNet)** is part of Stanford University’s CS231n course and the tiny ImageNet challenge [21]. The dataset contains 100,000 images across 200 classes. Each class has 50 test images. Tiny ImageNet is a strict subset of ILSVRC2014 (the actual ImageNet dataset). All images have a higher resolution of  $64 \times 64$  than the ones in the first two datasets.

#### 4.1.2 Target Model

We use three different convolutional neural network models for the three datasets (MNIST, CIFAR, and ImageNet) as listed below.

- **ModelA:** We follow the previous study on FGSM [5] and use the same model in the study to evaluate FGSM with MNIST. The model mainly consists of three convolutional layers with 64 neurons for each layer and ReLU as the activation function, and a densely-connected layer of 10 neurons for each digit.

Table 4.1: Test accuracy of the model used for each dataset.

Dataset	Test accuracy
MNIST(A)	99%
CIFAR(B)	77%
CIFAR(RES)	84%
IMAGENET(RES)	48%

- **ModelB**: We slightly adjust the model for CIFAR-10 in the study by C&W [2]. The model mainly consists of four convolutional layers with 64 neurons for the first two layers, 128 neurons for the next two layers, and ReLU as their activation function, two densely-connected layers of 256 neurons, and a densely-connected layer of 10 neurons for each class. The only difference is that we add dropouts in both the convolutional layers and densely-connected layers. We also add  $L_2$ -norm regularization in the first two densely-connected layers.
- **ResNet**: We use ResNet18, proposed in the previous study [1], to evaluate the performance of MULDEF on a larger model. ResNet18 consists of 18 different convolution layers. We also apply a technique of data augmentation while training the model to improve its accuracy as suggested in the study.

For **MNIST**, we apply **ModelA** denoted as MNIST(A). For **CIFAR-10**, we apply **ModelA** and **ResNet** denoted as CIFAR(B) and CIFAR(RES), respectively. For **ImageNet**, we apply **ResNet** denoted as IMAGENET(RES).

We set the max epoch equal to 50 for the MNIST dataset and 100 for the CIFAR-10 dataset. The training processes adopt the early stopping technique used to avoid overfitting. The technique stops the training process when the validation loss fails to reduce by at least 0.001 for 5 epochs. Table 4.1 shows the test accuracy of each model that we use. Note that the Tiny ImageNet is more difficult to classify as it has 200 classes, and the training set contains only about 500 examples per class.

#### 4.1.3 Attack Approaches

To check whether a model outputs a correct label for an adversarial example generated by an attack approach, we need to set the parameters of the attack approach to constrain the amount of perturbation on legitimate examples, so that we can use the original label as the ground truth.

In FGSM, the degree of perturbation is controlled by the parameter  $\epsilon$ . We set  $\epsilon$  to be 0.3/0.05/0.05 for MNIST/CIFAR-10/ImageNet.

In C&W, we use C&W attack of type  $L_2$ , and the degree of perturbation is controlled by the parameter *confidence*. We set this value to 0.01 for all datasets. Another parameter named *max\_iterations* is used to control the max number of iterations for generating adversarial examples. We set *max\_iterations* to 1000 for all datasets. This value is high enough to generate adversarial examples for reducing the target model’s accuracy to 0%. Other parameters in C&W are set as their default values.

In PGD, we set the parameter *eps* the same as in the FGSM experiments, because the PGD approach is built on top of FGSM. We reduce *eps\_iter* to 0.01 from its default value 0.05 for CIFAR-10 and ImageNet, because we use lower perturbation ( $eps = 0.05$ ), but increase *nb\_iter* to 30 from its default value 10.

#### 4.1.4 MULDEF Setup

There are two main parameters to configure MULDEF: (1) the percentage of adversarial examples augmented to the training set, and (2) the number of additional models to be constructed. As discussed in Chapter 3, we select 15% as the percentage of augmented adversarial examples. To explore an optimal number of additional models, we set the number of additional models to be 1, 2, 3, and 4. Thus in total, MULDEF has at most 5 models including the target model.

## 4.2 RESULTS AND ANALYSIS

We measure the effectiveness of MULDEF for (increasing the robustness of) the models on the three datasets against FGSM, C&W, and PGD.

### 4.2.1 Effectiveness of MULDEF in White-box Attack Scenario

To evaluate our approach in the white-box attack scenario, we perform two types of attacks to MULDEF:

1. **Non-adaptive Attack.** Because MULDEF constructs a family of models:  $T, M_1, M_2, \dots$ , we can try the three attacks (FGSM, C&W, and PGD) in a divide-and-conquer fashion by attacking each model separately to come up with the strongest adversarial examples for MULDEF.
2. **Adaptive Attack.** Defending against non-adaptive attacks is necessary but not sufficient. Thus, we introduce two adaptive attacks built on top of FGSM and PGD.

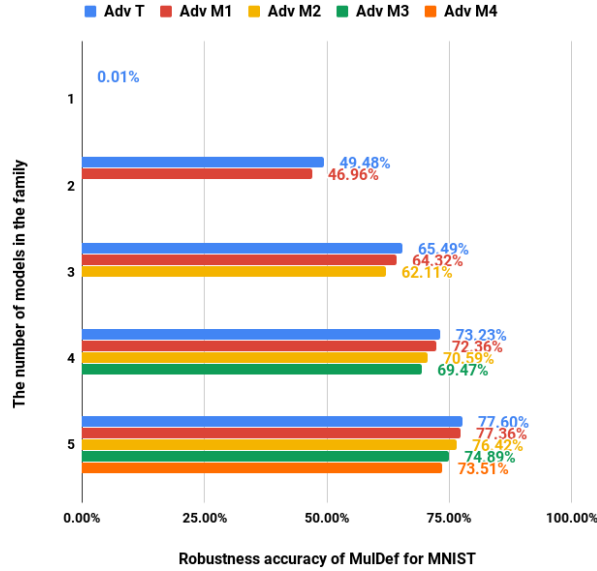


Figure 4.1: Robustness accuracy of MULDEF for MNIST(A) against PGD on different sets of adversarial examples. The figure contains five different settings of MULDEF with different sizes of model family.

**Effectiveness of MulDef under Non-adaptive Attack.** For non-adaptive attack, we generate a set of adversarial examples by using the existing attacks for each model in the model family to find the strongest adversarial set (the one with the highest attack success rate). We measure the robustness accuracies of MULDEF along with each model in the model family against all sets of adversarial examples. The robustness accuracy of MULDEF that we report is the lowest one (which is based on the strongest possible adversary) among all models. The median adversarial  $L_\infty$  distance from FGSM and PGD attacks on MNIST/CIFAR/ImageNet is 0.3/0.05/0.05. The median adversarial  $L_2$  distances from C&W attack on MNIST/CIFAR is 1.829/0.1823, respectively.

Figures 4.1, 4.2, 4.3, and 4.4 show the robustness accuracy of the target model  $T$  (MULDEF with only one model – the target model) and the robustness accuracy of MULDEF (with 2, 3, 4, and 5 models) on different sets of adversarial examples ( $Adv_T, Adv_{M_1}, \dots, Adv_{M_4}$  denoted as different colored bars). Due to space limit, we show the figures of results only against PGD (the state-of-the-art attack). The figures show that the more models MULDEF has in the family, the higher robustness accuracy MULDEF gains.

Table 4.2 summarizes the performance of MULDEF against different attacks. These results indicate that MULDEF can successfully defend **Non-adaptive Attack**. For the case of PGD/IMAGENET(RES), MULDEF achieves only 26.80%, which looks relatively low. One

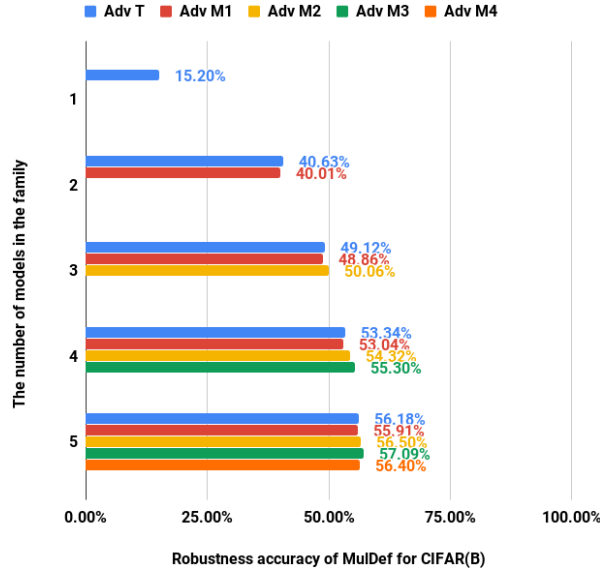


Figure 4.2: Robustness accuracy of MULDEF for CIFAR(B) against PGD on different sets of adversarial examples. The figure contains five different settings of MULDEF with different sizes of model family.

reason is that the Tiny ImageNet dataset contains very few examples (500) per class, and our **ResNet** model achieves only around 49% test accuracy.

**Effectiveness of MulDef under Adaptive Attack.** To extensively evaluate our MULDEF approach, we develop two adaptive attacks built upon FGSM and PGD. These two adaptive attacks are designed specifically to attack our approach. Because our approach randomly selects a model to predict for a given example, we can improve FGSM and PGD (which are made for attacking a single model) to consider all the models in the model family during their generation of adversarial examples. Instead of maximizing the loss function of a single model, we maximize the sum of all the loss functions as follows:

$$x' = x + \epsilon \cdot \text{sign}(\nabla_x \sum_i J_i(x, l)) \quad (4.1)$$

where  $J_i$  is the loss function of each model  $i$  in the model family. We conduct experiments on our MULDEF approach with 5 models for our target models. We also test our adaptive attack when the size of the perturbation is unbounded (setting the  $\epsilon$  to be 1.0 for MNIST), and the adaptive attack built upon PGD can reach about 97% attack success rate (i.e., MULDEF has only 3% accuracy). Thus, this adaptive attack is sufficiently effective. Table 4.4 shows the robustness accuracy of MULDEF against adaptive FGSM and adaptive PGD. We can



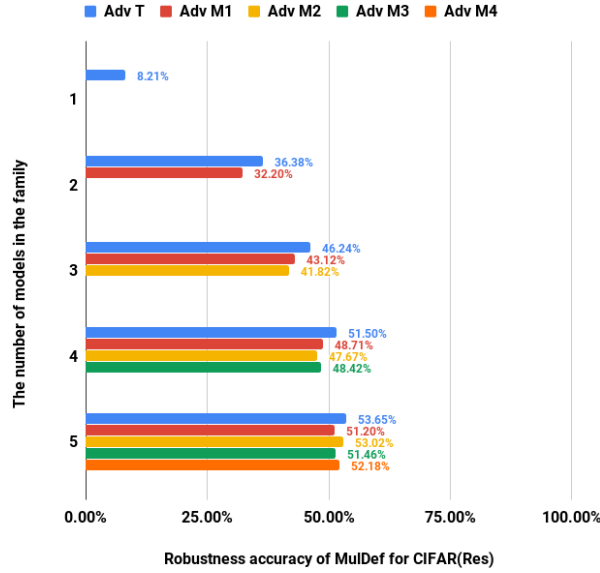


Figure 4.3: Robustness accuracy of MULDEF for CIFAR(Res) against PGD on different sets of adversarial examples. The figure contains five different settings of MULDEF with different sizes of model family.

see that MULDEF has relatively low robustness accuracy against **Adaptive Attack**. To increase the performance, we need to increase the size of the model family.

We also apply MULDEF on a baseline attack approach presented in previous work on gradient obfuscation attack [22], which is used to attack stochastic gradients in randomized defenses. Instead of maximizing the sum of all the loss functions, we maximize the sum of sampled loss functions (multiple randomly chosen loss functions). Against this baseline attack approach, MULDEF achieves higher accuracy against **Gradient Obfuscation**, denoted as the last column in Table 4.4. For all the adaptive attacks (including Gradient Obfuscation), the median adversarial  $L_\infty$  distance for MNIST/CIFAR-10 is 0.3/0.05.

#### 4.2.2 Comparison with Other Existing Defense Baselines

We also compare the performance of MULDEF with four existing defense baselines. (1) **Mean Blur** [23], proposed by Li et al., applies a  $3 \times 3$  average filter to blur examples/images before training or applying a model. (2) **Feature Squeezing** [24], proposed by Xu et al., is used to detect an adversarial example by “squeezing” out unnecessary input features. Median smoothing is an effective squeezer at removing noise in images and outstanding in their study of the MNIST, CIFAR-10, and ImageNet datasets. Median smoothing applies a

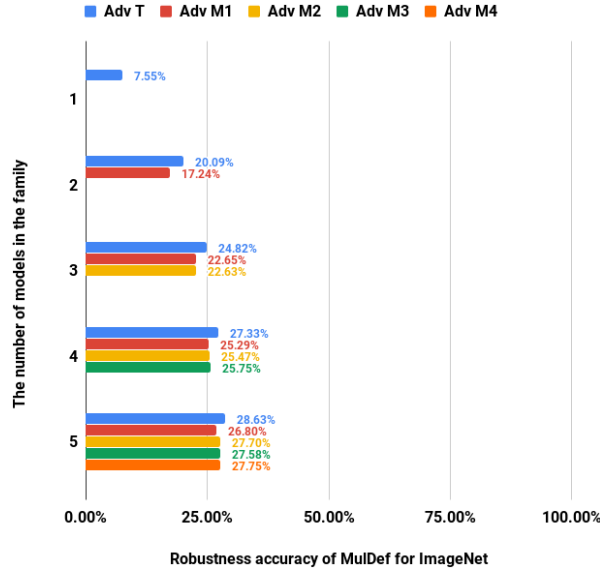


Figure 4.4: Robustness accuracy of MULDEF for IMAGENET(RES) against PGD on different sets of adversarial examples. The figure contains five different settings of MULDEF with different sizes of model family.

median filter with  $2 \times 2$  window size. (3) **PGD Training** [14], proposed by Madry et al., iteratively adversarially trains the model with PGD adversarial examples. (4) **Adaptive Diversity Promoting Training (ADP Training)** [25], proposed by Pang et al., is similar to our approach of using model ensemble. However, they introduce an ADP regularizer, which encourages the diversity among the ensemble, and then they combine this regularizer in the objective function to train all the models simultaneously. Their ADP training is not scalable for a high number of models and cannot handle large perturbation, so we use lower perturbation values and use only 3 models as in their evaluation. Tables 4.2 and 4.3 show that MULDEF outperforms these defense baselines in most of the cases under the same setting. Note that some results are missing for PGD Training, because some implementations are not publicly available and we cannot directly compare the results due to different model architectures and settings being used.

In addition, Na et al. [26] propose another kind of adversarial training, *cascade adversarial training*, which transfers the knowledge of the end results of adversarial training. Their approach performs better for MNIST as their approach can reach the robustness accuracy of 81–97%. Our approach performs better for CIFAR-10 as their approach reaches only the accuracy of 27–38%. We suspect that for complex images such as CIFAR-10 and ImageNet, our approach can outperform the existing defense approaches.

Table 4.2: Robustness accuracy comparison with other defense baselines.

Attack / Dataset	Robustness accuracy				
	No Defense	MulDef	Mean Blur	Feature Squeezing	PGD Training
FGSM/MNIST(A)	7.66%	<b>67.98%</b>	8.80%	14.37%	-
FGSM/CIFAR(B)	15.18%	<b>49.99%</b>	27.45%	21.77%	-
FGSM/CIFAR(RES)	9.09%	<b>50.79%</b>	26.02%	22.96%	-
FGSM/IMAGENET(RES)	6.34%	<b>24.86%</b>	8.82%	6.81%	-
C&W/MNIST(A)	0.00%	78.61%	<b>96.15%</b>	95.52%	-
C&W/CIFAR(B)	0.00%	<b>60.52%</b>	48.54%	59.95%	-
PGD/MNIST(A)	0.00%	73.51%	5.68%	2.13%	<b>93.54%</b>
PGD/CIFAR(B)	0.06%	<b>55.91%</b>	10.43%	4.00%	-
PGD/CIFAR(RES)	8.21%	<b>51.20%</b>	23.49%	28.89%	32.59%
PGD/IMAGENET(RES)	7.55%	<b>26.80%</b>	9.82%	9.95%	-

Table 4.3: Robustness accuracy comparison with ADP Training

Attack / Dataset	Para.	Robustness accuracy		
		MulDef (5 models)	MulDef (3 models)	ADP Training (3 models)
FGSM/MNIST(A)	$\epsilon = 0.2$	<b>72.4%</b>	68.6%	52.8%
FGSM/CIFAR(RES)	$\epsilon = 0.04$	<b>46.7%</b>	46.3%	46.2%
C&W/MNIST(A)	$\epsilon = 1.0$	<b>78.5%</b>	66.2%	78.1%
C&W/MNIST(A)	$\epsilon = 10.0$	<b>81.1%</b>	71.8%	23.8%
C&W/CIFAR(B)	$\epsilon = 0.01$	<b>60.5%</b>	50.6%	54.9%
PGD/MNIST(A)	$\epsilon = 0.15$	<b>78.6%</b>	66.4%	41.0%
PGD/CIFAR(RES)	$\epsilon = 0.02$	<b>59.7%</b>	49.2%	30.4%

#### 4.2.3 Cross-Attack Scenario

In the white-box attack scenario, we construct models in MULDEF to defend against one attack approach by using adversarial examples generated based on the same attack approach. We see that MULDEF performs very well. Thus, is MULDEF attack-dependent? Realistically, we cannot know in advance which white-box attack approach the attackers will use in reality, indicating that we cannot construct models in MULDEF based on the adversarial examples generated by the approach to be used by the attackers. Therefore, we further investigate the robustness of our approach in the cross-attack scenario, in which we launch one attack approach (e.g., C&W) to test MULDEF built on top of adversarial examples of a different attack approach (e.g., FGSM).

Table 4.5 shows that no matter which group of adversarial examples MULDEF uses to construct models, MULDEF performs better than the target model. Surprisingly, MULDEF built on FGSM adversarial examples is often more robust than it built on C&W adversarial examples. For example, against C&W attack, MULDEF built on FGSM adversarial exam-

Table 4.4: Comparison of the robustness accuracy of MULDEF against **Non-adaptive attack**, **Adaptive Attack**, and **Gradient Obfuscation** for different datasets.

Dataset	Non-adaptive		Adaptive		Gradient Obfuscation
	FGSM	PGD	FGSM	PGD	
MNIST(A)	67.98%	73.51%	65.68%	22.40%	48.31%
CIFAR(B)	49.99%	55.91%	34.80%	22.69%	35.30%
CIFAR(RES)	50.79%	51.20%	38.95%	24.53%	27.30%
IMAGENET(RES)	24.86%	26.80%	19.18%	13.13%	13.81%

Table 4.5: Robustness accuracies of MULDEF built on FGSM and C&W adversarial examples against FGSM and C&W white-box attacks.

Attack / Dataset	Tgt model	MulDef	
		FGSM adv exps	C&W adv exps
<b>FGSM/MNIST(A)</b>	7.66%	<b>67.98%</b>	14.96%
<b>FGSM/CIFAR(B)</b>	15.18%	<b>49.99%</b>	26.47%
<b>C&amp;W/MNIST(A)</b>	00.00%	70.52%	<b>78.61%</b>
<b>C&amp;W/CIFAR(B)</b>	00.00%	58.88%	<b>60.52%</b>

ples does not lose much robustness accuracy. However, the robustness accuracy of MULDEF built on C&W adversarial examples drops against FGSM attack. Note that it does not necessarily suggest that FGSM attack is always more powerful than C&W but instead suggests that FGSM adversarial examples are more suitable than C&W ones to construct diversified models in MULDEF.

We further measure the diversity of model families in MULDEF (defined in Chapter 3) **built on FGSM adversarial examples** for both datasets. Thus, we create two model families (of 5 models) built on FGSM attack for MNIST and CIFAR-10. In diversity measurement, we generate adversarial examples with both FGSM and C&W against the two models that we create. Figure 4.5 shows the distribution of adversarial examples by the number of models that each example successfully attacks. For example, on average, FGSM adversarial examples can attack 1.4463 (out of 5) models in the model family for MNIST. We observe that although previous results suggest that MULDEF built on FGSM examples is more effective, the diversity of the model family for CIFAR-10 (CIFAR-10/FGSM and CIFAR-10/C&W) is not very high ( $1 - \frac{2.47}{5} = 0.51$  and  $1 - \frac{3.06}{5} = 0.38$ ). Against C&W, the average number of models that can be attacked in the model family for CIFAR-10 is 3.06 out of 5. One possible reason is that the data in CIFAR-10 have high dimensions (RGB images with a larger number of pixels). Small changes in each dimension would enable the adversarial examples to be adapted to the diverse models. Model families trained for MNIST demonstrate more robustness diversity than model families trained for CIFAR-10.

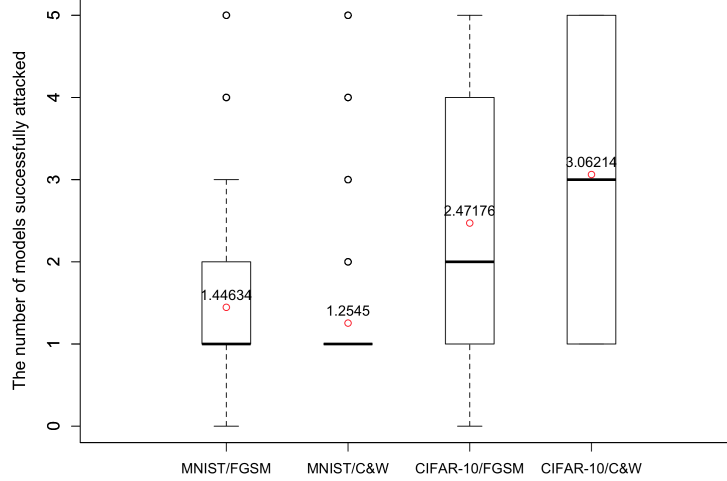


Figure 4.5: Comparison of the diversity measurement of two model families (built on FGSM adversarial examples on MNIST and CIFAR-10). The two boxplots on the left represent the diversity of the first model family on MNIST against FGSM and C&W attacks. The two boxplots on the right represent the diversity of the second model family for CIFAR-10 against the two attacks. Each red dot represents the mean.

#### 4.2.4 Effectiveness of MULDEF on Original Test Dataset

While achieving higher robustness accuracy, MULDEF also maintains about the same test accuracy as the target model. Table 4.6 shows that in each model family, the additional models have about the same test accuracy as the target model. The second column under MULDEF indicates the test accuracy of our defense, MULDEF.

In summary, MULDEF for all the models loses less than 3% test accuracy. From the experiments, we notice that MULDEF’s test accuracy tends to go lower for a larger target model or dataset. For example, model  $M_4$  of MULDEF for IMAGENET(RES) loses about 3% of its test accuracy. One reason can be that the Tiny ImageNet has a lot more classes (200 classes) and each class has only 500 examples, even the target model (without any defense) achieves only 48% test accuracy. Augmenting more adversarial examples can quickly mislead the model. To decrease the percentage loss in test accuracy, we suggest to reduce the augmentation rate.

#### 4.2.5 Impact of Different Numbers of Models

Figures 4.1, 4.2, 4.3, and 4.4 show that having more models in MULDEF can increase robustness accuracies. Surprisingly, each model can maintain about the same test accuracy as the target model’s. One may have the concern that when we generate more models,

Table 4.6: Test accuracy of each model in MULDEF when different attacks are used for different datasets. MULDEF contains five models:  $T$  (target model),  $M_1, M_2, M_3, M_4$ .

Attack / Dataset	MulDef	Test acc of each model (%)				
		$T$	$M_1$	$M_2$	$M_3$	$M_4$
<b>FGSM</b> /MNIST(A)	98.91	99.03	98.65	98.90	98.91	98.93
<b>FGSM</b> /CIFAR(B)	77.13	79.78	76.58	76.89	76.46	76.69
<b>FGSM</b> /CIFAR(RES)	82.14	84.09	82.77	82.35	81.30	81.06
<b>FGSM</b> /IMAGENET(RES)	45.63	48.23	46.89	45.93	44.41	43.93
<b>C&amp;W</b> /MNIST(A)	98.79	99.03	99.10	99.04	99.04	99.05
<b>C&amp;W</b> /CIFAR(B)	76.19	79.78	76.64	76.29	75.34	75.83
<b>PGD</b> /MNIST(A)	99.03	99.03	99.10	98.99	99.01	98.97
<b>PGD</b> /CIFAR(B)	73.18	76.78	73.90	72.41	72.72	71.50
<b>PGD</b> /CIFAR(RES)	82.25	84.09	82.37	81.61	81.13	81.06
<b>PGD</b> /IMAGENET(RES)	45.52	48.23	47.20	45.63	45.16	45.01

the last model may have lower test accuracy than the others, because the last model is constructed by the most adversarial examples augmented to its training set. It turns out that those adversarial examples do not have a negative impact or confuse the model when the model faces against legitimate examples. This result suggests that having more models could make the classifier more robust against adversarial examples.

Statistically, having more models increases the chance of selecting a robust model for each given input example at runtime. This assumption is true only when additional models are complementary to the existing models. In other words, the diversity of the model family is increasing. Thus we additionally evaluate our approach when using more models. Figure 4.6 shows the white-box robustness accuracy of MULDEF. The robustness accuracies converge when using 7–10 models. The results indicate that at some point, adding more models does not increase robustness diversity. Ideally, it is better to have more models to improve robustness diversity. So these results open up more future research directions on how to improve the model generator to be more effective in generating complementary models.

#### 4.2.6 Impact of Adversarial Training and Randomization in MULDEF

Based on the idea of adversarial training, the model generator in MULDEF constructs each additional model one after another, where  $M_i$  is trained with the union of the original training set and  $Adv_T, Adv_{M_1}, Adv_{M_2}, \dots, Adv_{M_{i-1}}$ . In other words, for  $Adv_{M_i}$ , all the other models constructed after  $M_i$  are trained with the training set that includes  $Adv_{M_i}$ . Therefore, most models constructed after  $M_i$  should be robust to  $Adv_{M_i}$ . To test this hypothesis, we measure the average robustness accuracy (on  $Adv_{M_i}$ ) of all the models constructed after  $M_i$ , denoted as the third accuracy in Table 4.7. We can see that the third accuracy is often the

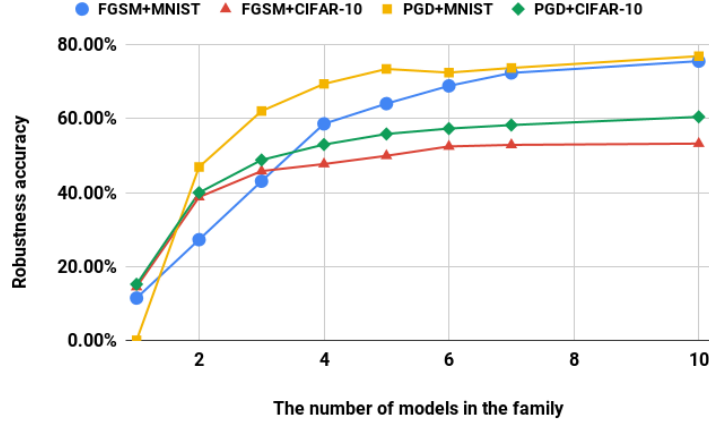


Figure 4.6: Robustness accuracy of MULDEF with different numbers of models based on **Non-adaptive Attack**.

Table 4.7: Three robustness accuracies on a set of adversarial examples generated for each model in the family of 5 models ( $T, M_1, M_2, M_3, M_4$ ). For the set of adversarial examples for model  $x$  ( $Adv_x$ ), (1) the **first** accuracy denotes the accuracy of model  $x$  on  $Adv_x$ , (2) the **second** accuracy denotes the average accuracy of all the models constructed (by MULDEF) **before** model  $x$  on  $Adv_x$ , and (3) the **third** accuracy denotes the average accuracy of all the models constructed (by MULDEF) **after** model  $x$  on  $Adv_x$ . Note that MULDEF constructs  $M_1, M_2, M_3$ , and  $M_4$  in order.

Attack / Dataset	Robustness accuracies on adversarial examples													
	$Adv_T$ (%)		$Adv_{M_1}$ (%)		$Adv_{M_2}$ (%)		$Adv_{M_3}$ (%)		$Adv_{M_4}$ (%)					
FGSM/MNIST(A)	07.08	- / <b>95.28</b>	01.70	68.22	<b>95.50</b>	02.10	79.48	<b>80.31</b>	02.94	80.65	<b>96.55</b>	03.85	<b>84.83</b>	-
FGSM/CIFAR(B)	14.47	- / <b>61.08</b>	14.62	57.10	<b>65.29</b>	13.71	63.01	<b>68.43</b>	19.27	59.75	<b>68.49</b>	20.85	<b>57.46</b>	-
FGSM/CIFAR(RES)	09.09	- / <b>64.56</b>	10.38	55.44	<b>65.08</b>	11.16	57.75	<b>64.53</b>	11.42	60.10	<b>64.79</b>	12.31	<b>62.25</b>	-
FGSM/IMAGENET(RES)	06.34	- / <b>32.14</b>	08.24	23.26	<b>31.69</b>	08.97	28.27	<b>32.76</b>	09.38	30.36	<b>33.46</b>	09.76	<b>31.17</b>	-
C&W/MNIST(A)	00.00	- / <b>98.57</b>	00.00	98.18	<b>98.58</b>	00.00	98.32	<b>98.67</b>	00.00	98.03	<b>98.75</b>	00.00	<b>98.07</b>	-
C&W/CIFAR(B)	02.09	- / <b>74.58</b>	03.33	<b>75.47</b>	74.50	03.53	<b>75.72</b>	74.57	05.54	<b>75.32</b>	74.73	07.55	<b>74.94</b>	-
PGD/MNIST(A)	00.01	- / <b>97.03</b>	00.00	94.72	<b>97.46</b>	00.04	92.85	<b>97.16</b>	00.55	91.74	<b>96.54</b>	00.71	<b>91.64</b>	-
PGD/CIFAR(B)	15.20	- / <b>65.81</b>	15.06	65.37	<b>66.18</b>	17.92	65.93	<b>66.49</b>	18.71	66.96	<b>67.01</b>	16.63	<b>66.63</b>	-
PGD/CIFAR(RES)	08.21	- / <b>66.77</b>	09.47	54.92	<b>65.69</b>	09.70	61.21	<b>65.74</b>	10.05	61.21	<b>66.46</b>	10.14	<b>62.26</b>	-
PGD/IMAGENET(RES)	06.34	- / <b>32.14</b>	08.24	23.26	<b>31.69</b>	08.97	28.27	<b>32.76</b>	09.38	30.36	<b>33.46</b>	09.76	<b>31.17</b>	-

highest as expected. There are a few cases that the third accuracy is not the highest but is slightly lower than the second accuracy in C&W/CIFAR(B). We inspect these cases and find that the third, fourth, and fifth models are not more resilient than the second model in the family, indicating that the second model in the family already reaches the saturation point of robustness accuracy. For example, we would need to improve the model construction in our approach in order to reach higher robustness accuracy for C&W/CIFAR(B). Note that the cells in the last column in Table 4.7 are missing the third accuracy, because  $M_4$  is the last model in the family.

In order to select a robust model by the runtime model selector with a high chance, most models constructed before  $M_i$  should be robust to  $Adv_{M_i}$  as well. Thus, we measure the average accuracy of all the models constructed before  $M_i$  on  $Adv_{M_i}$ , denoted as the second

Table 4.8: Comparison of robustness accuracy of the target model and MULDEF in the black-box attack scenario.

Attack / Dataset	Black-box	
	Tgt model	MULDEF
<b>FGSM</b> /MNIST(A)	70.19%	<b>81.76%</b>
<b>FGSM</b> /CIFAR(B)	70.32%	<b>72.82%</b>
<b>FGSM</b> /CIFAR(RES)	68.24%	<b>82.15%</b>
<b>C&amp;W</b> /MNIST(A)	72.10%	<b>78.89%</b>
<b>C&amp;W</b> /CIFAR(B)	<b>70.71%</b>	65.28%

accuracy in Table 4.7. Our results show that the second accuracy is also relatively high compared to the accuracy of  $M_i$  on  $Adv_{M_i}$  (denoted as the first accuracy in the table). These results raise a question on why most models constructed before  $M_i$  can achieve higher accuracy on  $Adv_{M_i}$ , even though they are not exposed to  $Adv_{M_i}$  at all during their training process. One possible reason is that  $Adv_{M_i}$  is very specific to  $M_i$ , especially for C&W adversarial examples. And every new model constructed is trained with more adversarial examples, causing the later constructed models to be different from previously constructed models.

#### 4.2.7 Effectiveness of MULDEF in Black-box Attack Scenario

For the black-box attack scenario where the attackers can access only the target model output, we first train a substitute model with synthetic inputs selected by a Jacobian-based heuristic [3] to approximate the target model’s decision boundaries. We use 150 hold-out images from the test set and run 5 Jacobian-based augmentation epochs, and set the augmentation parameter  $\lambda = 0.1$ . All of these parameters are default values. Then we apply white-box attacks on the substitute model to generate adversarial examples and evaluate the target model on those examples.

Table 4.8 shows that MULDEF still achieves higher robustness accuracy than the target model in the black-box attack scenario except when C&W and CIFAR-10 are used. Reasons that our defense does not substantially improve the target model in the black-box attack scenario are the fact that black-box attack approaches that we use are not powerful enough to reveal weakness in the target model and MULDEF is built upon the white-box adversarial examples. We also need a better tuning of parameters or substitute model to improve the effectiveness of the black-box attack.



## CHAPTER 5: DISCUSSION

MULDEF incrementally constructs additional models one by one. According to Figure 4.6, we expect that the adversarial accuracy of MULDEF should keep increasing as it has more models, and then eventually stabilize. Thus, we can implement MULDEF in such a way that every time it constructs a new model, we measure the diversity of the model family. We stop constructing more models, if the average number of models successfully attacked is non-trivial.

In our evaluations, augmenting 15% of the training set with adversarial examples seems to be sufficient. However, for some other datasets or target models, we may need to augment more than 15%. Moreover, one needs to ensure that the test accuracy of each additional model still preserves. The step of choosing this percentage parameter may require tuning to achieve a satisfactory result.

In our experiments, we also try a number of different target models for MNIST. Most of them achieve about the same accuracy around 98–99%; however, we notice that some of the target models are more robust than the others, due to weakening the transferability property of the adversarial examples. We find that if we use a more robust target model, our defense system can achieve substantially higher robustness accuracy.

According to our evaluation results, MULDEF slightly improves the target model’s adversarial accuracy by about 2–10% in the black-box attack scenario. We suspect the reason why we do get much improvement to be that the adversarial examples augmented to the training set to construct additional models are generated in the white-box attack scenario; however, we evaluate MULDEF in the black-box attack scenario. Another reason is that the adversarial accuracy baseline (of the target model) is already high. We may get a better result if we use a better substitute model that makes the black-box attack more effective.

## CHAPTER 6: RELATED WORK

### 6.1 EXISTING DEFENSES

A variety of approaches have been proposed for defending against adversarial examples. Meng et al. [15] propose a defense approach named MagNet against adversarial examples. MagNet consists of two main steps: detect and reform. Like MULDEF, MagNet does not modify the target model. MagNet first detects whether a given input example is adversarial by measuring the distance between the input example and the manifold of legitimate examples in the training set. If the input example is farther to the manifold of the legitimate examples, the input example is marked as an adversarial example and then gets reformed/reconstructed to be close to legitimate examples. Finally, the example is passed to the target model to classify. Their ideas are quite different from our MULDEF approach, because MULDEF does not reform the given example. Instead, MULDEF trains more models to handle any example. MagNet does not require the knowledge of the attack approaches, and can perform well in a *gray-box* attack scenario (where the attackers know about the target model and the defense, but not the parameters of the defense), but cannot handle a white-box attack scenario at all. In fact, MagNet achieves only less than 20% robustness accuracy against FGSM attack (in the white-box attack scenario) and less than 40% against C&W on the MNIST dataset [27]. So our approach outperforms MagNet.

Other approaches [28, 29] similar to MagNet generate images similar to the given adversarial example by using Generative Adversarial Net (GAN) [30]. These approaches still provide ineffective defense against C&W.

Rouhani et al. [31] propose an approach of using complementary but disjoint modular redundancies to defend against adversarial examples. Although Rouhani et al. claim their approach to be attack-independent, it still requires a lot of pre-constructed modules, which are needed to be trained with different attack algorithms. Their approach only guarantees to detect whether a given input is a legitimate input or not, but there is no guarantee on the accuracy of the output when the input is adversarial.

Song et al. [32] propose an approach of image purification named PixelDefend to defend against adversarial examples. PixelDefend requires no knowledge of the attack approach nor the target model, but uses the PixelCNN model for its state-of-the-art performance in modeling image distributions [33] to detect an adversarial example. Then PixelDefend purifies the adversarial example by searching for more probable images within a small distance of the adversarial example. Because PixelDefend only purifies the given adversarial example,

we can combine this approach with MULDEF to proceed with the purified example.

Zantedeschi et al. [34] propose an approach to make the target model more robust against adversarial examples by reinforcing the model architecture so that its prediction becomes more stable. Their approach uses the bounded ReLU activation function for hedging against the forward propagation of adversarial perturbation and Gaussian data augmentation during training. Their approach is mainly for making the attack visually detectable. However, it still does not perform well against C&W.

## 6.2 IMPROVING MACHINE LEARNING MODELS

Our work falls into the general research domain of improving machine learning models. DeepXplore [35] and DeepTest [36] introduce and leverage the metric of neuron coverage for a neural network with rectified linear units (ReLU) as the activation functions. Neuron coverage measures the percentage of hidden units that can have positive value (for at least one of the test inputs) in the neural network. DeepGauge [37] further generalizes neuron coverage by dividing the range of values of each neuron (obtained during training) into  $k$  chunks, and measures whether each of the  $k$  chunks can be covered by the test cases. DeepGauge also measures whether each activation has been made to go above and below a certain bound. In addition to neuron coverage, a feature-guided approach [38] and concolic execution [39] approach are also proposed to perform black box testing of image classifiers using image-specific operations.

## CHAPTER 7: CONCLUSION AND FUTURE WORK

In this thesis, we have proposed MULDEF, a defense approach against adversarial examples for neural networks. Our approach consists of (1) a general defense framework based on multiple models and (2) a technique for generating these multiple models to achieve high defense capability. In particular, we construct a family of models such that the models are complementary to each other to accomplish robustness diversity. Our approach is simple, scalable, and easy to be applied, because it does not modify the target model. We evaluate our approach on three attack strategies (FGSM, C&W, and PGD) and some adaptive attacks for three datasets (MNIST, CIFAR-10, and Tiny ImageNet). The evaluation results show that our defense approach substantially improves the target model’s robustness accuracy by 19–78% against both attack strategies and three datasets, while still maintaining similar accuracy as the target model on legitimate examples.

We choose the random strategy for the runtime model selector to introduce uncertainty in the target model. Nevertheless, there can be another strategy based on multiple-model majority voting [35, 13], which we plan to explore in future work. In particular, our previous work [13] uses multiple-implementation testing to test an implementation of a machine learning algorithm, where the majority output across multiple implementations of the same algorithm is used as a test oracle. MULDEF also contains multiple models that are robust to a given adversarial example, so we may be able to use the majority label across multiple models as an output, instead of randomly selecting one model to be applied on the given example at runtime.

## REFERENCES

- [1] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. CVPR*, 2016, pp. 770–778.
- [2] D. W. Nicholas Carlini, “Towards evaluating the robustness of neural networks,” in *Proc. Security and Privacy (SP)*, 2017, pp. 39–57.
- [3] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, “Practical black-box attacks against machine learning,” in *Proc. ASIA CCS*, 2017, pp. 506–519.
- [4] N. Narodytska and S. P. Kasiviswanathan, “Simple black-box adversarial perturbations for deep networks,” in *Proc. CVPRW*, 2017, pp. 1310–1318.
- [5] N. Papernot, N. Carlini, I. Goodfellow, R. Feinman, F. Faghri, A. Matyasko, K. Hambarzumyan, Y.-L. Juang, A. Kurakin, R. Sheatsley, A. Garg, and Y.-C. Lin, “cleverhans v2.0.0: an adversarial machine learning library,” *arXiv preprint arXiv:1610.00768*, 2017.
- [6] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” *arXiv preprint arXiv:1412.6572*, 2014.
- [7] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, “Distillation as a defense to adversarial perturbations against deep neural networks,” in *Proc. SP*, 2016, pp. 582–597.
- [8] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” in *Proc. ICLR*, 2013.
- [9] N. Papernot, P. D. McDaniel, and I. J. Goodfellow, “Transferability in machine learning: from phenomena to black-box attacks using adversarial samples,” *CoRR*, vol. abs/1605.07277, 2016. [Online]. Available: <http://arxiv.org/abs/1605.07277>
- [10] N. Carlini and D. A. Wagner, “Defensive distillation is not robust to adversarial examples,” *CoRR*, vol. abs/1607.04311, 2016. [Online]. Available: <http://arxiv.org/abs/1607.04311>
- [11] P. Larsen, A. Homescu, S. Brunthaler, and M. Franz, “SoK: Automated software diversity,” in *Proc. SP*, 2014, pp. 276–291.
- [12] K. Pei, Y. Cao, J. Yang, and S. Jana, “DeepXplore: Automated whitebox testing of deep learning systems,” in *Proc. SOSP*, 2017, pp. 1–18.
- [13] S. Srisakaokul, Z. Wu, A. Astorga, O. Alebiosu, , and T. Xie, “Multiple-implementation testing of supervised learning software,” in *Proc. EDSMLS*, 2018.
- [14] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” in *Proc. ICLR*, 2018.

- [15] D. Meng and H. Chen, “MagNet: A two-pronged defense against adversarial examples,” in *Proc. CCS*, 2017, pp. 135–147.
- [16] N. Papernot, P. D. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, “The limitations of deep learning in adversarial settings,” in *Proc. EuroSP*, 2016, pp. 372–387.
- [17] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “Deepfool: a simple and accurate method to fool deep neural networks,” in *Proc. CVPR*, 2016, pp. 2574–2582.
- [18] F. Tramer, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, “Ensemble adversarial training: Attacks and defenses,” in *Proc. ICLR*, 2018. [Online]. Available: <https://openreview.net/forum?id=rkZvSe-RZ>
- [19] J. Lu, T. Issararanon, and D. A. Forsyth, “Feature-guided black-box safety testing of deep neural networks,” in *Proc. ICCV*, 2017, pp. 446–454.
- [20] D. Krotov and J. J. Hopfield, “Dense associative memory is robust to adversarial inputs,” *CoRR*, 2017. [Online]. Available: <http://arxiv.org/abs/1701.00939>
- [21] S. U. CS231N, “The Tiny ImageNet dataset,” 2017, <https://tiny-imagenet.herokuapp.com/>.
- [22] A. Athalye, N. Carlini, and D. Wagner, “Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples,” in *Proc. ICML*, 2018, pp. 274–283.
- [23] X. Li and F. Li, “Adversarial examples detection in deep networks with convolutional filter statistics,” in *Proc. ICCV*, 2017.
- [24] W. Xu, D. Evans, and Y. Qi, “Feature squeezing: Detecting adversarial examples in deep neural networks,” in *Proc. NDSS*, 2018.
- [25] T. Pang, K. Xu, C. Du, N. Chen, and J. Zhu, “Improving adversarial robustness via promoting ensemble diversity,” in *Proc. ICML*, 2019.
- [26] S. M. Taesik Na, Jong Hwan Ko, “Cascade adversarial machine learning regularization with a unified embedding,” in *Proc. ICLR*, 2018.
- [27] P. Samangouei, M. Kabkab, and R. Chellappa, “Defense-gan: Protecting classifiers against adversarial attacks using generative models,” in *Proc. ICLR*, 2018.
- [28] S. Shen, G. Jin, K. Gao, and Y. Zhang, “AE-GAN: adversarial eliminating with GAN,” *Proc. CoRR*, 2017.
- [29] G. Krishnan Santhanam and P. Grnarova, “Defending against adversarial attacks by leveraging an entire gan,” *ArXiv e-prints*, 2018.
- [30] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” in *Proc. NIPS*, 2014, pp. 2672–2680.

- [31] R. Bitar Darvish, S. Mohammad, J. Mojan, J. Tara, and K. Farinaz, “Deepfense: Online accelerated defense against adversarial deep learning,” in *Proc. ICCAD*, 2018.
- [32] Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman, “Pixeldefend: Leveraging generative models to understand and defend against adversarial examples,” in *Proc. ICLR*, 2018. [Online]. Available: <https://openreview.net/forum?id=rJUYGxbCW>
- [33] A. v. d. Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu, “Conditional image generation with PixelCNN decoders,” in *Proc. NIPS*, 2016. [Online]. Available: <http://dl.acm.org/citation.cfm?id=3157382.3157633> pp. 4797–4805.
- [34] V. Zantedeschi, M.-I. Nicolae, and A. Rawat, “Efficient defenses against adversarial attacks,” in *Proc. AISEc*, 2017.
- [35] K. Pei, Y. Cao, J. Yang, and S. Jana, “DeepXplore: Automated whitebox testing of deep learning systems,” in *Proc. SOSP*, 2017, pp. 1–18.
- [36] Y. Tian, K. Pei, S. Jana, and B. Ray, “Deeptest: Automated testing of deep-neural-network-driven autonomous cars,” in *Proc. ICSE*, 2018, pp. 303–314.
- [37] L. Ma, F. Juefei-Xu, J. Sun, C. Chen, T. Su, F. Zhang, M. Xue, B. Li, L. Li, Y. Liu et al., “Deepgauge: Comprehensive and multi-granularity testing criteria for gauging the robustness of deep learning systems,” *arXiv preprint arXiv:1803.07519*, 2018.
- [38] M. Wicker, X. Huang, and M. Kwiatkowska, “Feature-guided black-box safety testing of deep neural networks,” in *Proc. TACAS*, 2018, pp. 408–426.
- [39] Y. Sun, M. Wu, W. Ruan, X. Huang, M. Kwiatkowska, and D. Kroening, “Concolic testing for deep neural networks,” *arXiv preprint arXiv:1805.00089*, 2018.