On the Evaluation of Adversarial Vulnerabilities of Deep Neural Networks

by

Mehrgan Khoshpasand Foumani

Bachelor of Computer Science, UNB, 2018

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Computer Science

In the Graduate Academic Unit of Computer Science

Supervisor(s):	Ali Ghorbani, Ph.D., Faculty of Computer Science
Examining Board:	Huajie Zhang, Ph.D., Faculty of Computer Science, Chair
	Ebrahim Bagheri, Ph.D., Faculty of Computer Science
	Maryhelen Stevenson, Ph.D., Electrical & Computer Engineering

This thesis is accepted by the
Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

August, 2020

Abstract

Adversarial examples are inputs designed by an adversary to fool the machine learning models. Despite having overly simplifying assumptions and extensive studies in recent years, there is no defense against adversarial examples for complex tasks (e.g., ImageNet). However, for more straightforward tasks like hand-written digit classification, a robust model seems to be within reach. Having the right estimation of the adversarial robustness of the models has been one of the main challenges researchers face. First, we present AETorch, a Pytorch library containing the strongest attacks and the common threat models that make comparing the defenses easier.

Most of the research about adversarial examples have focused on adding small l_p bounded perturbations to the natural inputs with the assumption that the actual label remains unchanged. However, being robust in l_p bounded settings does not guarantee general robustness. Additionally, we present an efficient technique to create unrestricted adversarial examples using generative adversarial networks on MNIST, SVHN, and fashion-mnist datasets. We demonstrate that even the state-of-the-art adversarially robust MNIST classifiers are vulnerable to the adversarial examples generated with this technique. We demonstrate that our method is better than the previous unrestricted techniques since it has access to a bigger adversarial subspace. Additionally, we show that examples generated with our method are transferable. Overall, our findings emphasize the need for further studying the vulnerability of neural networks to unrestricted adversarial examples.

Dedication

I would like to dedicate this thesis to my amazing parents. Their emphasis on education motivates me every day to become a better version of myself.

Acknowledgements

I would like to express my sincere appreciation of my supervisor's support. Dr. Ali Ghorbani's continuous guidance throughout my master study helped me stay motivated and his knowledge inspired me to push further.

Nomenclature

$\|x\|_p$ l-p norm of x = $(\sum_{r=1}^{n} |x_r|^p)^{1/p}$, page 16

$C(x)$ Deep detector filters, page 49

D Discriminator in GANs, page 13

$d(p, q)$ Jensen–Shannon divergence, page 14

$D(x, y)$ Some distance metric, page 16

D_{KL} Jensen–Shannon divergence, page 14

$F(x)$ Classification function, page 15

G Generator in GANs, page 13

$L(x, y)$ Loss function, page 17

O Oracle, page 17

p^* True Distribution of data, page 13

T Number of gradient steps, page 37

z Input noise of the generator of GANs, page 13

$Z(x)$ Logits of the model, page 18

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction

In recent years, machine learning models have become an essential part of many critical tasks in human society. By proving human-level or above human-level accuracy in many tasks like computer vision or voice recognition, machine learning models are replacing humans or helping them in some of the essential tasks. Machine learning models are making decisions that can significantly affect human lives. From driving cars to detecting credit card fraud, machine learning models are being adopted in many industries. Hence, the security of machine learning models is an important line of research in recent years.

The security of machine learning models is studied on multiple levels. Preserving the privacy of the training data, authenticating the owner of the model, defending against training time attacks (e.g. data poisoning) and defending against test time attacks (e.g. adversarial examples) are some of the main categories of machine learning security research. The main focus of this work is on measuring the security of the models against test-time attackers.

In 2013, Szeedy et al. [71] showed that an adversary could easily fool deep neural

networks into making a mistake with high confidence by providing carefully picked inputs to the model. These so-called adversarial examples have gained a significant amount of attention in recent years. Some works have shown the possibility of performing adversarial attacks in real-world scenarios [6, 36]. Despite the existence of adversarial examples in other domains(eg. audio [12], malware detection [26], Ad-blocking [75]), most of the studies has been focused on image classification. These vulnerabilities motivate the researcher to improve the performance and the security of deep learning models.

One of the primary motivations to study adversarial examples is to make deep learning models more secure in performing these tasks [75, 6, 36, 3, 18]. Another motivation for researchers to study adversarial examples is to make deep learning models more robust and improve the performance of these models in more challenging settings [57, 11, 63, 2, 81].

Many defenses have been proposed to overcome the shortcomings of deep learning models in the adversarial settings [42, 53]. It has been shown that training a deep learning model robust to all the possible adversarial examples might not be possible [64]. Hence, most of the proposed defenses focus on a limited number of threat models. In this context, a threat model is set of assumptions that the defender has on the capabilities and the goals of the attackers. For example in white-box threat models the attacker has access to the weights and the architecture of the model, however in black-box attacks, the attacker might only have access to the outputs of the model. The most common threat model studied in the literature is l_p bounded perturbation based adversarial examples. In this threat model, given a natural input, the adversary tries to find norm-bounded perturbations that adding it to the original image causes the model to make a mistake. By bounding the perturbation in perturbation-based adversarial attacks, we assume that the true label of examples does not change. This assumption solves the problem of having a ground truth for

adversarial examples. In some cases, for evaluating the robustness of a model, an adversary measures the mean distance of the unbounded adversarial examples from the original inputs. The common studied norms are l_∞ and l_2. Training a robust model even in this limited setting is surprisingly hard even for a simple dataset like MNIST. The inability of the machine learning community to create a robust model even has raised questions about the possibility of the existence of robust models against adversarial examples [64].

The approaches that the defenses in the literature take vary significantly. One line of research is to detect adversarial examples and refuse to classify them [45]. Another approach is to reduce the models' sensitivity and limit the change in the output caused by small perturbation of the input [14, 57]. Although some defense methods provide provable robustness against small perturbed inputs, the best empirical results on MNIST are achieved by non-provable defenses [62, 42].

Despite the much effort, the progress of defending against adversarial examples has been surprisingly slow. One of the main challenges researchers are facing is to evaluate the robustness of their models properly [8]. For the reason that the exact calculation of most models' adversarial robustness is infeasible [42], the claimed robustness of most of the proposed defenses relies on empirical evaluations. Many defenses have been proposed in the literature that claims some level of robustness against adversarial examples. However, most of the proposed defenses are shown not to be effective [8, 11, 2, 9]. This fact shows that the initial evaluation of the defenses was incomplete or incorrect, and the community can greatly benefit from better evaluations. Additionally, having a standard evaluation makes the process of comparing to previous works easier and leads to more meaningful results.

This thesis presents multiple contributions. We propose AETorch, a library that helps researchers better evaluate the robustness of their models. This library includes multiple state-of-the-art adversarial attacks in common scenarios for widely used

datasets. By applying standard attacks in a fixed set of threat models, AETorch can help researchers compare the robustness of their models with other approaches.

With the recent advancement, training a robust model against l_p bounded attacks for toy datasets line MNIST is within reach. These advancements lead to the need to evaluate the robustness of the models in more general settings. For this purpose, we propose a technique to create unrestricted adversarial examples. One main problem for generating the unrestricted adversarial examples is finding the actual label. For our attack, we train a Generative Adversarial Network (GAN) for each class of data to produce class-conditional images. We search inside data manifold captured by the class-conditional generator to find an adversarial example. With this approach, we can assume the true label of the generated labels based on the network used for the generation. While this attack does not scale well to the bigger datasets (e.g., Imagenet), It demonstrates that even the state-of-the-art defense for MNIST dataset are not robust in general. This technique provides a faster and more efficient way to produce unrestricted adversarial examples than other similar approaches [33]. We also show that our technique can be used to bypass the state-of-the-art defenses [42, 62].

The similarity of decision boundaries of different models causes the *transferability* of adversarial examples [78]. Transferability is a property of adversarial examples that cause the adversarial examples generated for one model to remain adversarial for other models. Our analysis shows that the previous GAN based unrestricted adversarial attack [68] only considers the adversarial examples that do not transfer between an auxiliary classifier and the target classifiers. We measure the difference between intra-boundaries distance of an auxiliary classifier and baseline CNNs and show the similarity of their decision boundaries. From these experiments, we conclude that the previous approach ignores a large part of the adversarial sub-space.

On the other hand, our method uses GANs to learn the actual distribution of the data

[65] and hence is superior to the previous approach. Additionally, we demonstrate that the adversarial examples generated with our proposed technique are transferable between different models. We also use the transferability to bypass the state-of-the-art defense of the common datasets.

1.2 Summary of Contributions

The main purpose of this thesis is to provide the tools that help researchers better estimate the weaknesses of their models. The contributions of this thesis can be summarized as follows:

- Defining high-level concepts of adversarial robustness of deep neural networks.
- Measuring the effectiveness of existing adversarial defense mechanisms.
- Developing a library that provides an efficient implementation for existing adversarial attacks.
- Evaluating some of the current defenses against the worst-case adversaries using the implemented library.
- Identifying the shortcomings of previous unrestricted adversarial attacks theoretically and experimentally.
- Proposing a new unrestricted adversarial attack using unconditional generative adversarial networks and increasing the stability of training such models.
- Leveraging the new adversarial attack to evaluate the state-of-the-are defenses.
- Evaluating the effectiveness of the proposed attack using human labeling.

1.3 Thesis Organization

The rest of this thesis is organized as follows:

Chapter 2 provides the background information needed for the proposed method. First, Chapter 2 presents an overview of deep learning techniques. Since we use generative adversarial networks to generate unrestricted adversarial examples, Chapter 2 provides an overview of GANs. Additionally, it explains the bias in AC-GANs that motivates us to propose a new adversarial attack. Chapter 2 also reviews approaches that have been previously taken to define and measure the adversarial robustness of machine learning models. It's infeasible to calculate the exact adversarial robustness for the models, since calculating the exact robustness of models requires knowing the exact decision boundary of the model and the task's decision boundary. Hence, the primary approach to measure the adversarial robustness of a model is to estimate an upper bound for it by performing adversarial attacks. In chapter 2 we reviews the common approaches used for this purpose.

Furthermore, chapter 2 reviews the common adversarial attacks and threat models studied in the literature. Additionally, chapter 2 categorizes and reviews the previously proposed techniques that can increase the robustness of a deep learning model. Finally, the second chapter provides background information about the generative adversarial networks.

Chapter 3 goes over the details of TorchAE; the implemented library used to show the weaknesses of some previously proposed defenses. This library uses Pytorch to provide GPU acceleration and contain the most common attacks and threat models. Afterwards, this chapter presents the details of the proposed unrestricted attack and how it can be used in different threat models. This chapter provides a general overview of the proposed adversarial attack, followed by the details of each of its modules. Finally, the algorithm for generating unrestricted adversarial examples from unconditional GANs is proposed.

Chapter 4 presents the experimental results for this thesis. First, this chapter goes over the experimental results that motivate us to present a new unrestricted adversarial attack. These experiments show the shortcomings of the previous approach which was using the AC-GANs to generate unrestricted adversarial examples. Secondly, this chapter provides the results of our experiments with AETorch. These results present a more accurate estimation of the robustness of three adversarial defenses proposed in recent years. We show that these defenses have fallen in some common pitfalls in the evaluation of their robustness. Afterwards, this chapter presents the experimental results of our proposed unrestricted attack. We show the effectiveness of this attack by attacking state-of-the-art defenses in the hardest threat models on three standard datasets, MNIST, fashion-mnist, and SVHN. Finally, we compare our attack to the previous approach by performing human study on the actual labels of the generated adversarial examples.

Finally, Chapter 6 provides a conclusion of the thesis by discussing the contributions. Additionally, it goes over the limitations of the proposed attack and possible future works to improve the work done in this thesis.

Chapter 2

Literature Review

2.1 Introduction

In recent years, machine learning has become the core element of many modern systems. These systems make use of machine learning to make decisions that can significantly affect human lives. Specifically, Deep learning is a widely adopted method that has shown promising results in computer vision, speech recognition, and speaker verification tasks.

While machine learning models have an adequate expected error in many tasks, the work by Goodfollow et al.[23] showed that they could be easily tricked. As it has been studied before, a pitfall for using machine learning model in critical systems is that an adversary can design an input that tricks the machine learning models into making a wrong decision[23][54][77]. This particular class of inputs is known as adversarial examples and has raised security concerns regarding machine learning models.

First, this chapter provides background information about deep neural networks. Specifically, it goes over generative adversarial networks used as a crucial part of the proposed attack. Afterwards, this chapter discusses the adversarial robustness of ma-

chine learning models by discussing the proposed definitions in the literature. Then, it provides a comprehensive literature review on the topic of adversarial examples in deep neural networks and discusses significant adversarial attacks and defenses proposed in the literature.

2.2 Machine Learning Background

Machine Learning (ML) is a field of study in computer science that gives computers the ability to learn to perform a task without being programmed for a specific task. In general, a machine learning task consists of running a machine learning algorithm on training data. The ML algorithms result in a set of learned parameters. Usually, these parameters are validated using a subset of available data not seen during the training. Finally, the learned parameters are used at the testing phase with the new data. Like other computer science systems, any part of the machine learning process can be the target of security attacks.

- attack on the training data: In the process of data collection, the adversary might change a subset of the data to manipulate the behaviour of the machine learning models. This type of attack is often called data poisoning.

- attack on the learned parameters: the attackers might be able to use the trained parameters of the published models to get access to the private data used for the training. Membership inference attacks are an area of interest in recent years. The purpose of these attack is to predict whether a specific data point was in the training data which puts the privacy of the training data at risk.

- attack on the test inputs: One of the most widely studied aspects of machine learning security are attacks at the inference time (testing time). Adversarial examples are the inputs that the adversary provides at the testing time with the

purpose of fooling the machine learning model into producing wrong outputs. This definition distinguishes between adversarial examples and common test errors by stating that the adversary has to provide these inputs carefully. This definition, however, does not clarify the ground truth to identify the mistakes of the machine learning models. The following section goes into more detail on the adversarial examples.

- attack using the test output: Machine learning models are becoming more and more expensive to train and require an extensive amount of time, computation power and hard-to-collect data. Model theft is a type of attack on the machine learning models that can be done solely using the output of the ML models [79]. The adversary can use the model's output to train another model that mimics the behaviour of the original models. ML model watermarking is another research topic where the owner of the ML model embeds the watermark into the model. In the context of ML, watermarking refers to the techniques used by the owner of the models to verify ownership of the model based on the output of the models on the specific set of inputs.

2.2.1 Neural Networks

Deep learning is a sub-field of machine learning, where many layers of information-processing stages are exploited inside a Neural Network (NN). The goal of deep learning is to automatically discover the representations needed for the detection or classification of raw data with minimal engineering and domain expertise needed. Deep Learning achieves this goal by feeding the raw data to the first layer. Each layer consists of several units. Each unit applies a simple non-linear function to the weighted sum of inputs that are coming from the previous layer. A layer transform representation of the previous layer into a slightly higher level of abstraction and pass it to the next layer. During the training, to minimize some error function, DL

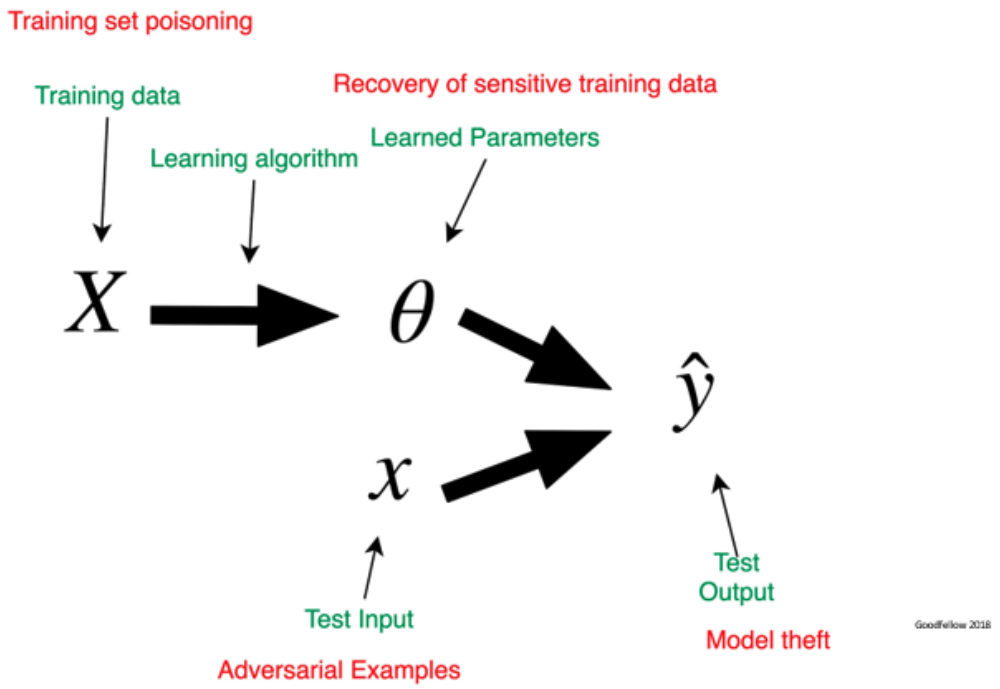


Figure 2.1: Type of attacks to the Machine Learning process

algorithms adjust some internal parameters, called weight. The weights are adjusted by calculating the gradient vector of the error for each weight. The gradient vector shows how much the error changes by changing the weight by a tiny amount. This method of finding the local minimum of a function is called Gradient Descent(GD). Since GD method weights are adjusted for each sample, it is very costly for big training datasets. Instead, Stochastic Gradient Descent (SGD) is often used. In SGD, the average gradient vector is computed for a small number of samples from the training set, and weights are adjusted accordingly [37]. Surprisingly, this simple procedure usually finds a good set of weights[37]. Numerically evaluating an analytical expression of the gradient can be costly. Hence, most DL algorithms use the back-propagation algorithm to compute the gradient. Back-propagation is an inexpensive procedure that allows information to flow backward through the network. [21] In a nutshell, back-propagation is an application of chain rule, computing the gradient of the function in respect of input by working backward from the gradient

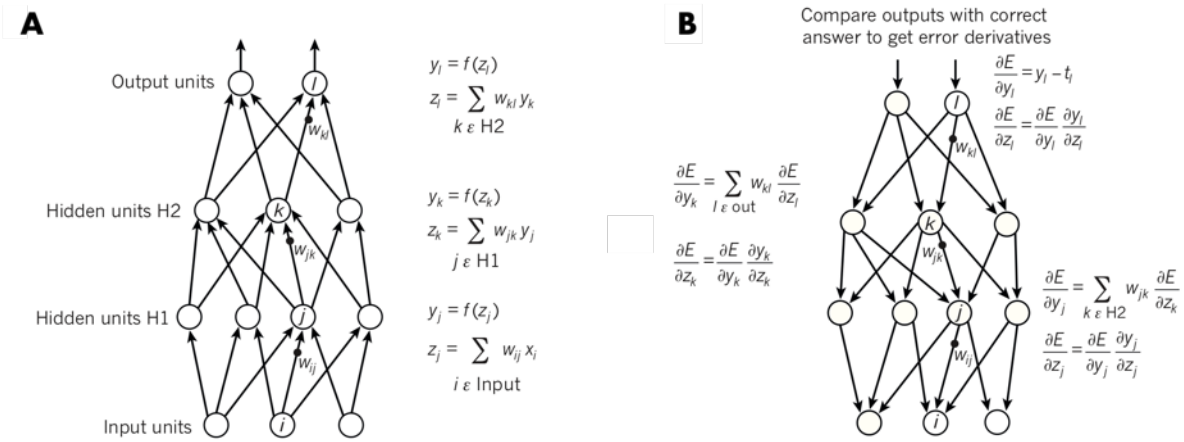


Figure 2.2: Adapted from [37] A. Shows equations for the forward pass in a multi-layer neural network with two hidden layers. At each unit, a non-linear function is applied to the weighted sum of the outputs of the units in the layer below. Then the unit passes its output to the layer. B. Shows the back-propagation procedure.

of a function in respect of output.

2.2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a class of deep neural networks that have been applied with great success to the detection, segmentation, and recognition of objects and regions in images[37]. CNNs are powerful representation learning techniques for the data that come in the form of multiple arrays. CNNs take advantage of the fact that the local statistics of images and other signals are usually invariant to location[37]. This characteristic makes CNNs useful to process sequence, natural language, image, and audio. CNNs are based on four key ideas: local connections, weight sharing, pooling and using many layers to build a deep neural network. Typical CNN consists of a series of processing stages. The first few stages contain convolutional layers and pooling layers. Units in a convolutional layer are connected to the units of the previous layer through a set of weights, and then, a non-linearity such as a ReLU is applied. All units in a feature map share the same weights. However, different feature maps in a layer use different weights[37]. Pooling layers are

used to merge the semantically similar features or select the significant features from the feature maps and reduce the number of features. Figure 2.3 shows the overall architecture of CNNs for classification tasks.

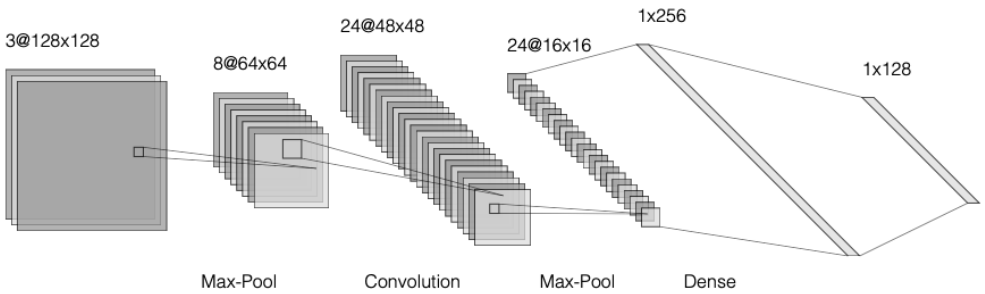


Figure 2.3: Overall architecture of CNN for classification tasks.

2.2.3 Generative Adversarial Networks

Generative Adversarial Networks (GANs) [22] consist of a generator G that is trained to map random noise z to points inside the data distribution and a discriminator D that is trained to distinguish between generated (fake) and real points. In recent years, various variations of GANs have been proposed to stabilize the training of GANs and improve their results [1, 27, 43]. Formally, the Min-Max game between the generator and the discriminator is:

$$\min_{G} \max_{D} \mathop{\mathbb{E}}_{\boldsymbol{x}\sim \mathrm{I}^*}[\log(D(\boldsymbol{x}))] + \mathop{\mathbb{E}}_{\hat{\boldsymbol{x}}\sim G(z)}[\log(1 - D(\tilde{\boldsymbol{x}}))], \tag{2.1}$$

where p^* is the true distribution, D is the discriminator which is a neural network that outputs a real number representing the likelihood of the data point being real, G is the generator, and z is random noise. It can be shown that GANs objective is equivalent to minimizing Jensen-Shannon divergence of true distribution and distribution learned by the generator.

Auxiliary Classifier GANs [50] are popular choice for generating class conditional images. The discriminator of an AC-GAN uses an auxiliary classifier to classify the image as well. Additionally, the generator learns to generate samples that appear to be real and has the desired class based on the auxiliary classifier. In a well-trained AC-GAN, the decision boundary of the auxiliary classifier approximates the distribution learned by the generator. Suppose $p^*(x, y)$ is the true joint distribution, $p_\theta(x, y)$ is the generator, and $q_\phi(y|x)$ is the auxiliary classifier, with x being natural images and y being the labels. AC-GANs minimize the following objective [65]:

$$\min_{\theta,\phi} d\left(p^*(x), p_\theta(x)\right) + \lambda_m \mathcal{L}_m(\theta, \phi) + \lambda_c \mathcal{L}_c(\phi), \tag{2.2}$$

where $d\left(p^*(x), p_\theta(x)\right)$ is Jensen–Shannon divergence, λ_m, λ_c are weighting constants and

$$\mathcal{L}_m(\theta, \phi) = -\mathbb{E}_{(x,y)\sim p_\theta} \ln q_\phi(y|x) \tag{2.3}$$

$$\mathcal{L}_c(\phi) = \mathbb{E}_{x\sim p^*} \mathrm{D_{KL}}\left(p^*(y|x) \| q_\phi(y|x)\right), \tag{2.4}$$

and D_{KL} is the Kullback-Leibler divergence.

The second term of the objective can be decomposed as follows:

$$\begin{aligned} -\mathbb{E}_{p_\theta}(x, y)\left[\ln q_\phi(y|x)\right] = \\ H_\theta(Y|X) + \mathbb{E}_{p_\theta(x)} \mathrm{D_{KL}}\left(p_\theta(y|x) \| q_\phi(y|x)\right), \end{aligned} \tag{2.5}$$

where $H_\theta(Y|X)$ is the conditional entropy. By looking at the second term of equation 2.5, during the training of AC-GANs, the distance between the distribution learned by the generator and the auxiliary classifier is minimized. This property indicates that AC-GANs are biased towards down sampling the points near the de-

cision boundary of the auxiliary classifier. We use this property in the Section 4.2.1 and by showing the similarity of the decision boundaries of the auxiliary classifier and the baseline CNN; we conclude the similarity of the distribution learned by the class conditional generator in the AC-GANs and the baseline CNN.

2.3 Adversarial Robustness

Recent works have demonstrated the existence of adversarial examples in deep learning models[71, 24]. Adversarial examples are the inputs created by an adversary with the intention of causing the model to make a mistake. More formally, for a classifier F_θ, the attacker is looking for an input x^* such that $F_\theta(x^*) \neq y$, where y is the true label of x^*. Adversarial examples with respect to the intention of the adversary can be classified into two classes. First, targeted attacks in which the adversary has a specific class,y^*, that wishes the model to produce: $F_\theta(x^*) = y^*$. Second, untargeted attacks where the only intention of the adversary is to produce an example that is misclassified by the model.

One of the critical problems in generating adversarial examples is knowing the ground truth. Hence, adversarial examples are often studied as small l_p bounded perturbation that is added to natural examples. In this threat model, given a natural input, the adversary tries to find norm-bounded perturbations that adding it to the original image causes the model to make a mistake. By bounding the perturbation in perturbation-based adversarial attacks, we assume that the correct label of examples does not change. This assumption solves the problem of having a ground truth for adversarial examples. One of the standard definitions for adversarial robustness of an classifier is the highest loss that can an adversary get inside the norm-ball of natural examples [8]:

$$\mathop{\mathbb{E}}_{(x,y)\sim\mathcal{X}}\left[\max_{x':\mathcal{D}(x,x')<\epsilon} L\left(f\left(x'\right),y\right)\right].$$

The common distance metrics are l_∞, l_2, and l_0, where $\mathcal{X}$ are the natural samples, L is cross-entropy loss, and $D(x,y)$ is some distance metric between the two points. Distance-based adversarial examples are also studied in unbounded settings. The robustness of the model against the unbounded attacks are often reported as the mean minimum-distance adversarial examples that are found around the natural data.

$$\mathop{\mathbb{E}}_{(x,y)\sim\mathcal{X}}\left[\min_{x'\in A_{x,y}} \mathcal{D}\left(x,x'\right)\right]$$

, where $\mathrm{A}_{x,y} isanyadversarialattackusedbythetheattackerwiththepurposeofmeasuringtheadversa$

Outside l_p bounded attacks, Wong et al. [85] has studied the Wasserstein distance bounded adversarial examples. Engstrom et al. have shown that simple transformations can cause a well-trained neural network to make a mistake [17]. Hosseini et al. showed that while negative images preserve the semantic meanings, they can cause miss-classification [31]. Adversarial examples can also be the result of the model being too invariant. Jacobsen et al. found that increasing the l_p bounded robustness of neural networks can increase the venerability to the invariance-based adversarial examples [33].

2.4 Adversarial Attacks

In this section, we present some of the common techniques to produce adversarial examples in the literature.

2.4.1 Restricted Attacks

The definition of adversarial examples does not restrict how the adversary can produce the input that is to be misclassified by the model. In the literature, however, in order to easily determine the ground-truth of the produced inputs, adversarial examples are usually restricted to be in a small distance of a actual sample. In many cases, adversarial examples are generated by adding a small perturbation to the actual samples. While these methods do not cover all the possible adversarial examples, defending against these types of attacks is still an active line of research [15].

In perturbation based adversarial attacks, given an input sampled from natural distribution x, the adversary looks for an adversarial example x^* such that $F(x) = O(x) = O(x^*) \neq F(x*)$ where O is the oracle. Perturbation based adversarial examples are usually ϵ-bounded so $||x - x^*||_p < \epsilon$, where $||.||_p$ is the l_p norm.

2.4.1.1 Gradient-based Attacks

- **Fast Gradient Method:** FGM [24] is one of the simplest adversarial attacks and involves moving in the direction of the gradient of loss function with respect of inputs. The l_∞ bounded FGM is called Fast Gradient Sign Method and is performed as follows:

$$\mathbf{x}' = \mathbf{x} + \epsilon \cdot \text{sign}\left(\nabla_{\mathbf{x}} L(\mathbf{x}, y)\right),$$

  where y is the true label of normal sample x and $L(x, y)$ is the loss function. It's common to use cross-entropy loss for the classification tasks.

- **Projected Gradient Descent:** PGD [42, 36] is an iterative variant of FGSM where the results are projected inside the norm-ball around the natural exam-

ple.

$$\mathbf{x}^{(m)} = \mathbf{x}^{(m-1)} + \epsilon \cdot \operatorname{sign}\left(\nabla_{\mathbf{x}^{(m-1)}} L\left(\mathbf{x}^{(m-1)}, y\right)\right)$$

Madry et al. showed that PGD finds the near-optimal adversarial examples inside the defined norm-ball [42].

- **Carlini Wagner L2:** CW is a gradient based attack that finds the perturbation with minimum l_2 distance that maximize the loss function. CW attack is formulated as:

$$\begin{aligned} &\text{minimize } \|\delta\|_2 + c \cdot f(x+\delta) \\ &\text{such that } x + \delta \in [0,1]^n \end{aligned}$$

  with f defined as

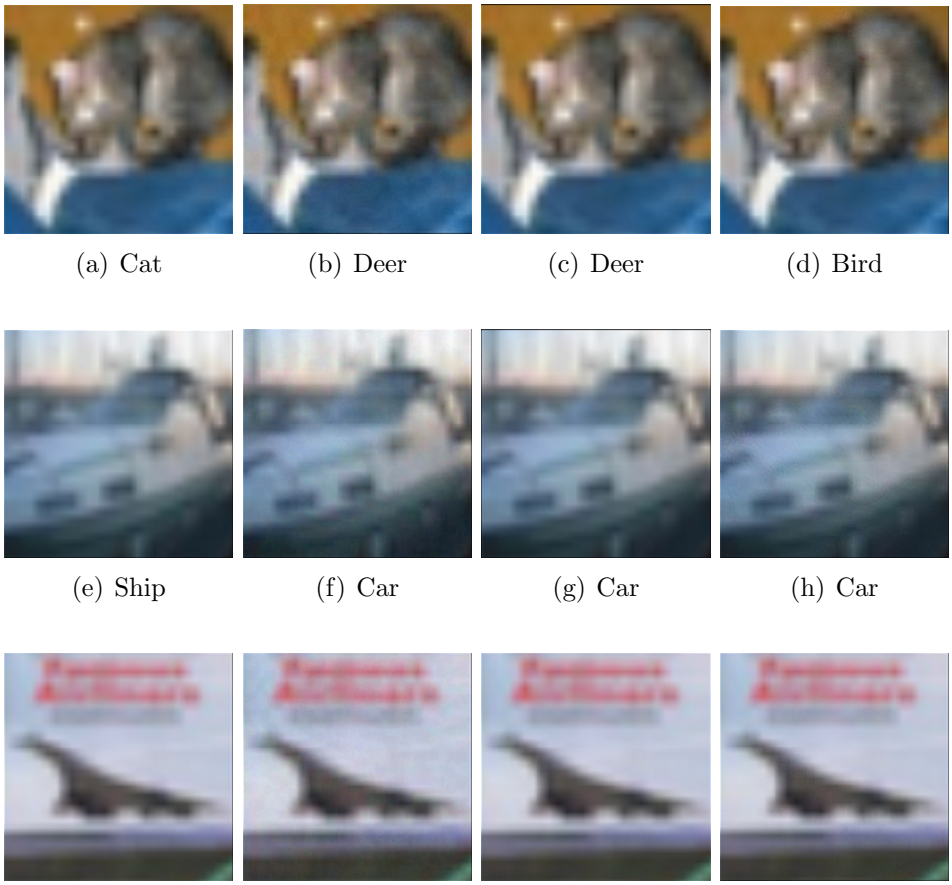
$$f\left(x'\right) = \max\left(\max_{0 \leqslant i \leqslant n}\left\{Z\left(x'\right)_i : i \neq t\right\} - Z\left(x'\right)_t, -\kappa\right),$$

  where $Z(x)$ is the *logits* of the model, t is the target class, and κ is a parameter that controls the confidence for the decision of the model [11].

- **EAD:** Elastic-Net Attacks to Deep Neural Networks is the state-of-the-art l_1 attack that uses Elastic-net regularization and combines L1 and L2 penalty functions:

$$\begin{aligned} &\text{minimize } {}_x c \cdot f(x,t) + \beta \left\|\mathbf{x} - x_0\right\|_1 + \left\|x - x_0\right\|_2^2 \\ &\text{subject to } x \in [0,1]^p \end{aligned}$$

  While change of variable is applied in CW attack to enforce the box-constraint, it's not effective when L1 penalty is used [13]. Chen et al. uses iterative shrinkage-thresholding algorithm [4] to solve this problem and proposes EAD attack [13].



(a) Cat (b) Deer (c) Deer (d) Bird

(e) Ship (f) Car (g) Car (h) Car

(i) Plane (j) Ship (k) Ship (l) Ship

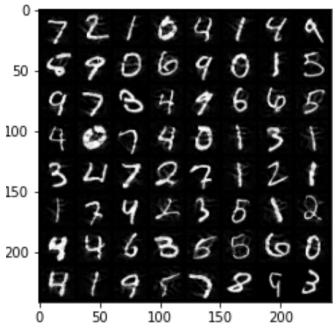
Figure 2.4: First column is a real image; second column is an example of the FGM attack; third column is an example of the PGD attack; and the last column is an example of the CW attack on CIFAR10 dataset.

2.4.1.2 Gradient-free Attacks

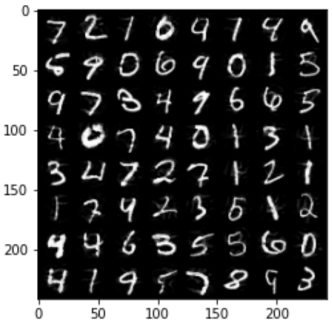
- **SPSA:** SPSA (Stochastic Approximation based on Simultaneous Perturbation) is a gradient-free attack that uses SPSA algorithm [69] to approximates the gradient instead of taking analytical gradient [81]. Uesato et al. used the SPSA attack to bypass multiple defenses that did not produce analytical gradients or made the gradients not meaningful [81].

- **Boundary Attack:** In real world applications the adversary might only has access to the top-k hard label output of the classifier; with the top-1 being the most limited setting. Brendel et al. proposed Boundary attack that works in these limited settings [5]. Boundary attack is an iterative process in which an adversary starts with a random noise that is classified as the target class. Then the adversary moves the adversarial example towards the original sample. by making a small movement such that the decision of the model remains the same [5]. Considering the limited knowledge that the adversary has in this threat model, boundary attack achieves excellent results.

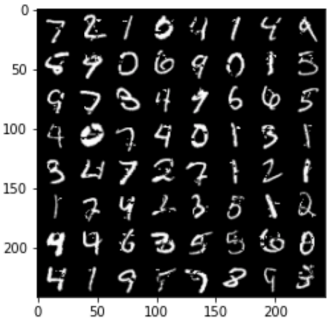
- **Transferablity-based Attack:** Transferability is the property of adversarial examples where the adversarial example generated for one model remains adversarial for other models as well [78, 24]. When attackers have limited knowledge about the target model or the defenses that are in place, they can generate adversarial examples for their model and feed them into the target model. Tramer et al. showed that due to the similarity of decision boundaries of different models, an adversarial example with slightly larger perturbation would transfer with high probability [78].



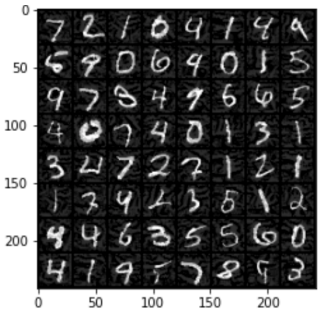
(a) Boundary attack



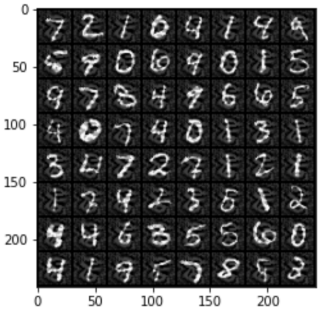
(b) CW attack



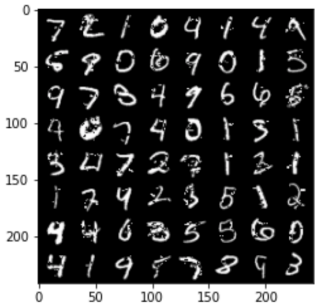
(c) EAD attack



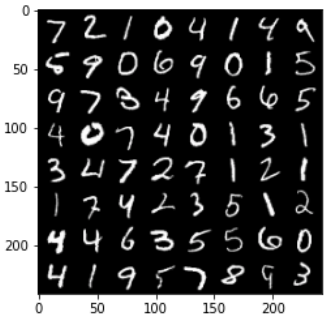
(d) PGD attack



(e) SPSA attack



(f) L_0-Brendel-Bethge attack



(g) Real Image

Figure 2.5: Different attack examples on MNIST dataset.

2.4.2 Unrestricted Attacks

In contrast to the perturbation-based adversarial examples, unrestricted adversarial examples do not have to be in the small distance to a natural example [68].

2.4.2.1 Invariance-based Attacks

For invariance-based adversarial examples, given a natural input, the adversary finds an adversarial example x^* such that the true class of x^* is different than x but ideally the model produces the exact same output: $O(x) = F_\theta(x) = F_\theta(x^*) \neq O(x^*)$. However, for the purpose of knowing the ground truth, the current implementations of invariance-based adversarial attacks are based on minimizing the l_p distance of an adversarial example from an instance of target class while the model still classifies it as the source class. Engstrom et al. argue that a robust model should at least covers both invariance-based adversarial and perturbation based adversarial examples [34, 33].

2.4.2.2 Unrestricted attacks with Generative Models

Song et al. used class-conditional GANs to find unrestricted adversarial examples [68]. Their attack searches in the manifold of an AC-GAN for the points where the decision of the target model does not agree with the distribution learned by the conditional generative model. Unrestricted adversarial examples can be generated without starting from a natural sample x. Song et al. showed that this approach could produce unrestricted adversarial examples, even for models proven to be robust against perturbation based attacks. Chapter 3 provides more details on how Song et al. attack works.

2.5 Adversarial Defenses

In recent years, many defenses have been proposed to overcome the shortcomings of deep learning models in the adversarial settings [42, 53]. Training a robust model even in limited settings is surprisingly hard even for simple a dataset like MNIST. The approaches that the defenses in the literature take vary significantly. One line of research is to detect adversarial examples and refuse to classify them [45]. Another approach is to reduce the sensitivity of the models; limit the change in the output caused by small perturbation of the input [14, 57]. Although some defense methods provide provable robustness against small perturbed inputs, the best empirical results on MNIST are achieved by non-provable defenses [62, 42]. In the rest of this section, we go over some of the notable efforts to defend against adversarial examples in the literature. The proposed defenses can roughly be categorized into the following categories [62]:

2.5.1 Adversarial training

In adversarial training, the model is trained on adversarial examples in addition to the natural data [42, 71, 24, 76]. Madry et al. [42] defense was one of the first defenses to hold an acceptable level of robustness against strong attacks in the defined l_∞ threat model. Madry et al. formulates the problem of training an adversarially robust model against a worst-case adversary as a saddle-point problem:

$$\min_{\theta} \rho(\theta), \quad \text{where} \quad \rho(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\max_{\delta\in\mathcal{S}} L(\theta, x+\delta, y) \right],$$

where $\mathcal{D}$ is the dataset used for training and is the set of allowed perturbations in the used threat model.

They follow the adversarial training[24] procedure and use strong attacks (e.g., Projected Gradient Descent) to generate adversarial examples during the training on the

MNIST dataset. They argue that this procedure leads to the near-optimal solution for the saddle-point problem. Despite being robust within the defined threat model, Madry et al. model show small robustness against strong l_2 attacks or adversarial examples with slightly higher l_∞ perturbation than they defined [62]. One of the significant problems of adversarial training with a strong attack is that it does not scale to a larger datasets like Imagenet [76, 36].

2.5.2 Defensive pre-processing

In defensive pre-processing, the input or the activation of the model is processed to reduce the effect of adversarial perturbations. Gau et al. use image cropping and re-scaling, bit-depth reduction, JPEG compression to remove the adversarial perturbation from the images before feeding them to the network. [28]. They also propose two methods to reduce adversarial perturbation. First, they randomly drop a small set of pixels and use total variation minimization to reconstructs the image is consistent with the selected pixels. Second, they reconstruct images by replacing small patches taken from original images, using minimum graph cuts in overlapping boundary regions to remove edge artifacts [28, 2]. Xu et al. suggest reducing the feature input space and, therefore, degrees of freedom available to an adversary by reducing the colour bit depth of each pixel and spatial smoothing [89]. By following the hypothesis by Goodfellow et al. that the vulnerability of neural networks to adversarial examples is due to their linear behaviour [24], Buckman et al. suggest applying non-linear input discretization would increase the robustness of the model to the adversarial examples [7]. However, it was shown that applying the above pre-processing does not make the model robust to adversarial examples [2].

2.5.3 Randomization

To defend against adversarial examples using randomization, defender adds stochasticity at the inference time. The randomization can be done at the input layer or the hidden layers. Xie *et al.* propose to defend against adversarial examples by adding randomized scaling and zero-padding layers before feeding the input to the convolution neural networks [88]. Dhillon *et al.* add randomness at the inference time by randomly setting the activation of neurons to zero [16]. The defense by Prakash *et al.* randomly shuffles the pixels of the images with their neighbour pixels before feeding it to the neural network [56]. One of the problems with randomization processes is that they can reduce the accuracy of the model on clean images. The accuracy reduction can be minimized by carefully measuring the amount of added noise similar to differential privacy techniques [39]. Some studies have found that the randomization defenses can have limited benefits if the attacker is aware of the defense [2].

2.5.4 Manifold projection

Researchers have attempted to defend against adversarial examples by projecting the inputs to the learned manifold at the inference time. Shen et al. proposed MagNet that uses auto-encoders to project the inputs into the natural inputs submanifolds [44, 10]. Defense-GAN used Generative Adversarial Networks to find the most similar points inside the manifolds that are learned on original data and feed that point to the model [61]. Song *et al.* use a PixelCNN generative model to project inputs onto the data manifold before feeding it into a classifier [67]. Athalye *et al.* bypassed PixelDefend and Defense-GANs and showed that they cause exploiting or vanishing the gradients instead of increasing the robustness of the model against worst-case adversaries [2].

2.5.5 Other Defenses

2.5.5.1 Deep k-Nearest Neighbors

Deep k-Nearest Neighbor (DkNN) [52] applies the k-nearest neighbour search on the representations of the data learned by layers of the neural network. Papernot et al. showed that DkNN could improve adversarial robustness, offer interpretability, and detect adversarial examples. The intuition behind DkNN is that samples from the same class typically have similar representation in different layers of the neural networks. Additionally, by using DkNN, Papernot et al. found some mislabeled data in common image classification datasets. Sitawarin et al. [66] performed experiments with the proposed DkNN on MNIST and CIFAR-10 and compare it with state-of-the-art defenses against l_2-perturbation. Their work shows that DkNN sets a better trade-off between accuracy of the model on clean data and adversarial robustness.

2.5.5.2 Analysis by Synthesis

In general, feed forward neural networks suffer from distal adversarials [49]. Distal adversarials refer to inputs that are classified by the model with high confidence but do not belong to any class and typically look like noise while still being [63]. However, generative classification models learn the distribution of their inputs and generally do better on distal adversarials [63]. Schot et al proposes to learn the image distribution within each class of MNIST dataset. They introduced a baysian classifier based on Variational Auto Encoders (VAEs) that provides adversarial robustness as well as high accuracy on clean input on MNIST dataset.

Their comprehensive robustness evaluation shows that Analysis by Synthesis (ABS) and its variations (Binary ABS) are the best defenses to this date considering all the threat models.

2.5.5.3 Certified Defenses

All the claimed robustness of the discussed defences so far, are based on the empirical results. The empirical defenses do not provide any robustness guarantee; they have only shown to have some level of robustness against some types of current adversarial attacks. Hence, empirical defenses typically overestimate the level of adversarial robustness they provide. To address this issue, some works provide a provable guarantee, which means a certification that the decision of the model does not change in a certain l_p ball around the target point. Current methods in the literature typically verify by computing either the norm of the minimal perturbation of the natural point in order to change the decision of the model [35, 72] or computer lower bounds on the perturbation[15, 30, 57, 84]. Provable defenses usually provide new training schemes to enhance the robustness of networks and to produce models that are easier to verify their robustness with the verification methods[15, 87, 25, 46].

2.6 Concluding Remarks

In the past few years, research on the security of machine learning models has seen an enormous increase. Specifically, the security of deep learning models at the inference time has been a topic of interest. Despite all these efforts, the community has not been able to train thoroughly adversarially robust models and also has failed to explain this phenomenon fully.

Some findings to this date try to explain why adversarial examples exist. Shafahi et al. [64] show that adversarial examples are inevitable in some settings and provide some evidence that this phenomenon might be the case in real-world settings. Ford et al. show a relationship between adversarial robustness and the accuracy of the model under randomly corrupted inputs [20]. This finding suggests that the adversarial robustness is tightly related to the robustness of the model in the presence of the

random noise. Iliyas et al. suggest that there are non-robust features that are highly predictive in machine learning datasets that lead to the existence of adversarial examples. This finding indicates that having a robust model is reachable by encoding the human priors into the training process [32].

Designing an adversarially, robust model has been a challenging task. Some works have studied these challenges fundamentally. Tsipras et al. show that by increasing the robustness, the accuracy of the model on clean data might decrease [80]. They prove this trade-off in simple settings that are in line with real-world findings. Making a model invariant to the perturbations can also cause the model to be *too invariant*[73]. Recent findings suggest that there might be a trade-off between the robustness of the model under different l_p-bounded and spatial perturbations [74]. These findings align with the work by Schott et al., which is one of the most robust models on the MNIST dataset to this date [62].

Transferability is a property of adversarial examples where a sample generated for a model can fool different models. Adversarial examples can transfer even if the target model has different architecture or was trained on a disjoint training set [71]. By using this property, an adversary might not even need to have access to the model's architecture or the training set in order to be able to fool DNNs. Some works explain the transferability of adversarial examples by measuring the similarity of decision boundaries of different models [78]. An adversary can use the similarity to find *universal perturbations*, perturbations that can be added to any sample and lead to misclassification by any model [47]. Despite all the works, measuring the transferability of adversarial examples and stopping adversarial examples from transferring remain open problems.

Chapter 3

Adversarial Robustness Evaluation

3.1 Overview

Despite the much effort, the progress of defending against adversarial examples has been surprisingly slow. One of the main challenges researchers are facing is to evaluate the robustness of their models properly [8]. For the reason that the exact calculation of most models' adversarial robustness is infeasible [42], the claimed robustness of most of the proposed defenses relies on empirical evaluations. Many defenses have been proposed in the literature that claims some level of robustness against adversarial examples. However, most of the proposed defenses are shown not to be effective [8, 11, 2, 9]. This fact shows that the initial evaluation of the defenses was incomplete or incorrect, and the community can greatly benefit from better evaluations. Additionally, standard evaluations make the process of comparing to previous works easier and lead to more meaningful results.

As one of the contributions of this thesis, we developed a tool to help researchers in the process of evaluation of the robustness of their models. Although the tool cannot perform adaptive attacks based on the specific characterization of defenses, it helps the researchers follow the suggested guidelines of Carlini *et al.* [8] to avoid

incorrect evaluation. Moreover, researchers can change the tool according to their needs in order to evaluate the robustness against the worst-case adversaries.

As was discussed, training an adversarially robust model even in limited settings on simple datasets like MNIST [38] is yet to be achieved. The machine learning community is investigating the possibility of creating an adversarially, robust model that resists all attacks. [64]. However, researchers have created techniques for simple tasks that resist adversarial attacks on some levels. Previous studies demonstrate the high robustness of the Schott *et al.* defense [62] in different threat models. As another contribution, we propose an unrestricted adversarial attack based on generative adversarial networks that successfully generate adversarial examples for Schott *et al.* and other state-of-the-art defenses, even in the most strict threat model which makes it more difficult for the attackers to produce adversarial examples. Additionally, we investigate the advantages of the proposed attack on the previous unrestricted attacks. We also show the points where future works can improve the proposed attack. While this attack does not scale well to bigger datasets (*e.g.*, Imagenet), It demonstrates that even state of the art defense for MNIST dataset are not robust in general.

3.2 TorchAE, An Adversarial Robustness Evaluation Library

This section goes over the implementation details of the tool we developed to help researchers to evaluate the robustness of their model. This tool is developed using Pytorch [55] and provides GPU acceleration. Currently, the tool only supports MNIST dataset; however, other datasets will be added in the future. The development of this tool was based on the suggestion of Carlini *et al.* [8]. At the designing stage, we tried to include a diverse set of attacks to perform a comprehensive evaluation. It

Approach	Attacks
Gradient-Based	EAD, CW, FGSM, PGD
Gradient-free	SPSA, Boundary attack, Transferability
Corruption	Gaussian noise, Uniform noise, Salt and pepper

Norm	Attacks
l_1	EAD
l_2	CW, Boundary attack
l_∞	PGD, FGSM, SPSA

Table 3.1: Categories of attacks implemented in our tool

includes different threat models including norm-bounded perturbation based attack, unbounded perturbation-based attacks, targeted and untargeted attacks, white-box and black-box attacks, gradient-based and gradient-free attack, and unrestricted attacks. While some libraries or tools provide some of the adversarial attack[58, 51], in this tool, we provide a standard evaluation on common datasets and help the researchers to evaluate their models and compare them with the previous works. This tool performs the attacks with the common hyper-parameter settings on MNIST models and reports the attack's success rates. Furthermore, this tool provides the ability to perform adaptive attacks by tuning the hyper-parameters. To the best of our knowledge, there is only one website [1] with the same goal that does not have several strong attacks. More details about the categories of implemented attacks can be found in Table 3.1.

The users can create Evaluation objects and call the desired threat model to perform the adversarial evaluation. Currently, the following threat models are supported:

- **restricted_basics:** Calling this method will add Gaussian and uniform noise to the fixed subset of MNIST test set. We project the final sample into the l_∞ ball with the $\epsilon = 0.3$. Additionally, it performs Salt and pepper, which is a basic adversarial attack where random features are set to their maximum or minimum randomly. It reports the accuracy of the model on noisy inputs as well as the average probability for the Salt and pepper attack.

- **restricted_whitebox:** In this document, we use the term white-box for the attacks where the adversary has access to parameters of the model. For this

[1] robust.vision/benchmark/leaderboard/

threat model, the tool performs state-of-the-art targeted gradient-based attacks for l_1 (EAD), l_2 (CW), and l_∞ (PGD). The hyperparameters of the attacks are set to the common values reported in the literature. While the tool performs the attacks with common parameters, in some cases, users might need to change them based on their models' characteristics. The tool reports the mean distortion for CW and EAD and the accuracy for PGD with $\epsilon = 0.3$

- **restricted_blackbox:** In contrast to white-box threat model, we use the term black-box for the attacks where the adversary does not have access to parameters of the model. Hence, the adversary cannot calculates the gradient of the outputs with the respect of inputs. By calling this method, the evaluation is done using SPSA as a gradient-free method where the adversary has access to logits and Boundary attack where adversary only has access to the top-1 prediction of the model. For SPSA, accuracy on $\epsilon = 0.3$ and for Boundary attack, mean l_2 distortion are reported. Additionally, in this threat model, the tool creates adversarial examples using PGD with $\epsilon = 0.3$ for a base CNN and tests their transferability by feeding them into the target model.

- **unrestricted_whitebox** In this threat model, the tool can be used to evaluate the robustness of the model using our unrestricted GAN-based attack. It performs a gradient-based search in latent space starting from 50 randomly sampled noise vector and returns all the adversarial examples it can find. The user can evaluate the robustness of the model to this type of attack based on the number of valid adversarial examples that are found.

- **unrestricted_blackbox** Similar to unrestricted_whitebox, the tool uses the proposed attack to generate adversarial examples. However, since the adversary does not have access to the gradient of the model in a black-box setting, it generates high confidence adversarial examples for the base CNN and finds

the ones that are transferred to the target model. Here, the adversary only needs the top-1 prediction of the target model.

3.3 Proposed Unrestricted Attack

An unrestricted adversarial example can be defined as an input provided by the adversary x^* such that the output of the classifier does not agree with the oracle's decisions on them: $F(x^*) \neq O(t^*)$. Similar to invariance-based adversarial examples, unrestricted adversarial examples are not the results of small perturbations; however, unrestricted adversarial examples can be generated without starting from a natural sample x. Previously, Song *et al.* [68] used AC-GANs to produce unrestricted adversarial examples even for models that are proved to be robust against perturbation based attacks. Song *et al.* assumes the prediction of the auxiliary classifier q_ϕ is consistent with the ground truth and searches for inputs of the generator z^* such that they results in the images that the auxiliary classifier recognizes as an instance of y_{source} but the targeted model F recognizes it as an instance of y_{target}. However, this approach cannot find points that are adversarial to both q_ϕ and F. In Section 4, we show that, in fact, the decision boundaries of these two networks are close to each other. The similarity of the decision boundaries means that Song *et al.* attack does not consider most of the target model's adversarial points. Additionally, adversarial examples generated by Song *et al.* have limited transferability.

Here, we go over the proposed method to generate adversarial examples by using non-conditional GANs. In summary, our method consists of training separate GAN for each class of the data and searching on the input noise of the generator in order to find an image that is misclassified by the target classifier.

The following section goes into more detail on how to train GANs, find adversarial examples in white-box and black-box settings and good hyper-parameters foe the

attack.

3.3.1 Overview

One of the main challenges in the generation of unrestricted adversarial examples is to know the actual class of the samples. Perturbation-based adversarial attacks assume that the sample's true label is not affected if the perturbations are small. However, determining the true label of the samples is not trivial in unrestricted adversarial attacks.

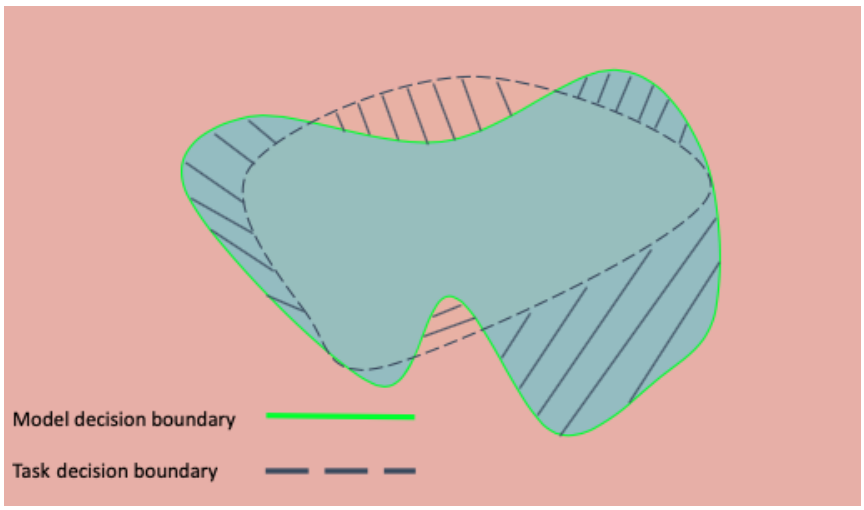


Figure 3.1: This figure represents a simple binary classification task. Unrestricted adversarial attacks look for any data point in the hashed area where the target model's decision does not agree with the true label.

Previously, Song *et al.* proposed an unrestricted attack that assumes the actual label of the generated data points is the same as the given labels to the AC-GANs. Their human studies showed that this assumption holds in most of the cases. However, as our analysis in section 2.2.3 shows, AC-GANs are biased towards down sampling the points near the decision boundary of the auxiliary classifier and hence, ignore a large potential adversarial subspace. This bias also encourages the Song *et al.* attack to generate adversarial examples that are correctly classified by the auxiliary classifier and hence, do not transfer to the auxiliary classifier. Inspired by the Song

et al. attack, we propose a method to produce unrestricted adversarial examples using GANs. Hence, as an alternative approach, we propose training non-conditional GANs with a gradient penalty for each class of data. In this way, the attacker has one generator for each class.

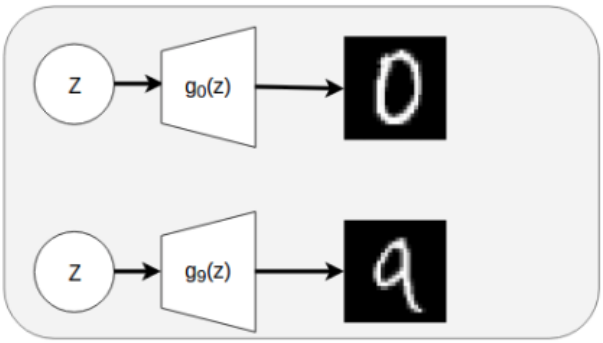


Figure 3.2: Using non-conditional GANs to generate class-conditional data points.

This approach for generating class-conditional images does not suffer from the bias discussed in the previous chapter. Hence, we hypothesize that our attack has access to a larger adversarial subspace.

One of the main challenges of training separate generators for each class of data is the instability of training of GANs, especially when the number of samples is small. In Section 3.3.2, we discuss the steps taken to stabilize the training process.

The next step in the proposed attack is to search for the data points, where the target classifier's decision does not agree with the class-conditional generator. Section 3.3.3 discusses the search algorithm in details. Finally, Section 3.3.4 provides the information on how to set good hyper-parameters for the attack.

3.3.2 Training the Generative Models

Training generative adversarial networks are known to be unstable, and many techniques have been proposed to stabilize the training process and improve the quality of generated data[27]. Limiting the training data also further destabilizes the training. To overcome this challenge, we take various steps.

As proposed in [27], penalizing the norm of the gradient of the discriminator with respect to its input would improve the convergence of training of WGANs. We find that applying the gradient penalty improves the quality of the generated class-conditional data in our experiments.

To further improve the training process on the limited data, we take advantage of fine-tuning. Fine-tuning is the process of taking advantage of model that has been trained on similar tasks. In fine-tuning, the weights of the pre-trained model is used for the initialization of the weights of the current model for another task. We stabilized the training by pre-training a GAN on the complete dataset and using it as the starting point for each class-conditional GAN training.

As was discussed in the previous chapter, the discriminator is a neural network that performs a binary classification and classifies each input as the real data point or samples generated by the generator. Since we train each generator and discriminator on a specific class of data, we saw feeding the images of other classes to the discriminator as the fake images during the training improve the convergence of the models and the quality of the generated images. More formally, during the training, the generator is trained to generate images indistinguishable by the discriminator from real samples of class y. To encourage the generator to produce better images that do not belong to other classes, we also add $-D(x_{y'})$ to the loss of our discriminator where for any class of data x_y, $x_{y'}$ is a data point with the label other than y.

3.3.3 Generating the Samples

3.3.3.1 White-box Settings

Let $g_{y,\theta}(z)$ be the generator of the source class. With our approach, this generator is a GAN that is trained only on samples of the data set with the label of y.

Our approach finds latent point z^* that $F(g_{y,\theta}(z^*)) \neq y$. The search for z^* is done

by minimizing an objective

$$\mathcal{L} = \mathcal{L}_0 + \lambda \mathcal{L}_1 \tag{3.1}$$

where λ is weighting hyperparameter. The first component of the loss is defined as:

$$\mathcal{L}_0 \triangleq -\log f\left(y_{\text{ target}} \left| g_{y,\theta}\left(z\right)\right.\right) \tag{3.2}$$

where f(.) is the target model, this component encourages the generated samples to be adversarial. The second part is defined as:

$$\mathcal{L}_1 \triangleq \frac{1}{m} \sum_{i=1}^{m} \max\left\{\left|z_i - z_i^0\right| - \epsilon, 0\right\} \tag{3.3}$$

where $z \in R^m, z_i^0 \ \left\{z_1^0, z_2^0, \cdots, z_m^0\right\} \overset{\text{i.i.d}}{\sim} \mathcal{N}(0,1)$. We found that this component helps the attack to produce adversarial examples that are more similar to the natural samples of the source class.

The search for adversarial examples is done by minimizing the Equation 3.2. As is shown in Algorithm 1, we use T step gradient descent. T is the hyper-parameter that specifies the the time the algorithms spend for generating each sample.

As this approach does not use any neural network to determine the ground truth, adversarial examples generated with our proposed method are not limited inside the decision boundary of the auxiliary classifier. This feature enables our method to have access to more points inside adversarial space. Additionally, adversarial examples generated with our method have high transferability.

3.3.3.2 Black-box Settings

If the adversary does not have access to the targeted model's parameters, it can perform the black-box variant of our unrestricted attack. We assume the adversary has access to another model trained on the same data distribution as the target

Algorithm 1 White-box Targeted Attack

```
procedure ATTACK(f, g_{source,θ}, y_{target}, λ, ϵ, α, T)
    Define L based on Equation 3.2
    Initialize x_attack ← ∅
    while x_attack = ∅ do
        Sample z^0 ~ N(0,1)
        Initialize z ← z^0
        for i = 1...T do
            x ← g_{source,θ}(z)
            if f(x) ≠ y_{target} then
                Update z ← z − α · ∂L/∂z
            else
                x_{attack} ← x
            end if
        end for
    end while
    return x_{attack}
end procedure
```

model. With this assumption, the adversary can train their model and use the transferability of adversarial examples. As we show in Chapter 4, our attack has much higher transferability than similar works. Hence, the adversary can generate adversarial examples in harder threat models. Here, we assume the adversary only has access to the top-1 prediction of the target model, which is one of the hardest threat models studied in the literature.

3.3.4 Hyper-parameter Selection

In this section, we emphasize the importance of some of the hyper-parameters of the proposed attack. During the experiments, we noticed how the tuning of some hyperparameters could increase the quality of the adversarial examples or speed up the process. The important hyper-parameters to consider are as follows:

- λ: λ is the weighting hyper-parameter for the second term of the loss of the attack. The second term of the loss measures how far the noise is from the naturally sampled point. When the λ is too small, the generator starts to

Algorithm 2 Black-box Targeted Attack

```
1: procedure ATTACK(f_target, f_attacker, g_source,θ, y_target, λ, ϵ, α, T)
2:     Define L based on Equation 3.2 and use f_attacker
3:     Initialize x_attack ← ∅
4:     while x_attack = ∅ do
5:         Sample z^0 ~ N(0, 1)
6:         Initialize z ← z^0
7:         for i = 1 ... T do
8:             x ← g_source,θ (z)
9:             if f_target(x) ≠ y_target then
10:                Update z ← z − α · ∂L/∂z
11:            else
12:                x_attack ← x
13:            end if
14:        end for
15:    end while
16:    return x_attack
17: end procedure
```

generate adversarial examples with ambiguous labels. Moreover, if the λ is too large, the attack gets extremely slow or does not find any adversarial examples. We noticed that the choice of $\lambda = 0.1$ was a good choice for MNIST and SVHN datasets.

- ϵ: Similar to λ, ϵ is a hyper-parameter that determines the sensitivity of the loss to the distance of the input of the generator to the naturally sampled data point. High ϵ results in low-quality adversarial examples. We found that $\epsilon = 0.05$ is a good choice for MNIST and SVHN datasets.

- Maximum number of gradient descent update for the adversarial attack: By setting the maximum number of gradient descent update for our adversarial attack, we prevent the algorithm from taking a very long time for some random noise. During the experiments, we realized that the generator has difficulties generating any adversarial example even with thousands of gradient updates; hence, choosing another noise would speed up the process. Experimentally, we found 500 is a good choice for this hyper-parameter. However, finding the best

number and using it to measure the robustness of the models is left for the future works.

3.4 Concluding Remarks

Without a reasonable estimation of the adversarial robustness of machine learning models, measuring the advances in securing machine learning models is difficult. A robust model has to be at least secure against well-known perturbation-based attacks, adaptive adversaries, and unrestricted adversarial attacks. This chapter went over details of fast, GPU-accelerated, and easy-to-use libraries that provide the tool for the researchers to approximate an upper level of adversarial robustness of their models.

Additionally, this chapter discussed the generation of unrestricted adversarial examples using generative adversarial networks in depth. In earlier chapters, we formally analyzed the bias in AC-GANs towards down-sampling near the discriminator's decision boundaries. This bias leads to ignoring large adversarial subspace in the unrestricted attack using AC-GANs. Our analysis also explains the lack of transferability of adversarial generated using previously proposed techniques. Here, we proposed a new approach to generated class-conditional samples that do not suffer from the discussed bias. We proposed white-box and black-box algorithms to generated unrestricted adversarial examples for defenses without using AC-GANs. Furthermore, we proposed using techniques to stabilize the training process of GANs on the fraction of the data sets.

In order to experimentally demonstrate our formal analysis, we show the improvement in access to adversarial subspace and transferability of our adversarial attack in the next chapter.

Chapter 4

Experimental Results

4.1 Overview

This chapter is organized into three sections. First, this chapter goes over the experiments that helped us design our proposed unrestricted attack. These experiments include the analysis of the decision boundaries of Auxiliary classifiers in AC-GANs used by the previous attacks. Furthermore, we evaluate the increase in the transferability of adversarial examples generated by our approach compared to Song *et al.* attack. The results of these experiments support our theoretical analysis and demonstrate the need for our attack. Additionally, this section includes the discussion on why each of the experiments was performed and what the results of each experiment represent.

In the second section, we focus on evading some of the recent adversarial defenses using TorchAE, the adversarial robustness evaluation tool we developed. The results in this category of experiments demonstrate the effectiveness of the proposed tool and emphasize the need for better adversarial robustness evaluation.

In the third section, we use our unrestricted attack to bypass state-of-the-art adversarial defenses. The results in this category of experiments demonstrate the effec-

tiveness of our attack. The ground truth label of unrestricted adversarial examples is hard to find. Hence, this section includes the results of the human-labelling experiments. By performing human-labelling, we measure the success rate of the proposed unrestricted attack.

4.2 Design Experiments

4.2.1 Decision Boundary Similarity Analysis

In this section, we demonstrate the shortcomings of Song*et al.* unrestricted attack. By showing the similarity of decision boundaries of Auxiliary classifiers in AC-GANs, paired with our analysis in Chapter 2, we conclude that Song *et al.* attack ignores a large adversarial subspace. Additionally, these experiments explain the lack of transferability in Adversarial examples generated by Song *et al.*

To compare the work by Song *et al.* with our proposed attack, we show the shortcomings of their attack. Based on our analysis in Section 2.2.3, the generator of AC-GANs are biased towards generating samples that lie far inside the decision boundary of the auxiliary classifier; hence, biased towards finding adversarial examples that do not transfer between the auxiliary classifier and the target model. By showing that a large adversarial sub-space is shared between the auxiliary classifier of an AC-GANs and baseline CNNs, we demonstrate that generating unrestricted adversarial examples with AC-GANs ignores many potential adversarial examples.

We use the method suggested by Tramer *et al.* to analyze the similarity of the decision boundaries of the auxiliary classifier of the AC-GANs and baseline CNNs. Tramer *et al.* approach uses the adversarial direction which for model f, a data point x and an adversarial example x' is defined as: $\boldsymbol{d}(f, \boldsymbol{x}) := \frac{\boldsymbol{x}'-\boldsymbol{x}}{\|\boldsymbol{x}'-\boldsymbol{x}\|_2}$. We use l_2 FGM to generated adversarial examples. Minimum distance is defined as the distance to the

Table 4.1: Architecture for baseline MNIST CNNs.

Layer Type	CNN 1	CNN 2
Conv + ReLU	5*5*64	5*5*20
Maxpooling	2*2	2*2
Conv + ReLU	5*5*64	5*5*50
Dropout	0.25	N/A
FC1	128	500
Dropout	0.25	N/A
FC2	10	10

nearest adversarial example in this direction:

$$MIN\text{-}DIST_d(f, \boldsymbol{x}) := \arg\min_{\varepsilon>0} f(\boldsymbol{x} + \varepsilon \cdot \boldsymbol{d}) \neq f(\boldsymbol{x}) \tag{4.1}$$

Given the direction calculated for one model, the inter-boundary distance for the two models is defined by:

$$\begin{aligned} &\text{INTER-DIST}_d\left(f_1, f_2, \boldsymbol{x}\right) := \\ &\left|\text{MIN-DIST}_d\left(f_1, \boldsymbol{x}\right) - \text{MIN-DIST}_d\left(f_2, \boldsymbol{x}\right)\right| \end{aligned} \tag{4.2}$$

Table 4.2: Architecture for the discriminator of the AC-GAN.

Layer Type	Discriminator
Conv + Leaky ReLU	3*3*16, stride(2*2)
Dropout	0.25
Conv + Leaky ReLU	3*3*32, stride(2*2)
Dropout + BN	0.25
Conv + Leaky ReLU	3*3*64, stride(2*2)
Dropout + BN	0.25
Conv + Leaky ReLU	3*3*128, stride(2*2)
Sigmoid	1
Auxiliary layer: Softmax	10

The adversarial examples are generated using l_2 FGM, similar to the original work. The results demonstrate that the intra-boundary distance of the model is smaller than the minimum distance from the test input. These results show that an adversary can transfer the adversarial example to the auxiliary classifier by slightly increasing

the size of perturbation in the adversarial direction, supporting the similarity of decision boundaries of the auxiliary classifier with the baseline CNNs.

Figure 4.1 contains the results of our experiments on the similarity of the decision boundary of the auxiliary classifier used by Song *et al.* and the baseline CNNs for MNIST. We use Tramer *et al.* [78] approach to estimate the intra-boundary distance in the adversarial direction.

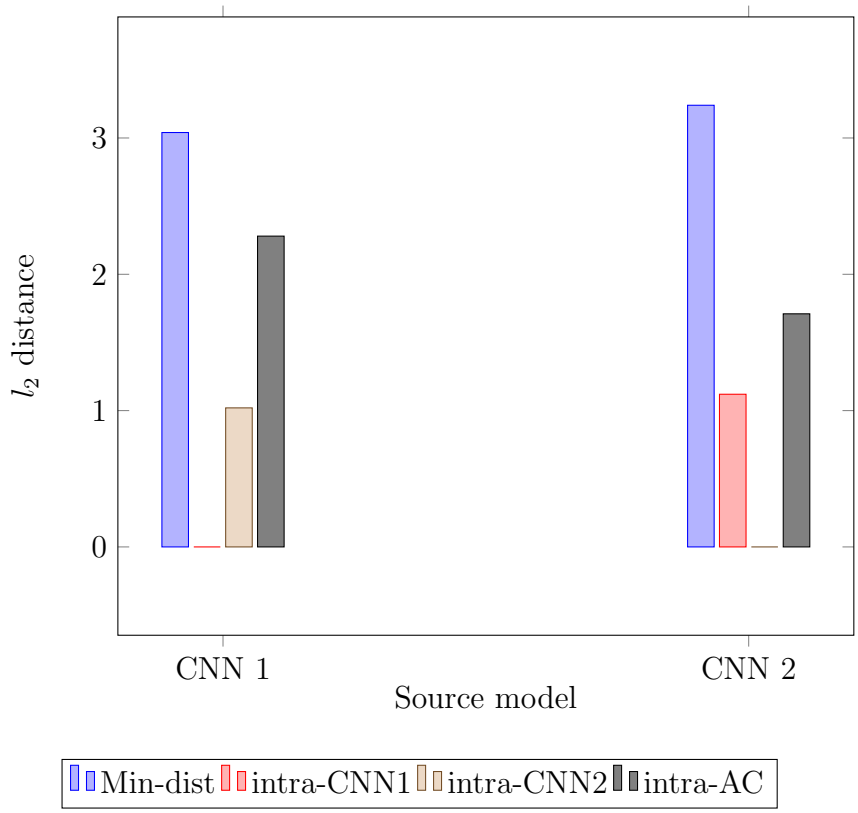


Figure 4.1: Minimum distances and inter-boundary distances in the adversarial directions for MNIST models. The blue bar shows the minimum distance to the decision boundary in the adversarial direction. Adversarial attack is done using l_2 FGM with $\epsilon = 5$. Other bars show the distance between the source model's decision boundary and the boundaries of other models.

Additionally, Song *et al.* attack assumed that the AC-GANs generates the samples with the correct ground truth. Considering the bias in AC-GANs, we can conclude that Song *et al.* attack is using the decision boundary of the auxiliary classifier as the ground truth. However, the auxiliary classifier performs poorly on the MNIST

test set with only 84.6% accuracy. The poor performance of the auxiliary classifier supports that the auxiliary classifier's decision boundaries are not well-suited to be used as the ground truth.

4.2.2 Transferability

The transferability of adversarial examples between machine learning models is an important property that enables attackers to fool machine learning models with minimal knowledge and effort. By having an attack that produces transferable adversarial examples, the attackers can generate adversarial examples once for their models and reuse it on other models, even in black-box settings. The mathematical analysis and experimental results of the previous sections show that AC-GANs are biased towards generating samples that fall far inside the auxiliary classifiers' decision boundary. Additionally, the decision boundaries of auxiliary classifiers are highly similar to widely used baseline classifiers. Hence, we can explain the low transferability of the Song *et al.* unrestricted attack that uses AC-GANs.

Furthermore, Table 4.3 demonstrates the high transferability of adversarial examples generated for baseline CNNs and the auxiliary classifier, which further supports the existence of a sizeable adversarial subspace shared between the auxiliary classifier and the target models.

Table 4.3: Transferability of adversarial examples between auxiliary classifiers and baseline CNNs. Simple l_2 FGM attack with $\epsilon = 5$ was used for the consistency with the [78]. The numbers represent accuracy.

Attack Type	Network		
	AC	CNN 1	CNN 2
No attack	84.6	99.3	97.6
source CNN2	21.78	9.91	2.12
source CNN1	30.46	9.14	20.80

Instead of using AC-GANs that learn a biased distribution [65] and ignores a large part of the adversarial subspace, we train a WGAN-GP for each class in our attack.

Having access to a larger adversarial subspace makes our attack stronger than previous unrestricted attacks. Furthermore, by not using any auxiliary classifier, our attack produces more transferable adversarial examples. Table 4.4 demonstrates the transferability of the adversarial examples generated by our attack. In comparison, while our attack achieves 71% success rate for transferring adversarial examples to the baseline CNN 1, Song *et al.* reported 4.9% success for transferring adversarial examples to the baseline CNN 1.

Table 4.4: Transferability of adversarial examples generated by our unrestricted attack. Based on 100 untargeted adversarial examples that are generated with the source 6. We performed the 500 step search using Adam optimizer with learning 0.001, $\lambda = 0.7$, and $\epsilon = 0.05$. The numbers represent the error rate.

Attack Source	Network	
	CNN 1	CNN 2
No attack	0.7	2.4
CNN2	71	100
CNN1	100	53

Table 4.5: Architecture for the generator of the AC-GAN.

Layer Type	Generator
z	100
Linear	8092
BN + Upsample	
Conv + Leaky ReLU	3*3*128
BN + Upsample	
Conv + Leaky ReLU	3*3*64
BN	
Conv + Tanh	3*3*1

4.3 Evaluation Results

4.3.1 Conducting Robustness Evaluation

More than 200 defenses against adversarial examples have been proposed in the literature in recent years. As was shown previously, many of these defenses are vulnerable

to strong adaptive attacks [8, 11, 2, 9]. This section evaluates the adversarial robustness of three recently proposed defense in the literature against strong adaptive adversaries.

4.3.1.1 Defensive Dropout

Dropout [70] is a regularizer technique that involves setting the output of some hidden units of the neoork to zero randomly. It has been demonstrated that using dropout at the training time reduces the amount of over-fitting substantially. Wang *et al.* suggest using Dropout at the inference time increases the adversarial robustness of the network without adding any extra computation [83]. This defense can be categorized as defenses that use randomization. We observed that the authors of Defensive Dropout do not perform adaptive attacks in their white-box evaluation. To perform this task, we used ensemble over randomness, similar to the process that Carlini and Wagner used to bypass Feinman *et al.* defense [9, 19]. The adversarial example generation was done using CW attack with the average loss over 50 outputs:

$$L'\left(x'\right) = \sum_{j=1}^{K} L_j\left(x'\right).$$

With this approach, we were able to generate targeted adversarial examples for 98% of MNIST test set with mean l_2 distortion of 1.75, compared to 1.64 for the unsecured network. Considering the drop in the accuracy of the model that defensive dropout causes, these results demonstrate that Defensive Dropout provides limited benefit against worst-case adversaries.

4.3.1.2 Random Self-ensemble

Random self-ensemble [41] is a defensive technique that adds extra layers of noise inside the neural network. To preserve accuracy, the authors suggest using the average

output of an ensemble of models. However, we observe that their evaluation lacks performing defense aware white-box attacks by taking the average of the outputs at the time of adversarial example generation. This flaw in their evaluation results in the illusional adversarial robustness. Since they claimed robustness for Cifar-10 classification, we trained a network with the same parameters as they reported. We used ensemble over randomness for generating adversarial examples using the CW attack. With this technique, we were able to generate targeted adversarial examples for 94% of the first 1000 Cifar-10 test set with the mean distortion of 0.45. The other flaw in their evaluation is that instead of using the binary search for parameter c of CW attack and finding minimally distorted adversarial examples, they report the results for different values of c, making the comparison difficult. However, our results show that an adversary can produce visually imperceptible adversarial examples for this defense.

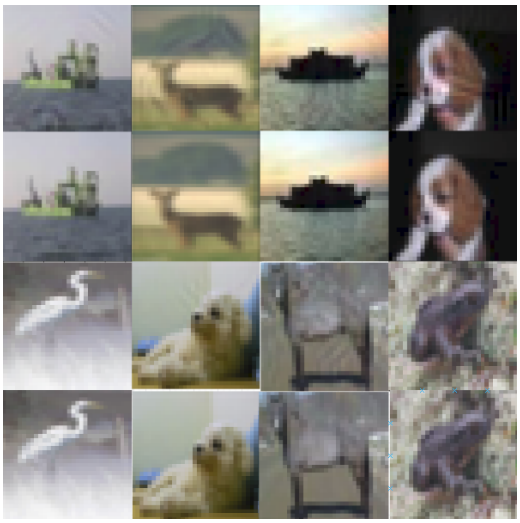


Figure 4.2: Targeted adversarial examples for Random self-ensemble compared to the original images. The top images are adversarial.

4.3.1.3 Deep Detector

Deep detector can be categorized as a defensive pre-processing defense and applies spatial smoothing and quantization adaptively. Based on the entropy of the image,

Entropy	Quantization Intervals	Smoothing?
<4	2	No
4∼ 5	4	No
>5	6	Yes

Table 4.6: Denoisening Strategy of Deep Detector

Deep Detector applies these pre-processing steps to remove the image's potential noise. If the model's prediction is changed after applying these steps, it detects the sample as an adversarial [40]. More details about Deep detector can be found on Table 4.6 and Figure 4.3.

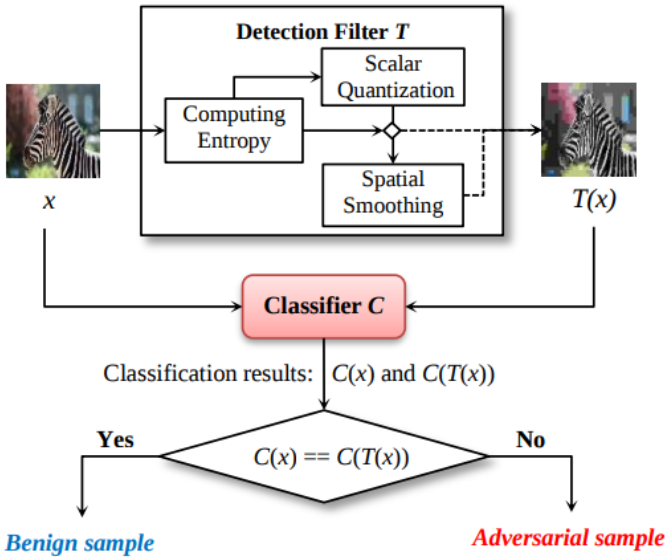


Figure 4.3: Detection strategy of Deep Detector. Adopted from the original paper[40].

The authors of the Deep detector performs defense-aware adversarial evaluation by using CW attack to generate adversarial examples and remove the examples the attacker thinks that would be detected. While we recognize their effort in performing the defense-aware evaluation, we propose a more robust defense aware attack that can bypass the Deep Detector. We propose adding a regularization term to the CW loss that encourages the adversarial example to be classified the same as the denoised version of itself. We perform the CW attack with the following loss function:

$$l'\left(x'\right) = l\left(f\left(x'\right), t\right) + c_1 \cdot l\left(f\left(x'\right), f\left(C\left(x'\right)\right)\right) + c_2 \cdot \left\|x' - x\right\|_2^2,$$

where $C(x)$ is Deep Detector filter. We used binary search for c_2 and c_1. We were able to generate successful targeted adversarial example for 96% of MNIST test set with mean distortion of 2.5 compared to 3.86 for only applying quantization with two intervals (binirization). Binirization is one of the baseline defenses for the MNIST dataset where the value of each pixel is set to the closest integer (0 or 1). [29]. Some adversarial examples generated for Deep detector is shown in Figure 4.4.

Figure 4.4: A sample of targeted adversarial examples for Deep detector.

4.3.2 Bypassing the State-of-the-art

In this section, we provide the experimental results of our new proposed unrestricted attack. To evaluate our attack's power, we choose two of the most strong state-of-the-art defenses, binary ABS and adversarial training, as the target model. Furthermore, we choose one of the hardest threat models used in the literature for our evaluation. In this threat model, the adversary does not have access to the weights of the mode and cannot calculates the gradients. Additionally, the adversary does not have access to the model's logits and only gets the top-1 prediction of the model.

4.3.2.1 ABS Defence

As was discussed in Chapter 2, Analysis by Synthesis proposed by Schott *et al.* is a Bayesian classifier based on VAEs that provides the state-of-the-art adversarial robustness on MNIST datasets. Like our proposed attack, their defense works by training separate generative models for each class of data. ABS defense calculates the likelihood of the input belonging to each class by checking the presence of all the class specific features learned by the VAEs in the input of the classifier.

To demonstrate the power of our attack, we perform an untargeted black-box attack with only having access to top-1 prediction to the ABS model [62], the state-of-the-art defence for MNIST dataset. We use our attack to generate adversarial examples for baseline CNN 2 and transfer them to the ABS model. Figure 4.5 contains the first ten untargeted adversarial examples generated by our attack with the source class of 0 to 9. Our method can bypass ABS defence with very little knowledge, while even adaptive white-box adversaries cannot reduce its accuracy significantly [62]. This results demonstrate that even the strongest defense is vulnerable to our adversarial attack in the hardest threat model.

Figure 4.5: Samples of untargeted adversarial examples generated by the proposed attack for the ABS defense with only top-1 prediction. The source classes are from 0 to 9. We performed the 500 step using Adam optimizer with learning 0.001, $\lambda = 0.7$, and $\epsilon = 0.05$. The red squares indicate failure in generating adversarial examples as decided by human labeling.

4.3.2.2 Adversarial Training Defense

Adversarial training is another defence mechanism that we use to evaluate our method. While adversarial training does not claim general robustness, it is one of the few defences in the literature that remains unbreakable in the defined threat model. Here, we show that our unrestricted attack can bypass Madry *et al.* defence in the general setting. Figure 4.6 contains adversarial examples generated with our proposed method against an adversarially trained MNIST classifier.

Training GANs is known to be unstable [60]. Since our method uses fewer samples to train each GAN, we expect that our method's success rate decreases as the data

Table 4.7: Architecture for the generator of the WGAN-GP.

Layer Type	Generator	BN?
z	100	
Transposed Conv + ReLU	4*4*1024	Yes
Transposed Conv + ReLU	4*4*512	Yes
Conv + Leaky ReLU	4*4*256	Yes
Conv + Tanh	4*4*1	No

Table 4.8: Model parameters.

Parameter	CNN 1	CNN 2	AC-GAN	WGAN-GP
Epoch	10	5	200	200
Optimizer	Adam	SGD	Adam	Adam
Learning rate	1e-4	1e-2	2e-4	1e-4
Momentum		0.5		
Batch size	32	32	64	64

become more complex. To evaluate our algorithm on more complex data, we use the fashion-mnist and Street View House Numbers (SVHN) Dataset. Fashion-mnist consists of MNIST-like 28×28 gray-scale images containing fashion products from 10 categories [86]. SVHN consists of coloured real-world images that are significantly harder to generate and to classify than MNIST [48]. To make the training more convergent, we started the process of training each GAN from a pre-trained GAN that was trained using all the available data. We used our algorithm to generate black-box adversarial examples for l_∞ adversarially trained SVHN and fashion-mnist classifiers. The ϵ for fashion-mnist is 0.1, and for SVHN is 0.02. Samples of generated adversarial examples are available in the Figures 4.8 and 4.7.

4.3.3 Human Labeling

In this section, we present the results of human-labelling used to show the success of our attack. We use three humans to assign a ground truth to adversarial examples generated for ABS defense and adversarially trained classifiers for MNIST and SVHN datasets. Since humans only achieve 83.5% accuracy on the fashion-mnist dataset,

we do not use human labelling on fashion-mnist adversarial examples. Each human in this experiment is presented with 1000 adversarial examples for each classifier, 60 for each class. Human-labeller is asked to provide the ground truth label for image or N/A if it belongs to none of the classes.

Figure 4.7: Randomly selected adversarial examples generated by untargeted white-box adversarial attack against adversarially trained fashion-mnist classifier. The defense was trained using PGD with $\epsilon = 0.1$. We performed the 500 step using Adam optimizer with learning 0.001, $\lambda = 10.0$, and $\epsilon = 0.02$.

For each of the binary ABS and adversarially trained classifiers, we generated 100 adversarial examples per class for CNN 1 that can be transferred to the target classifier. Three humans were asked to specify the ground truth for these images. After taking the majority vote, images with the ground truth different than the classifier's prediction are considered successful adversarial examples. The results show that for ABS model 73.8% of images, and for the adversarially trained model

85.7% of generated images are adversarial.

Table 4.9: Success rate of our attack demonstrated by human labeling

	Model		
Source	ABS	Madry	SVHN
0	74	81	64
1	75	88	65
2	74	84	75
3	85	92	69
4	59	74	80
5	70	88	96
6	92	93	61
7	71	84	81
8	71	87	74
9	68	86	62
Overall	73.8	85.7	72.1

For PGD adversarially trained SVHN classifier, we generated 100 adversarial examples per class by using Algorithm 1. The results of the success rate for each source class are presented in Table 4.9. As expected, since SVHN is more complicated than MNIST, and we only use a fraction of the available data to train our GANs, the success rate on the SVHN dataset is lower.

Table 4.10: Comparison with the Song et al. algorithm

	Ours	Song *et al.*
Madry *et al.*	85.7	85.2
adv-trained SVHN	73.8	84.2

4.4 Concluding Remarks

The experiment's results of this chapter demonstrate the following points:

- The decision boundary of the auxiliary classifier in AC-GANs is highly similar to the baseline-classifier. This finding paired with the known bias in AC-GANs in down-sampling samples near decision boundary of the generator demonstrates that previous unrestricted attacks ignore large adversarial subspace.

- Not using AC-GANs in our attack significantly increases the transferability of our attack compared to previous attacks.

- Our unrestricted attacks bypasses the state-of-the-art defenses in hardest threat model in the literature. This results demonstrate the effectiveness of our attack. Additionally, these results shows that even strongest defenses are vulnerable for easiest datasets (e.g. MNIST).

- We showed the use cases of our proposed tool by evaluating three of the recently proposed defenses. Our evaluation shows that these attacks are not robust against adaptive adversaries. These defenses have fallen into the common pitfalls pointed by the Carlini *et al.* [8]. These results emphasizes on the importance of following best practices for better adversarial robustness evaluations and the need for our proposed tool.

Figure 4.6: Untargeted white-box adversarial attack against adversarially trained MNIST classifier. Examples generated by the proposed attack. The defense was trained using PGD with $\epsilon = 0.3$. The source classes are from 0 to 9. We performed the 500 step using Adam optimizer with learning 0.001, $\lambda = 0.7$, and $\epsilon = 0.05$. The red squares indicate failure in generating adversarial examples as decided by human labeling.

Figure 4.8: Untargeted white-box adversarial attack against adversarially trained SVHN classifier. Examples generated by the proposed attack. The defense was trained using PGD with $\epsilon = 0.03$. The source classes are from 0 to 9. We performed the 500 step using Adam optimizer with learning 0.001, $\lambda = 0.7$, and $\epsilon = 0.05$.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

Adversarial examples are the results of the vulnerabilities of machine learning models to the adversaries' attacks at the test times. In recent years, machine learning models have become crucial parts of most software systems that are affecting human lives. Hence, reducing the adversarial vulnerability of machine learning models by changing the training procedure or adding extra layers of processing has become a research interest for many machine learning and cybersecurity researchers.

In this work, we investigated how to measure the adversarial robustness of deep learning models. One of the most common approaches for measuring the robustness of models and defenses is measuring the success of different adversarial attacks in different threat models. However, one can easily assume delusional high robustness by not performing comprehensive and adaptive attacks [8]. It is also common to report unrelated numbers when comparing a proposed defense with related works. The unrelated numbers can be the result of the diversity of threat models and attack's hyper-parameters. To help researchers in overcoming these difficulties, we developed AETorch. AETorch is a tool with most common attacks and threat models and is

implemented in Pytorch to provide GPU acceleration. In this work, we showed how AETorch could measure the adversarial robustness of some of the recently proposed defenses. The lack of real robustness of these defenses emphasizes the importance of careful experiments of newly proposed defenses. A standard framework for adversarial robustness evaluation can accelerate the progress of defending against adversarial examples. We take a step by developing a tool that can perform strong attacks in different threat models and perform some of the necessary sanity tests based on suggestions of Carlini et al. [8]. This tool does not eliminate the process of coming up with the worst-case adaptive adversary but helps the researchers compare their model with previous works. For the future work, we intend to keep the tool updated with new attacks and adds some advanced features like attack bundling.

Furthermore, our analytical and experimental investigations showed the bias in previous unrestricted adversarial attacks. This bias yields to lack of transferability, as well as not having access to a substantial adversarial subspace of the deep learning models. Based on our analysis, we proposed a new unrestricted adversarial attack that improves on Song et al., the previous state-of-the-art unrestricted attack. Our experiment showed that this strong unrestricted attack bypasses the most robust defenses for standard datasets on the hardest threat models.

Table 4.10 compares the success rate of our attack with the previous unrestricted approach. However, we argue that our attack can effectively measure the general robustness of defenses on MNIST-like datasets. We demonstrate that the problem of training an adversarially robust model in MNIST is far from being solved. This finding is aligned with Tramer and Boneh's suggestion that MNIST should not be abandoned or considered solved yet [74]. While the proposed attack requires the adversary's access to the data sampled from the same distribution as the training data of the target model, it can successfully generate adversarial examples in one the most limited threat models for the state-of-the-art general defense for MNIST.

Additionally, the human-labelling proves that the generated adversarial samples have a high success rate. We acknowledge that this attack does not scale to the more massive datasets (e.g., ImageNet). We hope that our attack helps the community to study adversarial examples outside the restricted threat models.

5.2 Future Work

The findings of this thesis can be further studied in multiple directions. By developing AETorch, we are taking the first step towards providing a framework that can measure the robustness of ML models and provide comparable results. However, AETorch lacks the adaptive adversarial attack based on the specification of each model. While providing all the scenarios for the adaptive attack for each defense model is infeasible, it is beneficial to the community if AETorch includes some common scenarios for the next steps. Additionally, maintaining the AETorch with new attacks and adversarial threat models is crucial for providing the best estimation of the adversarial robustness.

While this thesis identifies some of the shortcomings of generating unrestricted adversarial using AC-GANs, we recognize that further human-study is required. A more extensive human-labelling on wider variety of GANs can be the next step for helping the community fully understand the characterization of unrestricted adversarial examples generated by GANs.

Developing better generative models is an active line of research. With the success of GANs, most research has been focusing on variants of GANs in recent years. While GANs produce great quality samples, they lack the desired training stability [60]. Recently, other types of generative models have considerably improved in the quality of samples [59, 82]. Hence, studying the generation of unrestricted adversarial examples with other types of generative models is an open research direction.

Every sample found with our unrestricted attack or previous approach is adversarial by definition. However, there are no metrics to measure the amount of work that has been done to find each adversarial example. Hence, one of the challenges that we faced for comparing unrestricted attacks was the lack of a metric that can be used to compare defenses' strengths. One can use the number of gradient steps as a metric, but this number depends on many other hyper-parameters. Studying a new metric to compare the strengths of adversarial defenses against unrestricted attacks is another research direction for future works.

Bibliography

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou, *Wasserstein gan*, arXiv preprint arXiv:1701.07875 (2017).

[2] Anish Athalye, Nicholas Carlini, and David Wagner, *Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples*, arXiv preprint arXiv:1802.00420 (2018).

[3] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok, *Synthesizing robust adversarial examples*, arXiv preprint arXiv:1707.07397 (2017).

[4] Amir Beck and Marc Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM journal on imaging sciences **2** (2009), no. 1, 183–202.

[5] Wieland Brendel, Jonas Rauber, and Matthias Bethge, *Decision-based adversarial attacks: Reliable attacks against black-box machine learning models*, arXiv preprint arXiv:1712.04248 (2017).

[6] Tom B Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer, *Adversarial patch*, arXiv preprint arXiv:1712.09665 (2017).

[7] Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow, *Thermometer encoding: One hot way to resist adversarial examples*, (2018).

[8] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, and Aleksander Madry, *On evaluating adversarial robustness*, arXiv preprint arXiv:1902.06705 (2019).

[9] Nicholas Carlini and David Wagner, *Adversarial examples are not easily detected: Bypassing ten detection methods*, Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, ACM, 2017, pp. 3–14.

[10] _____, *Magnet and" efficient defenses against adversarial attacks" are not robust to adversarial examples*, arXiv preprint arXiv:1711.08478 (2017).

[11] _____, *Towards evaluating the robustness of neural networks*, 2017 IEEE Symposium on Security and Privacy (SP), IEEE, 2017, pp. 39–57.

[12] _____, *Audio adversarial examples: Targeted attacks on speech-to-text*, (2018), 1–7.

[13] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh, *Ead: elastic-net attacks to deep neural networks via adversarial examples*, Thirty-second AAAI conference on artificial intelligence, 2018.

[14] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier, *Parseval networks: Improving robustness to adversarial examples*, Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, 2017, pp. 854–863.

[15] Francesco Croce and Matthias Hein, *Provable robustness against all adversarial l_p-perturbations for $p \geq 1$*, May 2020.

[16] Guneet S Dhillon, Kamyar Azizzadenesheli, Zachary C Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Anima Anandkumar, *Stochastic activation pruning for robust adversarial defense*, arXiv preprint arXiv:1803.01442 (2018).

[17] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry, *A rotation and a translation suffice: Fooling cnns with simple transformations*, arXiv preprint arXiv:1712.02779 (2017).

[18] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song, *Robust physical-world attacks on deep learning models*, arXiv preprint arXiv:1707.08945 (2017).

[19] Reuben Feinman, Ryan R Curtin, Saurabh Shintre, and Andrew B Gardner, *Detecting adversarial samples from artifacts*, arXiv preprint arXiv:1703.00410 (2017).

[20] Nic Ford, Justin Gilmer, Nicolas Carlini, and Dogus Cubuk, *Adversarial examples are a natural consequence of test error in noise*, arXiv preprint arXiv:1901.10513 (2019).

[21] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, 2016.

[22] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, *Generative adversarial nets*, Advances in neural information processing systems, 2014, pp. 2672–2680.

[23] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy, *Explaining and harnessing adversarial examples*, International Conference on Learning Representations, 2015.

[24] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy, *Explaining and harnessing adversarial examples*, arXiv preprint arXiv:1412.6572 (2014).

[25] Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli, *On the effectiveness of interval bound propagation for training verifiably robust models*, arXiv preprint arXiv:1810.12715 (2018).

[26] Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel, *Adversarial perturbations against deep neural networks for malware classification*, arXiv preprint arXiv:1606.04435 (2016).

[27] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville, *Improved training of wasserstein gans*, Advances in Neural Information Processing Systems, 2017, pp. 5767–5777.

[28] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten, *Countering adversarial images using input transformations*, arXiv preprint arXiv:1711.00117 (2017).

[29] Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song, *Adversarial example defense: Ensembles of weak defenses are not strong*, 11th {USENIX} Workshop on Offensive Technologies ({WOOT} 17), 2017.

[30] Matthias Hein and Maksym Andriushchenko, *Formal guarantees on the robustness of a classifier against adversarial manipulation*, Advances in Neural Information Processing Systems, 2017, pp. 2266–2276.

[31] Hossein Hosseini, Baicen Xiao, Mayoore Jaiswal, and Radha Poovendran, *On the limitation of convolutional neural networks in recognizing negative images*, 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE, 2017, pp. 352–358.

[32] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry, *Adversarial examples are not bugs, they are features*, Advances in Neural Information Processing Systems, 2019, pp. 125–136.

[33] Jörn-Henrik Jacobsen, Jens Behrmann, Nicholas Carlini, Florian Tramèr, and Nicolas Papernot, *Exploiting excessive invariance caused by norm-bounded adversarial robustness*, CoRR **abs/1903.10484** (2019).

[34] Jörn-Henrik Jacobsen, Jens Behrmann, Richard Zemel, and Matthias Bethge, *Excessive invariance causes adversarial vulnerability*, arXiv preprint arXiv:1811.00401 (2018).

[35] Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer, *Reluplex: An efficient smt solver for verifying deep neural networks*, International Conference on Computer Aided Verification, Springer, 2017, pp. 97–117.

[36] Alexey Kurakin, Ian Goodfellow, and Samy Bengio, *Adversarial examples in the physical world*, arXiv preprint arXiv:1607.02533 (2016).

[37] Yann LeCun, Yoshua Bengio, Hinton Geoffrey, Geoffrey Hinton, Lecun Y., Bengio Y., Hinton G., and Hinton Geoffrey, *Deep learning*, Nature **521** (2015), no. 7553, 436–444.

[38] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al., *Gradient-based learning applied to document recognition*, Proceedings of the IEEE **86** (1998), no. 11, 2278–2324.

[39] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana, *Certified robustness to adversarial examples with differential privacy*, arXiv preprint arXiv:1802.03471 (2018).

[40] Bin Liang, Hongcheng Li, Miaoqiang Su, Xirong Li, Wenchang Shi, and Xiaofeng Wang, *Detecting adversarial examples in deep networks with adaptive noise reduction*, arXiv preprint arXiv:1705.08378 (2017).

[41] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh, *Towards robust neural networks via random self-ensemble*, Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 369–385.

[42] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu, *Towards deep learning models resistant to adversarial attacks*, arXiv preprint arXiv:1706.06083 (2017).

[43] Xudong Mao, Qing Li, Haoran Xie, Raymond Y.K. Lau, Zhen Wang, and Stephen Paul Smolley, *Least squares generative adversarial networks*, The IEEE International Conference on Computer Vision (ICCV), Oct 2017.

[44] Dongyu Meng and Hao Chen, *Magnet: A two-pronged defense against adversarial examples*, Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (New York, NY, USA), CCS '17, ACM, 2017, pp. 135–147.

[45] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff, *On detecting adversarial perturbations*, arXiv preprint arXiv:1702.04267 (2017).

[46] Matthew Mirman, Timon Gehr, and Martin Vechev, *Differentiable abstract interpretation for provably robust neural networks*, International Conference on Machine Learning, 2018, pp. 3578–3586.

[47] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard, *Universal adversarial perturbations*, Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1765–1773.

[48] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng, *Reading digits in natural images with unsupervised feature learning*, NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011, 2011.

[49] Anh Nguyen, Jason Yosinski, and Jeff Clune, *Deep neural networks are easily fooled: High confidence predictions for unrecognizable images*, Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 427–436.

[50] Augustus Odena, Christopher Olah, and Jonathon Shlens, *Conditional image synthesis with auxiliary classifier gans*, Proceedings of the 34th International Conference on Machine Learning-Volume 70, JMLR. org, 2017, pp. 2642–2651.

[51] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, et al., *Technical report on the cleverhans v2. 1.0 adversarial examples library*, arXiv preprint arXiv:1610.00768 (2016).

[52] Nicolas Papernot and Patrick McDaniel, *Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning*, arXiv preprint arXiv:1803.04765 (2018).

[53] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami, *Distillation as a defense to adversarial perturbations against deep neural networks*, 2016 IEEE Symposium on Security and Privacy (SP), IEEE, 2016, pp. 582–597.

[54] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami, *The limitations of deep learning in adversarial settings*, CoRR **abs/1511.07528** (2015).

[55] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer, *Automatic differentiation in pytorch*, NIPS-W, 2017.

[56] Aaditya Prakash, Nick Moran, Solomon Garber, Antonella DiLillo, and James Storer, *Deflecting adversarial attacks with pixel deflection*, Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8571–8580.

[57] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang, *Certified defenses against adversarial examples*, arXiv preprint arXiv:1801.09344 (2018).

[58] Jonas Rauber, Wieland Brendel, and Matthias Bethge, *Foolbox: A python toolbox to benchmark the robustness of machine learning models*, arXiv preprint arXiv:1707.04131 (2017).

[59] Ali Razavi, Aaron van den Oord, and Oriol Vinyals, *Generating diverse high-fidelity images with vq-vae-2*, Advances in Neural Information Processing Systems, 2019, pp. 14866–14876.

[60] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen, *Improved techniques for training gans*, Advances in neural information processing systems, 2016, pp. 2234–2242.

[61] Pouya Samangouei, Maya Kabkab, and Rama Chellappa, *Defense-gan: Protecting classifiers against adversarial attacks using generative models*, arXiv preprint arXiv:1805.06605 (2018).

[62] Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel, *Towards the first adversarially robust neural network model on mnist*, (2018).

[63] Lukas Schott, Jonas Rauber, Wieland Brendel, and Matthias Bethge, *Robust perception through analysis by synthesis*, arXiv preprint arXiv:1805.09190 (2018).

[64] Ali Shafahi, W Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein, *Are adversarial examples inevitable?*, arXiv preprint arXiv:1809.02104 (2018).

[65] Rui Shu, Hung Bui, and Stefano Ermon, *Ac-gan learns a biased distribution*, NIPS Workshop on Bayesian Deep Learning, 2017.

[66] Chawin Sitawarin and David Wagner, *Defending against adversarial examples with k-nearest neighbor*, arXiv preprint arXiv:1906.09525 (2019).

[67] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman, *Pixeldefend: Leveraging generative models to understand and defend against adversarial examples*, arXiv preprint arXiv:1710.10766 (2017).

[68] Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon, *Constructing unrestricted adversarial examples with generative models*, Advances in Neural Information Processing Systems, 2018, pp. 8312–8323.

[69] James C Spall et al., *Multivariate stochastic approximation using a simultaneous perturbation gradient approximation*, IEEE transactions on automatic control **37** (1992), no. 3, 332–341.

[70] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, *Dropout: a simple way to prevent neural networks from overfitting*, The Journal of Machine Learning Research **15** (2014), no. 1, 1929–1958.

[71] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus, *Intriguing properties of neural networks*, arXiv preprint arXiv:1312.6199 (2013).

[72] Vincent Tjeng, Kai Xiao, and Russ Tedrake, *Evaluating robustness of neural networks with mixed integer programming*, arXiv preprint arXiv:1711.07356 (2017).

[73] Florian Tramèr, Jens Behrmann, Nicholas Carlini, Nicolas Papernot, and Jörn-Henrik Jacobsen, *Fundamental tradeoffs between invariance and sensitivity to adversarial perturbations*, arXiv preprint arXiv:2002.04599 (2020).

[74] Florian Tramer and Dan Boneh, *Adversarial training and robustness for multiple perturbations*, Advances in Neural Information Processing Systems 32 (H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, eds.), Curran Associates, Inc., 2019, pp. 5866–5876.

[75] Florian Tramèr, Pascal Dupré, Gili Rusak, Giancarlo Pellegrino, and Dan Boneh, *Ad-versarial: Defeating perceptual ad-blocking*, arXiv preprint arXiv:1811.03194 (2018).

[76] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel, *Ensemble adversarial training: Attacks and defenses*, arXiv preprint arXiv:1705.07204 (2017).

[77] Florian Tramer, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel, *Ensemble adversarial training: Attacks and defenses*, International Conference on Learning Representations (ICLR), 2018.

[78] Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel, *The space of transferable adversarial examples*, arXiv preprint arXiv:1704.03453 (2017).

[79] Florian Tramèr, Fan Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart, *Stealing machine learning models via prediction apis*, 25th USENIX Security Symposium (USENIX Security 16) (Austin, TX), USENIX Association, August 2016, pp. 601–618.

[80] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry, *Robustness may be at odds with accuracy*, arXiv preprint arXiv:1805.12152 (2018).

[81] Jonathan Uesato, Brendan O'Donoghue, Aaron van den Oord, and Pushmeet Kohli, *Adversarial risk and the dangers of evaluating against weak attacks*, arXiv preprint arXiv:1802.05666 (2018).

[82] Arash Vahdat and Jan Kautz, *Nvae: A deep hierarchical variational autoencoder*, arXiv preprint arXiv:2007.03898 (2020).

[83] Siyue Wang, Xiao Wang, Pu Zhao, Wujie Wen, David Kaeli, Peter Chin, and Xue Lin, *Defensive dropout for hardening deep neural networks under adversarial attacks*, Proceedings of the International Conference on Computer-Aided Design, ACM, 2018, p. 71.

[84] Eric Wong and J Zico Kolter, *Provable defenses against adversarial examples via the convex outer adversarial polytope*, arXiv preprint arXiv:1711.00851 (2017).

[85] Eric Wong, Frank R Schmidt, and J Zico Kolter, *Wasserstein adversarial examples via projected sinkhorn iterations*, arXiv preprint arXiv:1902.07906 (2019).

[86] Han Xiao, Kashif Rasul, and Roland Vollgraf, *Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms*, CoRR **abs/1708.07747** (2017).

[87] Kai Y Xiao, Vincent Tjeng, Nur Muhammad Shafiullah, and Aleksander Madry, *Training for faster adversarial robustness verification via inducing relu stability*, arXiv preprint arXiv:1809.03008 (2018).

[88] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille, *Mitigating adversarial effects through randomization*, arXiv preprint arXiv:1711.01991 (2017).

[89] Weilin Xu, David Evans, and Yanjun Qi, *Feature squeezing: Detecting adversarial examples in deep neural networks*, arXiv preprint arXiv:1704.01155 (2017).

Vita

Candidate's full name: Mehrgan Khoshpasand Foumani

University attended:
University of New Brunswick, Master of Computer Science 2018-2020
University of New Brunswick, 2015-2018, Bachelor of Computer Science

Publications:
Mehrgan Khoshpasand and Ali Ghorbani. On the generation of unrestricted adversarial examples. The DSN Workshop on Dependable and Secure Machine Learning (DSML), 06 2020
Mehrgan Khoshpasand, Ali Ghorbani, Samaneh Mahdavifar, and Hessam Mohammadian. Deep learning in adversarial settings. Book chapter submitted for publication., 2020

Conference Presentations:
Mehrgan Khoshpasand and Alireza Manashty. Smart-phone based human fall detection using recurrent neural networks. 26th Annual Graduate Research Conference, UNB, 2019