UNIVERSITY OF CALIFORNIA, MERCED

Defense Frameworks Against Adversarial Attacks on Deep Learning Models

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering and Computer Science

by

Zhixun He

Committee in charge:

Dr. Mukesh Singhal, Chair
Dr. Shawn Newsam
Dr. Florin Rusu

2024

The dissertation of Zhixun He is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

(Dr. Shawn Newsam)

(Dr. Florin Rusu)

(Dr. Mukesh Singhal, Chair)

University of California, Merced

2024

DEDICATION

I dedicate this thesis to my family, for instilling in me a love for learning and always believing in my abilities. This work is also dedicated to my advisors, mentors, and colleagues, whose guidance and insights have shaped my research and scholarly growth. Finally, I dedicate this thesis to all those whose names may not be mentioned but whose contributions, large and small, have left an indelible mark on my life and academic endeavors. Your belief in me and your encouragement along the way have been deeply appreciated.

EPIGRAPH

Stay hungry,
stay foolish.
—Steve Jobs

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisor, Professor Mukesh Singhal, for his unwavering support, guidance, and encouragement throughout my doctoral studies. His expertise, mentorship, and insightful feedback have been invaluable in shaping this thesis and my academic growth.

I am also grateful to the members of my dissertation committee, Professor Shawn Newsam and Professor Florin Rusu, for their valuable feedback and suggestions that have enhanced the quality of this work.

I extend my appreciation to my colleagues and friends for their support and companionship. Their encouragement and discussions have enriched my academic experience and provided much-needed motivation.

I am indebted to the University of California, Merced for the financial support provided during my doctoral research. Their support enabled me to focus on my studies and research, and I am grateful for the opportunity.

Finally, I would like to thank my family for their love, understanding, and encouragement throughout this journey. Their unwavering support has been a source of strength, and I am truly grateful for their presence in my life.

VITA

2013	B. E. in Composite Material Science & Engineering, Nanjing Tech University, China
2013-2015	Private Foreign Language Coach, Guangxi, China
2019	AI & ML instructor, Digital Media Academy, Stanford, CA
2015-2024	Teaching Assistant, University of California, Merced, CA
2015-2024	Ph. D. candidate in Electronic Engineering & Computer Science, University of California, Merced, CA

PUBLICATIONS

Zhixun He, Mukesh Singhal, "VQUNet: Vector Quantization U-Net for Defending Adversarial Attacks by Regularizing Unwanted Noise", in 7th International Conference on Machine Vision and Application, March 2024.

Zhixun He, Mukesh Singhal, "Defense-CycleGAN: A Defense Mechanism Against Adversarial Attacks Using CycleGAN to Reconstruct Clean Images", in 3rd International Conference on Pattern Recognition and Machine Learning, July 2022.

Zhixun He, Mukesh Singhal, "Adversarial Defense Through High-Frequency Loss Variational Autoencoder Decoder and Bayesian Update With Collective Voting", 17th International Conference on Machine Vision Applications, Jun. 2021.

Chandrayee Basu, Erdem Biyik, Zhixun He, Mukesh Singhal, and Dorsa Sadigh, “Active Learning of Reward Dynamics from Hierarchical Queries”, Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), Nov. 2019.

ABSTRACT OF THE DISSERTATION

Defense Frameworks Against Adversarial Attacks on Deep Learning Models

by

Zhixun He

Doctor of Philosophy in Electrical Engineering and Computer Science

University of California Merced, 2024

Dr. Mukesh Singhal, Chair

Deep learning has made remarkable progress over the past decade across various fields, such as Computer Vision, Natural Language Processing (NLP), and Speech Recognition, driving innovation and advancements of various applications. One key challenge is to improve deep learning models' generalization, which refers to the capability of a model to handle unseen data. Several factors contribute to the challenge of generalization in deep learning models, including limited data availability, overfitting tendencies, and the inherent complexity of the models themselves.

The phenomenon of adversarial samples is symptomatic of the limitation in the generalization capability of deep learning models. Adversarial samples refer to data instances manipulated from their originals, often appearing visually similar and imperceptible to human senses, yet causing incorrect predictions from deep learning models. This phenomenon presents a substantial concern for security-sensitive domains like medical diagnosis, autonomous driving, and anomaly detection, where model reliability is crucial.

The development of various adversarial defense methods in recent years, such as adversarial training, noise reduction, and gradient masking, emphasizes the considerable efforts to enhance the robustness and reliability of deep learning models. Meanwhile, as innovative adversarial attacks continue to evolve, they effectively expose the vulnerabilities

inherent in deep learning models, thereby raising challenges for existing defense methodologies. Although adversarial defense research has made advancements, the root cause of the vulnerability in deep learning models is still not fully understood. Additionally, there is a pressing need for defense mechanisms that offer comprehensive protection and high resilience against a wide range of adversarial attacks. The research presented in this dissertation aims to contribute to the enrichment of knowledge within the research community by providing deeper insights into adversarial attacks and defense mechanisms. It endeavors to develop novel defense methods that are robust and reliable when protecting deep learning models against adversarial threats. Through this work, we seek to advance the field of adversarial defense and contribute to the development of more effective defense strategies.

In this dissertation, we introduced three novel defense mechanisms aimed at enhancing the robustness of deep learning models against adversarial attacks. In the first work, we tackled the issue of image blurring in traditional Variational Autoencoder (VAE)-based generative networks by focusing on improving high-fidelity data reconstruction. Additionally, this work optimized the model's decision-making strategy through a Bayesian update, allowing a model to incorporate multiple sources of supporting evidence for the final decision. The second study proposed a new generative network structure coupled with a new two-step noise reduction approach designed to effectively filter out adversarial noise. The third method introduced a new noise reduction mechanism called VQUNet. This method features a learnable quantization of latent features and a hierarchical network structure for high-fidelity data reconstruction. VQUNet's unique design significantly enhances the data reconstruction quality after the filtering process, while effectively regularizing adversarial perturbation within the network, thereby improving its resilience against adversarial attacks.

Extensive experimental investigations demonstrated that the proposed methods provided superior robustness to the targeted deep learning models. They exhibited superior performance over other state-of-the-art noise-reduction-based defense methods, achieving prediction improvement with a notable margin over existing methods under adversarial attacks across both Fashion-MNIST and CIFAR10 datasets. The experimental analysis

underscored the effectiveness, resilience, and robustness of the proposed methods against adversarial attacks. These findings offered valuable insights into the development of effective defense strategies, shedding light on the mechanisms and principles for future research.

Chapter 1

Introduction

1.1 Background

Over the past few decades, with the availability of large amounts of data and the computation power from hardware, the deep learning approach has become the 'cream of the crop' for solving various traditionally challenging problems [58, 87, 76]. The research community demonstrates its superior performance on various tasks for images [53], natural language processing [78], acoustics [79] and videos [48], etc.

Recently, however, the vulnerability of deep learning algorithms has been also broadly observed in research community [92, 26, 105, 106], that an adversary can purposely craft data that can cause the deep learning models to give wrong predictions. Those crafted data are usually similar to natural data and imperceptible to human sensation. For example, in Figure 1.1, the two images both show the same airplane, where the one on the left is the original image, while the one on the right is an adversarial counterpart. It's safe to assume that many decently performed image recognition deep learning models would be capable of correctly predicting what object is in the image. However, when a well-performed classifier is being attacked it will give a wrong prediction on the object in the adversarial image as a "deer" instead of "airplane".

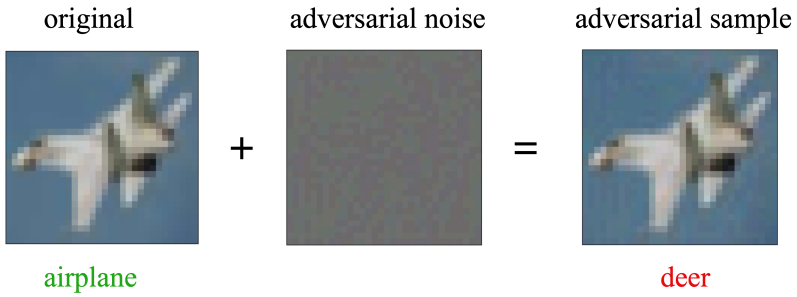


Figure 1.1: Left: an airplane image sampled from CIFAR10 data set. Middle: adversarial noise generated using the Fast Gradient Sign Method (FGSM) method, an adversarial attack algorithm. Right: an adversarial counterpart generated with FGSM.

Such practice that makes small, often imperceptible, changes to the input data in order to impair the integrity of deep learning algorithms is known as adversarial attacks [24, 87]. Adversarial samples raise severe security concerns to largely deploy real-world deep learning applications. For example, by injecting malicious adversarial noise into a model input, the trained model will be misled to give a wrong prediction, impairing its integrity and performance. Such threat makes users hesitate about the accuracy and reliability of machine learning systems, raising concerns about largely deploying deep learning systems, such as the vision system in autonomous vehicles, surveillance systems, medical analysis with deep learning, and so on. Therefore, it is important to conduct a thorough examination of the factors that make deep learning algorithms susceptible to adversarial attacks. Developing effective defense mechanisms against such attacks is crucial for the widespread and secure deployment of deep learning applications in human society.

The susceptibility of deep learning models to adversarial attacks has been a focal point of research within the deep learning community over the past decade, with a growing body of works developed [2, 105]. Researchers have explored various aspects of this phenomenon, seeking to understand the underlying mechanisms and safeguard deep learning models against adversarial attacks [2, 96, 86, 61, 26, 105, 27, 67, 72, 30, 109] (more will

be discussed in chapter 2). These efforts reflect a persistent effort within the research community to develop more resilient and reliable defense mechanisms.

1.2 Problem Statement

Despite significant advancements, achieving complete immunity against adversarial attacks remains a persistent challenge. The evolving nature of these attacks, coupled with the complexity of deep learning models, underscores the ongoing need for robust defense strategies. Traditional methods for defending against adversarial attacks often face difficulties in effectively mitigating the impact of emerging attack strategies.

With the recent surge in artificial intelligence, numerous major corporations across different industries have begun integrating deep-learning-based services into their products. This trend highlights the pressing need to enhance the robustness of deep learning models. The ongoing efforts within the research community to develop more reliable and effective defenses against adversarial threats aim to fulfill several key requirements for a robust defense strategy:

- A robust defense mechanism should be able to provide protection against a wide range of adversarial attacks, regardless of their nature or strategy.
- Given that adversarial attacks can introduce varying levels of noise to images, a defense method should maintain consistent performance across a broad spectrum of adversarial noise levels.
- In the absence of an adversarial attack, the presence of a defense mechanism should not compromise the original performance of the deep learning application.
- For widespread adoption, a defense method should impose minimal computational overhead, ensuring that it can be efficiently run on most devices without significantly draining battery resources.

However, existing defense mechanisms often suffer from limitations such as reduced model performance on clean data, limited robustness against sophisticated attacks, and lack of consistency over various adversarial noise levels. Therefore, it is imperative to attain a comprehensive understanding of the vulnerability inherent in deep learning models and develop robust defense mechanisms capable of securing the reliability of these models.

This dissertation aims to tackle these challenges by proposing novel defense mechanisms that improve the robustness of deep learning models, thereby advancing the current knowledge base in the field. By contributing novel insights and analytic evaluations, this work seeks to facilitate the development of more effective and efficient defense strategies.

1.3 Contributions

This dissertation introduced and evaluated three innovative adversarial defense methods through extensive experimental analysis. The primary objective of these methods is to further the understanding of adversarial attacks and enhance the reliability and security of deep learning systems in facing adversarial attacks. The contributions of this dissertation can be summarized as follows:

- The dissertation introduced a novel VAE structure enhanced by spacial frequency loss, aimed at improving the sharpness of edges and the contrast in reconstructed image data. Furthermore, a collective voting mechanism was proposed, consistently providing a notable gain in performance accuracy under various adversarial attacks. Additionally, this study proposed a Bayesian Update mechanism, which serves as a statistical framework for systematically incorporating multiple forms of supporting evidence into the final decision-making process.

- A new adversarial noise reduction framework, Defense-CycleGAN, was introduced, significantly enhancing the data reconstruction fidelity. The key idea is to design a new end-to-end training framework that consists of two GANs, which are guided

by cycle consistency loss [47] to learn the true data distribution separately while endeavoring to clean the harmful noise in each other's outputs. The new design leads to high-fidelity data reconstruction quality, which not only enhances the overall performance of targeted models against various adversarial attacks but also mitigates the performance degradation issues often encountered by previous defense methods.

- The dissertation presented a novel adversarial noise reduction method named VQUNet, which achieved state-of-the-art data reconstruction quality while incurring less than a 1% performance degradation for the deep learning models. The key to achieving detail-rich data reconstruction lies in the hierarchical structure to combine both primitive latent features and highly abstract latent features when rebuilding the data. Notably, VQUNet is the first work, to the best of our knowledge, that dynamically learns a vector quantization mapping between continuous vectors and quantized vectors for various hierarchies to mitigate the adversarial perturbation within a network structure. This quantization mechanism provides unique regularization to the noise perturbation, significantly reducing the impact of adversarial attacks.

- The efficacy of the proposed defense methods has been thoroughly evaluated through extensive experiments and analytical assessments across various adversarial contexts. A comprehensive comparison of the proposed methods with multiple other state-of-the-art defense methods has been provided to demonstrate the effectiveness and reliability offered by the proposed methods. The experimental results consistently showed that the proposed methods outperform other contemporary techniques by a notable margin, highlighting their efficacy and reliability in challenging adversarial scenarios. These findings underscored the potential impact of the proposed methods in advancing the development of adversarial defense techniques.

- The comprehensive analysis presented in this dissertation provides valuable insights into adversarial attacks and defense mechanisms, enriching the knowledge base of the field and facilitating the ongoing efforts to develop more potent and efficient

defense mechanisms in the research community.

1.4 Dissertation Outline

In Chapter 2, we delve into the related work and essential notations pertinent to this dissertation. This chapter encompasses three domains crucial to understanding and developing defense methods for deep learning systems: adversarial attacks, adversarial defenses, and generative networks.

Chapter 3 presents the first novel adversarial defense method that focuses on enhancing the quality of data reconstruction by introducing a Spacial Frequency Loss-enhanced VAE. This defense approach further improves the reliability of the target deep learning models by combining multiple supporting evidence through the Bayesian Update to enhance the confidence for the models' final decision.

In Chapter 4, a second innovative method called Defense-CycleGAN is proposed to mitigate the impact of adversarial noises and enhance high-fidelity data reconstruction. This method features an end-to-end training framework that consists of two GANs that offer a paralleled noise reduction process.

Chapter 5 presents the third novel method named VQUNet, aiming to further reduce the information loss during the data compression stage while regularizing adversarial noise. The key idea revolves around a novel data reconstruction pipeline and a vector quantization mechanism to mitigate the impact of adversarial noise.

Chapter 6 provides a summary of the dissertation, highlighting the results and contributions of the proposed works.

Chapter 2

Literature Review

In this chapter, we provide a thorough review of related literature, techniques, and research progress in both adversarial attacks and defense. We begin with definitions and notations that will be used in this dissertation. We then highlight key insights and foundational concepts of adversarial attack algorithms. Furthermore, we discuss relevant studies and the progress of adversarial defense methods, categorizing recent advancements and techniques in defense strategies based on their theoretical foundations. Toward the end, we examine and analyze two of the most important state-of-the-art deep learning generative paradigms, aiming to provide an overview of the theoretical implications in terms of data transformation and noise reduction in the realm of adversarial defense.

2.1 Definition and Notation

Symbol	Meaning
x	An original unperturbed data point.
x^{adv}	An adversarial sample generated from data point x.
y	The ground truth label for data x.
$\hat{y}$	The predicted label from a deep learning model.

θ	All the parameters in a deep learning model.
$L(\theta, x, y)$	The loss of a deep learning model given the parameters, input, and label.
$sign(\cdot)$	A function that outputs the sign of its input (e.g., + or −).
$\nabla \cdot$	The derivative of a variable.
$\nabla_x L(\theta, x, y)$	The derivative of a model's loss w.r.t. x
$clip_{min,max}(\cdot)$	A function that clips the value of its input if the value is beyond the maximum (max) or below the minimum (min).
f	A general function that takes some inputs.
$f(\cdot)$	The output of a general function f with some inputs.
$\{\}$	An unordered set.
$\lvert \cdot \rvert$	The absolute value of a given variable.
$\lvert \{...\} \rvert$	The number of the elements in a given set.
$\lVert v \rVert_p$	The p-norm of a vector v, which equals to $(\sum_i \lvert v_i \rvert^p)^{1/p}$.
$\lVert v \rVert_p^2$	The square of the p-norm of a vector v, which equals to $(\sum_i \lvert v_i \rvert^p)^{2/p}$.
$[a, b]^n$	A n dimensional vector space, where each element $v_i : a \leq v_i \leq b$.
ϵ	A scalar, the perturbation magnitude that an adversarial attack adds on top of the original input x.
$Pr(\cdot)$	The probability of a given condition.
$\mathcal{D}$	The distribution for natural training data or samples.
$\mathbb{E}_{x \sim \mathcal{D}} [f(x)]$	The expected value of a given function, where the input x follows a certain distribution.
$G(x;\, \theta)$, $G_\theta(x)$	The output of a generative neural network, which is parameterized by θ and takes x as the network's input.
$D(x;\, \phi)$, $D_\phi(x)$	The output of a discriminator neural network, which is parameterized by ϕ and takes x as the network's input.

$E(x;\,\theta)$, $E_\theta(x)$	The output of an encoder neural network, which is parameterized by θ and takes x as the network's input.
$D(x;\,\phi)$, $D_\phi(x)$	The output of a decoder neural network, which is parameterized by ϕ and takes x as the network's input (Overloaded with discriminator's notation, this is used based on context).
$\min_\theta L(x,\theta)$	Given a function L, optimizing the value of θ in order to minimize the overall value of $L(x,\theta)$
$\max_\theta L(x,\theta)$	Given a function L, optimizing the value of θ in order to maximize the overall value of $L(x,\theta)$

Table 2.1: Notations that are used in this dissertation

2.2 Adversarial Attacks

Adversarial attacks are techniques that cause machine learning models to give wrong predictions by intentionally injecting malicious values into the input data, often small enough to be unnoticeable for human perception. Adversarial attacks can cause serious failures among machine learning applications in various domains, such as in images, videos, natural language processing, and custom data structures. Various adversarial attack algorithms have been developed [88], but based on how much information an adversary has about the targeted model, the attacking models can be generally categorized into *white-box*, *black-box* and *gray-box*.

2.2.1 White-box Attack

If an adversary has full access to the targeted deep learning models, such as the learned weights, network structure, and hyper-parameters, such attacks are categorized as white-box attacks. In comparison, under black-box attacks, the adversary does not have access

to most of the information of the targeted models except the prediction results from the models (further discussion about black-box attacks will be given in section 2.2.2). Because a white-box attacker has full knowledge of the models, this type of attack can usually craft more efficient and effective adversarial noise. With full access to the target model, the adversarial noise is carefully calculated and almost imperceptible to humans or traditional anomaly detection algorithms.

It is particularly concerning when a white-box attacker systematically probes and exploits the vulnerabilities of some foundation models that might be already used and adopted in various AI-powered products in the real world, as adversarial samples created using one model can also impede other different models [104]. Such a scenario reveals sensitive information about a series of models, compromising models' security, integrity, and reliability.

Fast Gradient Sign Method (FGSM)

FGSM is a well-known adversarial attack algorithm, introduced by [26], which leverages the information of neural network structure and its parameters' value to calculate the adversarial noises that are applied to the original image x. FGSM is renowned for its early observation of the adversarial attack, highlighting its simplicity in computation and the theoretical implications of neural networks' vulnerability.

Given the network's parameters θ, the input image x, and their corresponding ground truth y, the fundamental idea behind FGSM is to compute the derivative of the deep learning model's loss function with respect to the model's input space, $\nabla_x L(\theta, x, y)$, then generates the adversarial counterpart using gradient ascent:

$$x^{adv} = x + \epsilon \cdot sign(\nabla_x L(\theta, x, y)) \tag{2.2.1}$$

FGSM has been widely studied and has become one of the standard testing attacks to evaluate the robustness of machine learning models in various domains [3].

Basic Iterative Method (BIM)

BIM [56], also known as the Iterative Fast Gradient Sign Method (I-FGSM), calculates

the adversarial perturbation in a similar fashion as FGSM does. As an extension of the FGSM, BIM generates stronger adversarial samples that surpass the success rates compared to FGSM [56]. Both adversarial attack algorithms compute the derivative of the loss of the targeted deep learning model with respect to the input space. However, the key idea behind BIM is to perform multiple times of FGSM-based adversarial noise computations. The output image from the previous iteration is used as the input for generating a new adversarial image until a certain number of iterations is reached. In each iteration, only a small amount of adversarial perturbation will be injected into the generated data from the previous iteration. Each iteration crafts the adversarial noise in the direction that maximizes the targeted model's loss while maintaining a small amount of overall adversarial perturbation, keeping it as indistinguishable from the original data as possible.

Formally, given the network's parameters θ, the input image x, and their corresponding ground truth y, a learning step α, and a clip function that makes sure each pixel of the final output adversarial image will not surpass the maximum difference of $\epsilon - neighborhood$ from the original input image x, the adversarial sample for each iteration i is computed as:

$$\begin{aligned} x_0^{adv} &= x \\ x_{N+1}^{adv} &= clip_{x-\epsilon,x+\epsilon}\{x_N^{adv} + \alpha \cdot sign(\nabla_x L(\theta, x_N^{adv}, y))\} \end{aligned} \tag{2.2.2}$$

where L is the loss function for the model during the training phase, and x_i^{adv} represents the adversarial sample crafted at iteration i.

BIM has been demonstrated to be more transferable compared to FGSM across different deep learning models [35]. Compared to FGSM, BIM extends the search in input space, allocating more weights to weaker input space and discovering a more vulnerable region in the input space for a model. Both FGSM and BIM provide theoretical insights into a model's vulnerability and inspire the development of models with more robustness.

Carlini and Wagner Method (CW)

CW [12] is one of the most powerful and widely recognized adversarial attack algorithms. It is known for being able to generate adversarial samples that are effective and

imperceptible to human sensations [104]. It is designed to create adversarial samples by minimizing the adversarial perturbation applied to the original images while minimizing the probability of correct prediction from the targeted deep-learning model.

This adversarial attack is formulated to a constrained optimization problem, aiming to find the smallest perturbation that causes a misclassification [92]. Unlike previous methods, the CW method formulates the adversarial perturbation as a continuous optimization problem [12]:

$$\begin{aligned} &\textbf{\textit{minimize}} \ \|\epsilon\|_p + c \cdot P(f(x+\epsilon) = y) \\ &\textbf{\textit{such that}} \ x + \epsilon \in [0,1]^n \end{aligned} \tag{2.2.3}$$

where $f(\cdot)$ is the prediction output of a deep learning model given its input image x, and c is a constant that is chosen heuristically.

The key innovation of the CW method includes that (1) it formulates the adversarial noise as a continuous optimization problem; (2) it considers both the overall noise perturbation level and the confidence of a targeted model's predictions on the generated adversarial samples, minimizing the adversarial noise level while maximizing the likelihood of wrong prediction.

Deepfool Method

Deepfool [75] is also one very effective and efficient adversarial attack algorithm, which also takes advantage of the information of a neural network's parameters to calculate the adversarial noise. The adversarial samples are crafted by calculating the shortest distance between the input data and the decision boundary given a trained model. It is done by modifying the original data in order to cause it to trespass the model's decision boundary. Unlike other gradient-based approaches, the Deepfool method distinguishes itself from other adversarial attack algorithms by formulating the searching of adversarial perturbation as an optimization problem with a closed-form solution [75].

The key idea of the Deepfool method is to approximate a model's decision boundary with linearization, such that the closest boundary can be found using other out-of-shell

optimization algorithms [75]. The adversarial perturbation that is required for the adversarial sample to cross the decision boundary is computed in an interactive fashion. Because most deep-learning models do not train for buffering the decision boundary for a learning task, Deepfool can find a minimum distance to the decision boundary, crafting effective adversarial noise that is used to generate adversarial samples similar to their original counterparts.

In the version for a binary classifier, the $sign(\cdot)$ of the output of the binary classifier, $f(x_i)$, is the decision of the classifier. The adversarial sample is generated iteratively:

$$\begin{aligned} &\textbf{while } sign(f(x_i)) = sign(f(x_0)) \textbf{ do} \\ &\quad x_{i+1} \leftarrow x_i - \frac{f(x_i)}{\left\|\nabla_{x_i} f(x_i)\right\|_2^2} \nabla_{x_i} f(x_i) \\ &\quad i \leftarrow i + 1 \end{aligned} \tag{2.2.4}$$

Different from FGSM and BIM, Deepfool crafts adversarial samples using an iterative approach and the iteration stops when the prediction on an image is different from its ground truth label. Also different from BIM, it does not simply add a fixed level of noise to the images as a threshold, e.g., ϵ, instead, a new set of adversarial noise values is calculated for each step. Compared to the CW method, where both methods formulate the adversarial perturbation as an optimization problem, the Deepfool method shows better computational efficiency during the actual implementation, demonstrating both faster implementation and lighter hardware resource requirements [35]. The Deepfool method has become one of the most widely studied adversarial attack algorithms and is used as one major attack for evaluating and analyzing adversarial defense methods.

2.2.2 Black-box Attack

Under black-box attacks [21], the adversary has limited knowledge or does not have access to most of the information of a classifier. Often, the knowledge available to such attackers is collected by feeding adversarial samples that are generated with some

adversarial algorithms and studying the input-output pairs as feedback. Despite the limited knowledge available to black-box attackers, it has been shown such attacks are challenging as they work effectively against a wide range of machine-learning models across various domains [107].

The black-box attacker can not apply a gradient-based approach for attacking the targeted model because they do not have access to the learned parameters, hyper-parameters, structure of the network, training sets, etc. However, they can generate adversarial samples using various heuristic approaches. For example, some adversarial samples can be generated using the information of a different but known model that serves a similar alternative in place of the targeted model, and some of those adversarial samples are also able to deceive the targeted model.

Those alternative models usually have different structures, hyper-parameters, or training processes from the targeted model, however, the adversarial samples generated from an alternative model can still impair the targeted model. This is known as adversarial transferability among different machine learning models and among different datasets, which raises particular cautions in the research community [85]. Such transferability reveals the susceptibility of deep learning models, which indicates that different models might share a similar decision boundary even if they are designed with different structures, parameters, or learning schemes. In a real-world situation, if a product team bootstraps their models on top of some foundation models or publicly available large meta-models to develop an application, this application might be potentially impeded by adversarial samples created from the foundation models, raising serious security concerns.

The black-box attacks become a significant challenge in machine learning, raising potential risk to largely deploying deep neural network models to real-world applications. Therefore, it's important to understand the vulnerability of deep learning models in order to prevent the system from being exploited by block-box attackers.

2.2.3 Gray-box Attack

Gray-box attack [113] refers to a hybrid adversarial scenario in between white-box and black-box attacks. In comparison, for gray-box attacks, only partial knowledge of the target model will be available to adversarial attackers, instead of full access or zero access. For example, one possible scenario is that the adversary does not have access to the trainable parameters of the model, but other information like the hyper-parameters, optimizer, and/or the structure of the network, etc., are known to the adversarial attacks.

Gray-box attackers can also take advantage of the adversarial transferability between different machine learning models to exploit the vulnerability of target models, like black-box attacks. The goal of gray-box attacks is to leverage whatever knowledge is available to the attackers to generate effective adversarial samples to deceive the target models.

One common approach in a gray-box attack is that the attacker knows the structure of the targeted model and trains a network with a similar structure but a different set of trainable parameters, then the attacker uses this trained model to launch adversarial attacks. The adversarial samples generated from the new model are also effective in deceiving the original target model, due to the transferability of adversarial attacks [85]. The fact that the original deep learning models can be as susceptible as the alternative model with a different set of parameters raises concerns about the reliability and security of deep learning systems.

With gray-box attacks, the vulnerability of the target model will be exploited to harm the robustness of machine learning systems. The gray-box attacks represent both challenges and opportunities to develop a further theoretical understanding of the machine learning system in real-world applications to promote new strategies for defending unwanted attacks, such as new protocols for information leakage through model responses.

2.3 Adversarial Defenses

There are ongoing efforts to develop more effective and robust adversarial defense mechanisms among the research community, and various defense mechanisms have been proposed during recent years [3, 92, 13, 116, 88, 2, 67, 72, 96, 86, 63]. Despite the recent remarkable advancements made in adversarial defense, it still remains a challenge to develop defense mechanisms that are efficient to withstand miscellaneous adversarial attacks. As the adversarial attack is also a rapidly evolving field of research, the potential variation in attacks poses a significant threat to machine learning systems.

This section provides an overview of some current state-of-the-art adversarial defense work. For adversarial defense research, various methods, techniques, and strategies can be used for defense. In general, different defense mechanisms can be summarised into several categories: adversarial training, denoising, provable defense, gradient masking/obfuscation, and detection.

Adversarial Training

Deep learning relies on the training samples from a domain to learn the mapping from input data to desired outputs. By extending the training samples with adversarial samples, the machine learning models can be early exposed to the adversarial perturbations during training, thus, the robustness of deep learning models against adversarial samples can be improved. The training scheme that involves augmenting the training data to improve the resistance of machine learning models against adversarial noise is called adversarial training. Adversarial training has shown promising results in withstanding adversarial perturbations [92].

Not only deterministic adversarial attack algorithms [26, 56, 12, 75, 85] can be used to generate x^{adv} for adversarial training, but also some non-deterministic generator networks [59, 64] that simulate adversarial samples can be an effective way to generate x^{adv}.

Although adversarial training has shown some progress for robustness improvement

[11, 84, 26], there are still challenges: (1) the computational overhead is huge when generating adversarial samples from a large number of original training samples; (2) adversarial perturbation is in continuous space, and it is unknown which specific adversarial counterparts should be used to make the data augmentation more data efficient; (3) in order to reduce the computational cost, only a few selective perturbation levels are used to generate adversarial samples, thus, the adversarial training using few perturbation levels might not cover most other adversarial counterparts; (4) the risk of being attacked by black-box and gray-box attacks still exist, and it is shown that such attacks can still impair an adversarially trained model [12, 14]. It is crucial to find the right balance between the coverage of the possible adversarial perturbations and the computational consumption. Ongoing research strives for effectiveness and scalability in bringing better security to machine learning systems.

Adversarial Noise Reduction

Adversarial noise reduction is a genre of defense techniques, that reduces the impact of adversarial perturbations through regularizing the possible unwanted noise in the input data, in order to improve the robustness of the machine learning models.

Generative neural networks can be used as an efficient paradigm to analyze and understand the underlying distribution of the dataset. A well-trained generative model is capable of generating data that are similar to those in its training data set. Generative models can also add variety to the generated data by injecting random noise to the model's input or intermediate hidden features [95, 49]. Various machine learning techniques can be used to build generative models [80], such as Boltzmann Machines [19, 39], Variational Autoencoder Decoder (VAE) [50] and Generative Adversarial Network (GAN) [25] etc.

In the context of adversarial defense, generative models can be utilized to simulate the original data distribution and generate outputs similar to the original data in disregard of the adversarial noise. By setting the training objective to generate data that belong to a specific domain, the unwanted noise in the input data can be subdued, maintaining

the underlying signal of the original training data. The adversarial noise in the input can be largely reduced or translated into a new feature in the output, although, when the the adversarial noise is too strong, the generated samples might not be meaningful anymore.

Besides handling the unwanted noise in the input, there are other techniques to regularize unwanted signals in the intermediate outputs from the model. Neural networks have internal regularization mechanisms to subdue the impact of adversarial noise, such as Dropout layer, batch normalization, or random noise injection, which also show effectiveness in reducing the impact of adversarial perturbation [36]. Additionally, by injecting random noise into the hidden layers' outputs from the intermediate layers in a model, the dynamic change in hidden features can also act as a regularizer to prevent a model from overfitting the training data.

A more in-depth discussion about different generative models will be given in section 2.4. This section summarizes some related adversarial defense work using different generative models without comparing the advantage of one generative model over the other.

VAE-based Noise Reduction

VAEs [50, 120, 57] is a class of generative neural networks based on the idea of encoding the input into a latent space and then decoding the latent feature back to the original input space. It features a layer with much smaller neurons in the middle of the network compared to the number of neurons for the input and output space. The input and output space usually have the same dimension. The process of converting the input space into a much smaller vector space z defined by this middle layer is called **Encoding** . Similarly, the process of converting z back to the output space is called **Decoding** . The part of the neural network that encodes is named **Encoder**, while the portion for decoding is named **Decoder** accordingly .

The unique structure of VAEs provides an elegant and efficient way to learn a continuous and structured probabilistic distribution over the latent space. At the same time, the decoding process aims to reconstruct data to be similar to the input using the latent feature, which makes it suitable for applications like data compression, Principle Component

Analysis (PCA), feature extraction, posterior inference, and so on [24, 7].

In the adversarial defense context, the potential of VAEs for defending against adversarial attacks has been explored in the research community [110]. The premise behind using VAEs for adversarial defense is the heuristic observation that the encoding process, which transforms the input space x to the middle layer output z, can help disentangle the underlying signal of the training data from the unwanted adversarial noises. The encoding process functions as a regularizer to separate the adversarial noise and stabilize the latent features as learned from the unperturbed original training data. Then the regularized latent features will be fed into the decoding process to convert the z back to outputs that share a domain similar to the unperturbed original data.

Defense-VAE [61] and Magnet [72] represent two distinct yet conceptually similar VAE-based noise reduction adversarial defense methods. Both methods aim to mitigate the impact of adversarial noise in the input space by reducing the unwanted noise through the VAE encoding-decoding process. Specifically, Defense-VAE emphasizes the strategy that incorporates the reconstructed data from the VAE into the targeted model's retraining process. In contrast, Magnet opts not to retrain the targeted model with extra training data, instead, it endeavors to train a VAE capable of reconstructing similar outputs as to the original training data, relying on the inherent robustness of the data generation process from the VAE.

[23] is a unique defense mechanism that imposes a Gaussian mixture prior onto the latent space layer output z as a key mechanism to regularize the training process of a VAE network. The key assumption is that if the difference Dif between the output $z = E(x^{adv})$ from the encoder given an adversarial sample and the default Gaussian mixture prior value μ, $D(z, \mu)$, is found to be sufficiently large compared to a predefined threshold τ, $Dif(z, \mu) > \tau$, then the model will optimize the value of z in order to lower $Dif(z, \mu)$. This process will result in an optimized z^*, which can be subsequently fed into the decoder $D(z^*)$ to generate a new output. Toward the end, this new output is used for re-classification by the target classifier. The refined z^* is the central piece to produce more accurate images

to enhance the model's resilience of prediction.

[123] proposed a novel defense strategy against adversarial attacks, which leverages a conditional VAE and a Bayesian network structure [22]. In addition to the standard VAE training scheme, in their work, the VAE's latent space layer output $z = E(x^{adv})$ is used to calculate its distance from each class's centroid in the latent space, which serves as an indicator for anomaly detection. After an adversarial sample is detected, to recover the correct prediction, a new z_{new} is given by optimizing $z_{new} = \arg\min_{z} \left\| D(z) - x^{adv} \right\|$. Then a separate classifier model, $f_c(z)$, which is trained with sample z and label y, where $z = E(x^{clean})$, gives the correct prediction using z_{new}, as followed: $f_c(E(D(z_{new})))$.

Although VAEs show their ability to generate new samples and reconstruct similar data as its input, the reconstructed data from the VAEs often exhibit various levels of noise, which can be observed as a noticeable blur in images [10]. This blurriness is a well-documented characteristic in VAE-generated images, for example, the reconstructed images from VAE often show a noticeable blur effect [111, 108], with unclear separation in the edges of objects. Not only is this blurry effect noticeable in the case of reducing adversarial noise [123, 23], but in the absence of adversarial noise, the reconstructed images still exhibit blurry effects. This blurry effect causes a negative impact not only on the image quality but also on the case when reusing those images for adversarial defense. The inherent blurriness of the reconstructed images can make it challenging to recognize the objects during the testing phase accurately. While there are existing works focused on high-fidelity image data reconstruction [8, 49], such as those that use large-scale network structure in VAE [90], extra work is still needed in adversarial defense in order to enhance the data generation quality, ensuring that they remain suitable for downstream tasks.

GAN-based Noise Reduction

GANs have emerged as powerful generative models and have been extensively studied over the past years[43, 1]. GANs are capable of generating various types of data, such as images, text, videos, and so on.

One of the early discussions of GAN [25] introduces two essential components of GAN:

(1) a generator neural network and (2) a discriminator neural network. GANs have gained large popularity, and nowadays it often refers to a general deep learning training scheme, where the deep learning system consists of a generator and a discriminator, and the two networks are trained simultaneously in a competitive manner.

The role of the discriminator, D, is to distinguish if a sample belongs to a genuine sample from the training set or a fake one generated by the generator. The role of the generator, G, is to create samples that can mislead the discriminator, making the fake samples as close to genuine samples as possible in order to confuse the discriminator into thinking the fake sample is a genuine sample.

The training of G and D endeavors to maximize the probability that the discriminator can correctly classify whether an input is real or fake, while the generator aims to maximize the probability that the discriminator makes the wrong prediction. Formally:

$$\min_{\theta} \max_{\phi} \mathbb{E}\left[\log D_{\phi}(x) + \log(1 - D_{\phi}(G_{\theta}(z)))\right] \tag{2.3.1}$$

To mimic the min-max game framework during the training of a GAN, the two neural networks (the generator and the discriminator) are optimized alternately to lower their loss functions.

GANs demonstrate their capability to generate high-fidelity and realistic images of faces, animals, and objects [82]. They are particularly versatile for their flexibility in network architecture when designing a GAN model [110]. Because of GANs' impressive performance in their variety and flexibility for data generation, they have shown promise in adversarial defense, especially in the task of noise filtering. Various adversarial defense approaches adopt GANs as the paradigm when designing their framework [62].

The Adversarial Perturbation Elimination Generative Adversarial Network (APE-GAN), introduced by Shen et al. (2017) [99], leverages the training scheme of GANs [25], to generate artificial data to mimic the real dataset. The primary objective of PE-GAN is to enhance the robustness of machine learning models against adversarial attacks by generating noise-reduced counterparts of the given adversarial samples. Specifically, when

presented with an adversarial sample x^{adv}, the trained generator G endeavors to produce a noise-reduced version of the sample $G(x^{adv})$ that is similar to the original counterpart x. By employing a GAN-based training approach, the generator G learns to generate the underlying distribution of the real dataset, mitigating the impact of the adversarial perturbations in the inputs when they are mapped to the learned data distribution. By regularizing the unwanted noise, the defense approach enhances the model's resilience to adversarial attacks.

Defense-GAN, as proposed by Samangouei et al. (2018) [96], represents an innovative approach to adversarial defense that harnesses the power of Generative Adversarial Networks (GANs). The essence of Defense-GAN's strategy lies in its innovation to look into the seeds used by the generator for defense purposes, instead of directly searching for noise-reduced images.

During the testing stage, Defense-GAN employs a multi-step process to generate noise-reduced counterparts for adversarial samples. Initially, multiple random seeds are sampled from a pre-defined distribution, and each seed is fed into the trained generator to produce a noise-reduced output. Subsequently, a distance function is utilized to compare each generated output to the original adversarial input. The output that is deemed most similar to the adversarial input is then selected as the final noise-reduced counterpart for the adversarial sample. The selected noise-reduced sample is then passed through the targeted deep-learning model for prediction. In this way, the targeted model can retain its performance and robustness while facing adversarial attacks.

It is worth noting that while Defense-GAN demonstrates promising results for simpler datasets such as Fashion-MNIST [114], its performance diminishes when applied to more complex datasets such as CIFAR-10 [52]. This suggests that although Defense-GAN offers an innovative way to mitigate adversarial noise, further refinement and optimization are needed to enhance its performance for more challenging datasets.

Miscellaneous Generative Model Noise Reduction

In their work titled "Feature Distillation: DNN-Oriented JPEG Compression Against

Adversarial Examples," Liu et al. (2019) [66] presented a novel approach for mitigating adversarial noise for JPEG compressed images. The key innovation of their method lies in the integration of JPEG compression with a dequantization-quantization-dequantization (DQD) step. The DQD step is applied to the input JPEG image before feeding it into the targeted deep learning model. This preprocessing step aims to effectively reduce the impact of partial adversarial noise present in the original input image and output the image with subdued unwanted noise.

The idea behind this approach is rooted in the observation that the JPEG compression process inherently introduces a certain level of noise into images. Such noise injection is regarded as a regularizing technique often used in deep learning for preventing overfitting during the training process. With the regularization, the JPEG compression can partially have the effect of disturbing the harmful adversarial noise. During the dequantization stage, the input images are recovered back to floating-point representation, and with the subsequent re-quantization and dequantization again, the adversarial perturbations are attenuated compared to the initial adversarial noise. It is worth noting that the innovation of this JPEG-compression-based method distinguishes itself from many other noise-reduction-based adversarial defense methods. This is because JPEG compression happens during the data cleaning stage when developing deep learning applications. It is possible to enhance the robustness of deep learning models by incorporating the DQD processing step as a routine during the data cleaning phase in the life cycle of machine learning application development.

In the domain of image processing, researchers have explored a range of image transformations specifically tailored for JPEG images to enhance robustness against adversarial attacks. Guo et al. (2017) [31] delved into various transformations, including image cropping and rescaling, bit-depth reduction, JPEG compression, total variance minimization, and image quilting. These transformations offer unique benefits for defense purposes due to their non-differentiability and inherent randomness, making them effective in perturbing the adversarial noise in input images.

The concept of using image transformations as a defense mechanism against adversarial attacks has gained traction among the research community [44, 101, 104].

Jia et al. (2019) [44] propose a novel approach, where it involves a two-step process of compressing and then rescaling the image data. The essence of their idea of employing such transformations is rooted in the assumption that adversarial attack algorithms exploit inherent patterns in the natural data in order to impair the machine learning systems' integrity. By introducing controlled perturbations to the input image, their method leverages the non-differentiability and randomness inherent in these transformations, to disrupt the harmful patterns that adversarial attacks introduce, thereby bolstering the robustness of deep learning models against such attacks.

PixelDefend, introduced by Song et al. (2017) [101], presents a sophisticated approach to noise reduction in adversarial defense, leveraging the capabilities of PixelCNN++ [95]. The core concept of PixelDefend revolves around the calculation of rectified pixel values for each pixel in an unknown image using the PixelCNN++ model. This process results in the generation of a purified image, which serves as the noise-reduced input for the original targeted model.

The integration of PixelCNN++ into the PixelDefend framework offers an innovative perspective in designing adversarial defense strategies. PixelCNN++ is a state-of-the-art model for image generation. It is unique to re-purpose its usage for the task of injecting carefully calculated values into the images in order to disturb the adversarial noises. PixelCNN++ excels in capturing complex dependencies within images [95], which can be applied to compute rectified pixel values for defense purposes [101]. By calculating the rectified pixel values for images, the PixelDefend approach demonstrates the efficacy of their framework in a principled and effective manner.

The contribution from the research work conducted by Song et al. exemplifies the evolving nature of adversarial defense strategies. They highlight the ongoing research for innovative methods to enhance the security of deep learning models in real-world applications.

Gradient Masking/Obfuscation

Gradient masking, also known as gradient obfuscation, is a technique used in adversarial machine learning to mitigate adversarial attacks. This technique involves deliberately obscuring or hiding the gradient of the loss with respect to the model's input from the attackers [86]. Because most adversarial attacks take advantage of the gradient information to craft adversarial samples, this technique aims specifically to make such computational processes more challenging for crafting adversarial examples.

There are various adversarial defense strategies under the category of gradient masking to modify or obscure the gradient information, mitigating the efficacy of crafting adversarial noises.

Defense Distillation, a methodology proposed by Papernot et al. (2016) [86], draws inspiration from the concept of 'Distillation' originally introduced by Hinton et al. (2015) [38]. This approach seeks to explore the possibility of adjusting parameters in the deep learning models to mitigate the calculation of adversarial noise. Specifically, Defense Distillation investigates the impact of the 'Temperature' parameter in the softmax activation function on the calculation of the gradient of the model's loss with respect to its input.

The 'Temperature' parameter in the softmax function controls the smoothness of the prediction output for all categories. By adjusting the 'Temperature' parameter with higher values, it results in smoother distributions among all categories. Defense Distillation aims to control the calculation of the gradient of the model's loss function with respect to its input to reduce the magnitude of the calculated adversarial noises.

However, it is important to note that while increasing the 'Temperature' parameter can decrease the magnitude of adversarial noises, subduing the level of adversarial perturbation, it may not be effective against attacks that leverage sign information of gradients [86]. The key contribution of Defense Distillation in terms of the adversarial defense strategy is that it demonstrates the importance of understanding the underlying mechanisms of adversarial attacks and the fundamental computation of DNNs, to gain a better insight into defense strategies.

Thermometer Encoding, introduced by Buckman et al. (2018) [9], presents a novel approach to enhancing the robustness of deep learning models by mapping the continuous pixel values of training samples into discrete encodings. This technique endeavors to impede the direct computation of the adversarial noises from using the gradients of the model's loss with respect to the input space. Despite the improvement achieved by Thermometer Encoding, it is important to note that this defense mechanism is not without limitations. Adversarial attacks that rely on random search or iterative methods can still effectively craft adversarial examples even when the input is discretely encoded. Furthermore, achieving optimal performance with Thermometer Encoding requires careful selection and design of the encoding scheme, which sets challenges for the model trained with encoded samples to match the performance of a model trained with regular, continuous-valued samples.

While gradient masking represents a promising avenue for enhancing model security, it is important to note that it is not a foolproof defense and can sometimes lead to reduced model performance or unintended consequences [2]. As such, it is important to evaluate the trade-offs in a defense mechanism carefully, and further research is needed to address its potential limitations.

Provable Defense

Provable adversarial defense represents a paradigm shift in the field of adversarial defense. It offers an innovative perspective for understanding adversarial attacks, with a goal to provide assurance with mathematical guarantees. Unlike traditional defense mechanisms that rely on empirical performance, provable defense approaches aim to provide rigorous analysis within a mathematical model under specific conditions, providing a certain level of confidence for a model to resist adversarial attacks.

This new paradigm in adversarial defense revolves around the idea of formulating the problem of adversarial defense into a mathematical model, from which to analyze and derive the worst-case scenario within the framework. The analysis can provide some theoretical certificate under which the model can be protected against potential attacks.

While a mathematical model and rigorous proof can provide transparency on a model's robustness against adversarial perturbations, the mathematical models are derived often based on certain assumptions. For example, such assumptions include the smoothness of the decision boundary of a trained model in a high-dimensional space, or a vicinity distribution of adversarial samples around a true data point, which helps simplify the mathematical model for rigorous proof but might not provide a whole picture of what happens in the black box. This highlights the need to further the research of provable defense as an emerging field in order to overcome multiple challenges and limitations. Therefore, in order to fit a model's defense mechanism into a certain workable mathematical model, trade-offs are often made on model complexity or feature dimensions, etc. Despite these challenges, the provable adversarial defense paradigm offers unique transparency and clarity for a defense methodology, offering invaluable perspective to attacks and defense mechanisms.

The Certification of Learning with Trees (COLT) framework, proposed by Balunović et al. (2019) [4], represents a sophisticated defense approach. The method combines the concepts of verification and adversarial perturbation. At the core of the COLT framework are two key components: the verifier network and the adversary network.

During each iteration of the training process, the verifier network plays a crucial role in providing certification for the targeted neural network. This certification is achieved through a convex relaxation, which serves as a bounding mechanism for subsequent use. By bounding the network's output within a convex set, the verifier network aims to ensure that the network's predictions remain robust and reliable [4].

In contrast, the adversary network operates with an opposing objective. Within the bounds of the convex set provided by the verifier, the adversary network intends to search for inputs that can potentially mislead the neural network [4]. Because of the search for potential vulnerability of the network in the input space, the weaknesses of the network are exposed. The COLT framework aims to offer a principled and systemic approach to understanding the security of deep learning models.

In the work from Raghunathan et al. (2018) [89], they focused on certifying one-hidden-

layer neural network models to bound the adversarial error within a specified threshold. Their approach relied on the semidefinite relaxation technique, which is approximated as a complex optimization problem [89]. By leveraging semidefinite relaxation, the authors were able to construct a certification procedure. The certification provides a robustness guarantee to the network against adversarial attacks.

Similarly, Wong et al. (2018) [112] proposed a novel adversarial defense method by utilizing a convex outer approximation technique for provable defense. This convex outer approximation served as the basis for an optimization procedure aimed at minimizing the worst-case loss over the output region [112]. The optimization procedure was formulated using linear programming techniques, which allowed for efficient computation under the convex approximation framework.

While the pursuit of provable defense mechanisms represents a promising avenue in the field of adversarial defense, it is important to acknowledge certain limitations and challenges faced by current provable defense methods. One such limitation is that many provable defense methods are designed to prove robustness under a specific type of bound, such as the $L1$-norm, $L2$-norm, or L_∞-norm. This narrow focus on a single type of bound can restrict their applicability to various types of DNN models. And in real-world scenarios, adversarial perturbations may not always conform to the specified bound.

Furthermore, the effectiveness of provable defense methods can be limited to the certified level of adversarial noises. When the magnitude of the adversarial perturbation in the data exceeds the bounded level, the defense mechanism might be compromised. In such cases, the overall defense performance of provable defense methods may not be as effective as heuristic defense approaches that do not rely on strict bounds or assumptions about the type of adversarial attacks.

The ongoing efforts highlight the growing interest in developing provable adversarial defense mechanisms for neural networks [88]. These approaches represent important progress toward DNN models' security and reliability from a more mathematical perspective. By leveraging modeling techniques such as semidefinite relaxation and linear

programming, researchers are able to construct rigorous certification procedures that guarantee the robustness of neural networks against adversarial attacks. The current limitations in the provable defense also underscore the need to expand their applicability and scalability in order for the provable defense to handle a wider range of adversarial attacks.

Detection

Alongside efforts to rectify adversarial samples back to their benign counterparts, there exists a complementary line of defense research dedicated to detecting potential adversarial samples in the model inputs. Before adversarial samples impact deep learning applications, successful detection of those anomalies also serves as an extra layer of protection. Adversarial detection hinges on two critical aspects: correctly identifying adversarial inputs and avoiding misclassification of benign samples. Over the past decade, researchers have proposed a whole genre of detection mechanisms aimed at achieving these objectives [116, 3].

Detection mechanisms have made remarkable progress over the past decade [13], and they vary largely. Different characteristics of the data have been utilized to distinguish real data from their adversarial counterparts, such as statistical analysis of input data, which distinguishes anomalous patterns from those given by the real data. Others can also rely on a learning-based approach, such as using deep learning models to understand the underlying signal from real data to differentiate adversarial samples from real data. Both sophisticated algorithms and heuristic handling can be applied in order to achieve reliability in detecting potential adversarial samples.

One such approach, detailed in the work by Grosse et al. (2017) [29], conducted a comprehensive study on the distributional features difference between adversarial samples and original training data. The key idea of their research revolved around the statistical test designed to distinguish adversarial samples from benign data [29]. The study explores three different models—decision trees, support vector machines, and neural networks—to discern adversarial samples from benign data.

Feinman et al. (2017) [20] propose a novel method for detecting adversarial samples by comparing an input's normal and noisy versions. The authors suggest that a model's confidence in identifying an adversarial attack can be inferred from density estimation on the latent feature space derived from a trained neural network [20]. This approach leverages the distribution differences for normal and adversarial samples to identify potential attacks.

Xu et al. (2017) [117] proposed a detection method based on analyzing the consistency of a trained model's output for a given input. Their approach involves comparing the model's output before and after the input is compressed to a smaller size. Their work suggests that adversarial samples often result in inconsistencies in the model's output, particularly when the input is manipulated to deceive the model [117]. By quantifying these inconsistencies, the authors devised a mechanism for detecting adversarial attacks with high accuracy.

In the study by Goswami et al. (2019) [27], the authors observed significant variations in the intermediate layer outputs of a neural network with and without adversarial samples. Building on this observation, they proposed a detection approach that involves training a Support Vector Machine (SVM) classifier to detect the anomalies in intermediate outputs, distinguishing if they belong to benign samples or adversarial samples. By leveraging the distinct patterns in the intermediate layer representations, their approach aims to accurately identify adversarial samples.

Safetynet, as proposed by Lu et al. (2017) [67], takes a different approach by focusing on the classification of intermediate layer latent features. The authors propose to find the centroid for each class based using the intermediate layer outputs. During the testing phase, the distance of an unknown input's hidden features from each centroid is computed. If the distance exceeds a certain threshold, the input is classified as an adversarial sample. This method capitalizes on the notion that adversarial samples often deviate from the latent feature patterns that are unique to benign data.

In a related approach, Magnet, introduced by Meng and Chen (2017) [72], proposes to utilize a Variational Autoencoder (VAE) trained with benign samples to output reconstruc-

tions similar to those of benign samples. Leveraging these output pairs, Magnet compares the reconstructed output with its input. If the reconstructed output deviates significantly from the input, the input is flagged as potentially adversarial. This method leverages the generative capabilities of VAEs to detect adversarial samples based on the reconstruction error.

These approaches underscore the diversity of detection strategies developed within the research community. By leveraging statistical analysis, noise comparison, output consistency, the nuances of intermediate layer representations, and reconstruction errors, researchers are developing innovative solutions to address the challenges posed by adversarial attacks, to better detect potential harmful inputs for deep learning models.

2.4 Generative Models

Generative models, as a foundational concept in machine learning, are designed to elucidate the underlying data generation process [28]. These models aim to capture the intrinsic relationships and dependencies within the data, making it possible to generate realistic samples closely related to those from the original data distribution. Examples of generative models include Bayesian networks, Variational Autoencoders (VAEs), and Generative Adversarial Networks (GANs), each offering unique approaches to modeling the data-generating process.

In contrast, discriminative models focus on distinguishing between different classes or categories within the data. Rather than modeling the entire data distribution, discriminative models concentrate on learning the decision boundaries that separate one class from another [28]. Examples of discriminative models include logistic regression, decision trees, and Support Vector Machines (SVMs), which are particularly good at identifying underlying patterns and latent features that can be employed to differentiate different classes in the data.

In the realm of probabilistic modeling, if Y is the ground truth labels and X is the

features, a generative model aims to approximate both the prior probability $P(Y)$ and the likelihood $P(X \mid Y)$ in order to infer the posterior probability through Bayes' theorem:

$$posterior = \frac{prior \times likelihood}{evidence} \Leftrightarrow P(Y \mid X) = \frac{P(Y) \times P(X \mid Y)}{P(X)} \tag{2.4.1}$$

In contrast, discriminative models focus solely on estimating the conditional probability $P(Y \mid X)$ directly, without modeling the underlying data generation process. These models learn a mapping function $f : X \rightarrow Y$, such as a classifier, using the training data. The goal of discriminative models is to accurately classify or label data points by estimating the posterior probability $P(Y \mid X)$.

The key distinction between generative and discriminative models lies in their objectives and methodologies. Generative models seek to understand and model the entire data distribution, enabling the generation of new data points based on the learned data distribution. In contrast, discriminative models focus on learning the decision boundaries between different classes in the data, without explicitly modeling the data generation process. Both types of models have unique advantages and applications, depending on the specific task and dataset at hand [28].

Over the past few decades, the field of machine learning has witnessed tremendous advancements in the development of generative models [43]. Generative models, such as Gaussian mixture models, hidden Markov models, decision trees, random forests, Bayesian networks, Boltzmann machines, Variational Autoencoders (VAEs), and Generative Adversarial Networks (GANs), have been studied extensively for modeling data distributions and generating new data samples [80]. Overall, the development of generative models has made significant progress and a diverse set of applications for data generation have been proposed within the research community [32].

The proposed adversarial defense framework leverages the power of Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs) as key components. VAEs and GANs are the two most well-known generative model paradigms that have shown promise in generating realistic data samples and capturing complex data distribu-

tions. This section provides an introduction to VAEs and GANs, highlighting their role in the proposed defense mechanisms and reviewing relevant work from the perspective of adversarial defense.

Variational Autoencoder (VAE)

Variational Autoencoders (VAEs) have been known as a versatile framework with diverse applications across various fields. VAEs are renowned for their ability to perform latent feature extraction, content reconstruction, and dimension reduction tasks [120]. This versatility has led to their widespread adoption in fields such as computer vision, natural language processing, and speech recognition [123, 72, 121].

A distinguishing feature of VAEs is their architecture, which consists of a relatively smaller number of neurons in the middle layer compared to the input and output layers. This design choice enables VAEs to efficiently capture and encode the essential features while maintaining a compact latent representation. Furthermore, VAEs are classified under the family of Bayesian Networks [17, 22], where the middle layer can be designed to approximate certain distributions rather than merely serving as latent features.

Motivation of VAE

In many scenarios, the generative process includes the utilization of unobserved latent variables that are intrinsic to the observed data. Understanding these latent variables and their relationships with the observable data is crucial for tasks, such as analyzing the occurrence patterns of various types of earthquakes or modeling the complex generative processes underlying chemical reactions [43]. For example, in Figure 2.1, a simple form of a generative model is $p(x, z) = p(x \mid z)p(z)$. x belong to an i.i.d dataset $X = \left\{x^{(i)}\right\}_{i=1}^{N}$ that are generated by some random autonomous process. z are latent variables, which are usually not observed by us, thus z usually are not part of the dataset. In order to describe the data of interest, the model evidence (marginal likelihood), $p(x)$, can be calculated by

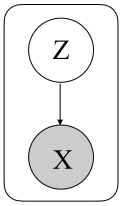


Figure 2.1: A basic probabilistic directed graphical model.

marginalizing the $p(x, z)$ over z:

$$\begin{aligned} p(x) &= \int p(x, z)dz \\ &= \int p(x \mid z)p(z)dz \end{aligned} \tag{2.4.2}$$

The distribution $p(z)$ is conventionally called the prior distribution over z, where in actual application $p(z)$ and/or $p(x \mid z)$ is usually specified according to the application context [51].

However, the marginal likelihood, $p(x)$, of the data in the directed graphical model is in most cases intractable [51] because there is usually no closed form of the integral over $p(z)$ and $p(x \mid z)$ if their distributions are not well known. When $p(z)$ and $p(x \mid z)$ are not complicated, the integral in 2.4.2 might be calculated or analytically solved, but in many cases of real-world models the prior and the conditional likelihood are complicated and there is no efficient analytical solution or computationally efficient way to calculate the integral in 2.4.2.

For the same reason, the posterior distribution for z is in most cases intractable:

$$p(z \mid x) = \frac{p(x \mid z)p(z)}{p(x)} \tag{2.4.3}$$

because $p((z \mid x))$ is directly relevant to the $p(x)$, the intractability of $p(x)$ leads to the intractability of $p((z \mid x))$ and vice versa. Traditional inference techniques to solve integral require expensive computation, such as per-datapoint optimization loop or bad posterior approximation [51]. Therefore, more efficient approximations for the posterior inference

of the latent variables z and for the marginal inference of x are required, especially when the dataset and the number of parameters in the model are large.

However, a good approximation to the posterior distribution $p(z \mid x)$ and marginal likelihood of x can be achieved and trained jointly in the frame of VAE [50], by (1) reforming the optimization of the marginal likelihood of x to the optimization of a variational bound, (2) using Monte Carlo estimator for approximating some distribution's expectation, and (3) replacing complicated posterior distribution $p(z \mid x)$ with an approximation that can be parameterized and differentiable w.r.t. its parameters. VAE is one of the examples to efficiently solve the intractability issue raised by the complicated probabilistic distribution in the directed graphical model.

Variational Bound

One objective for a generative model is to optimize marginal likelihood with respect to its parameters θ to better describe the observations (the dataset, $\mathcal{D}$, where $x \in \mathcal{D}$ and x is generated in an i.i.d. process:

$$\log p_\theta(\mathcal{D}) = \sum_{x \in \mathcal{D}} \log p_\theta(x) \tag{2.4.4}$$

As the true underlying posterior inference, $p_\theta(z \mid x)$, is unknown, it's usually assumed to be in a form similar to some well-known distribution families, e.g., Gaussian distribution, parameterized by ϕ:

$$q_\phi(z \mid x) \approx p_\theta(z \mid x) \tag{2.4.5}$$

Now start from the log-likelihood of the individual observation x:

$$\log p_\theta(x) = log \int_z p_\theta(x,z)dz \tag{2.4.6a}$$

$$= \log \int_z p_\theta(x,z)\frac{q_\phi(z \mid x)}{q_\phi(z \mid x)}dz \tag{2.4.6b}$$

$$= \log(\mathbb{E}_{z\sim q_\phi(z|x)}\left[\frac{p_\theta(x,z)}{q_\phi(z \mid x)}\right]) \tag{2.4.6c}$$

$$\geq \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z \mid x)}\right] \tag{2.4.6d}$$

$$= \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log \frac{p_\omega(x \mid z)p_\theta(z)}{q_\phi(z \mid x)}\right] \tag{2.4.6e}$$

$$= \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log \frac{p_\theta(z)}{q_\phi(z \mid x)}\right] + \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\omega(x \mid z)\right] \tag{2.4.6f}$$

$$= -D_{KL}(q_\phi(z \mid x) \parallel p_\theta(z)) + \mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\omega(x \mid z)\right] \tag{2.4.6g}$$

From 2.4.6b to 2.4.6c, the expectation with certain distribution is derived through Importance Sampling [81]. From 2.4.6c to 2.4.6d, the inequality is derived through Jensen's inequality [118, 71]. From 2.4.6d to 2.4.6e, a Bayesian theorem [5] is used to derive the product of the prior distribution and conditional likelihood of evidence. From 2.4.6f to 2.4.6g, the expectation w.r.t. the true distribution on the log of the ratio of proposal distribution and true distribution is equivalent to the KL Divergence [55] of these two distributions.

It becomes obvious that the equation 2.4.6g is a lower bound of $\log p_\theta(x)$. To optimize the original evidence likelihood, we can instead optimize this lower bound.

Monte Carlo Estimator

The expectation of the log-likelihood in function 2.4.6g can be estimated using Monte Carlo estimator [51, 74]:

$$\mathbb{E}_{z\sim q_\phi(z|x)}\left[\log p_\omega(x \mid z)\right] \simeq \frac{1}{L}\sum_{l=1}^{L}\log p_\omega(x \mid z^{(l)}) \tag{2.4.7}$$

As $p_\omega(x \mid z)$ is specified and known to us, the optimization of the $p_\omega(x \mid z)$ with respect to its parameters can be solved using stochastic gradient ascent with mini-batch.

Reparameterization Trick

The true underlying distribution of the latent variable $p^*(z)$ is unknown to us, but it's usually assumed and specified to be one of the well-known parameterized distribution families, such as Gaussian distribution. In practice, the true prior distribution for z is assumed to be the centered isotropic multivariate Gaussian $p_\theta(z) = \mathcal{N}(z; 0, I)$ [50], and the form for the proposed posterior model is also assumed to be a multivariate Gaussian: $q_\phi(z \mid x) = \mathcal{N}(z; \mu, \sigma^2 I)$. We can see the difference between these two Gaussian distributions is due to whether the z's distribution is conditioned on data x.

To reduce the the KL Divergence $D_{KL}(q_\phi(z \mid x) \parallel p_\theta(z))$ in equation 2.4.6g is to find the parameters in $q_\phi(z \mid x)$ such that the distribution of $q_\phi(z \mid x)$ is close to that of $p_\theta(z)$. With the assumption that both $q_\phi(z \mid x)$ and $p_\theta(z)$ to be multivariate Gaussian distribution, the KL divergence can be calculated analytically.

Because $p_\theta(z) = \mathcal{N}(z; 0, I)$ and $q_\phi(z \mid x) = \mathcal{N}(z; \mu, \sigma^2 I)$, first:

$$\int q_\phi(z \mid x) \log p_\theta(z) dz = \int (-\frac{1}{2} \log (2\pi) - \frac{1}{2} z^2) q_\phi(z \mid x) dz \tag{2.4.8a}$$

$$= -\frac{1}{2} \log (2\pi) \int q_\phi(z \mid x) dz - \frac{1}{2} \int z^2 q_\phi(z \mid x) dz \tag{2.4.8b}$$

$$= -\frac{1}{2} \log (2\pi) - \frac{1}{2} (\mu^2 + \sigma^2) \tag{2.4.8c}$$

from 2.4.8b to 2.4.8c, as $q_\phi(z \mid x)$ follows normal Gaussian distribution, $\int q_\phi(z \mid x) dz = 1$. Also from 2.4.8b to 2.4.8c the expectation of z^2 is equal to $\mu^2 + \sigma^2$ according to $\mathbb{E}\left[X^2\right] = V[X] + (\mathbb{E}[X])^2$.

Second:

$$\int q_\phi(z \mid x) \log q_\phi(z \mid x) = \int (-\frac{1}{2} \log (2\pi\sigma^2 - \frac{1}{2}\frac{(z-\mu)^2}{\sigma^2}))q_\phi(z \mid x)dz \tag{2.4.9a}$$

$$= -\frac{1}{2} \log (2\pi\sigma^2) \int q_\phi(z \mid x)dz - \frac{1}{2}\frac{1}{\sigma^2} \int (z-\mu)^2 q_\phi(z \mid x)dz \tag{2.4.9b}$$

$$= -\frac{1}{2} \log (2\pi\sigma^2) - \frac{1}{2}\frac{1}{\sigma^2}\sigma^2 \tag{2.4.9c}$$

$$= -\frac{1}{2} \log (2\pi) - \frac{1}{2}(1 + \sigma^2) \tag{2.4.9d}$$

Note that from 2.4.9b to 2.4.9c, $\int (z-\mu)^2 q_\phi(z \mid x)dz$ is the definition of variance, thus, it's equal to σ^2.

Therefore [50]:

$$-D_{KL}(q_\phi(z \mid x) \parallel p_\theta(z)) = \int (\log p_\theta(z) - \log q_\phi(z \mid x))q_\phi(z \mid x)dz \tag{2.4.10a}$$

$$= \frac{1}{2}(1 + \log \sigma^2 - \mu^2 - \sigma^2) \tag{2.4.10b}$$

Thus, the computation of the KL Divergence part in equation 2.4.6g can also be analytically solved. All the parameters in the model can be optimized using, e.g., stochastic gradient ascent and mini-batch.

In the realm of deep learning, the approach differs slightly from traditional statistical methods. Instead of explicitly defining probabilistic density functions as in classical statistics, deep learning typically involves specifying a parameterized distribution family, such as a Gaussian distribution characterized by its mean and standard deviation. This parameterization allows for greater flexibility in modeling complex data distributions. In practice, deep learning models learn the parameters of these distributions through the training process. Rather than explicitly calculating the exact multiplication of different density functions, the training process involves optimizing these parameters with respect to some objective function, such as maximizing the likelihood of the observed data given the model parameters.

Generative Adversarial Networks (GAN)

One of the main purposes of a generative model is to learn how dataset x are distributed, e.g., approximating the true underlying data distribution $p^*(x)$ by building a model, $p_{model}(x)$. The $p_{model}(x)$ is optimized in order to approximate $p^*(x)$ as closely as possible.

The model $p_{model}(x;\theta)$ is usually parameterized by some θ and optimizing the model is equivalent to searching for the values for θ such that the value of $\sum_{i=1}^{n} \log p_{model}(x^{(i)};\theta)$ for all the data collected is maximized. Traditionally, some common practices for the optimization include maximum likelihood estimation is used to minimize KL Divergence between $p^*(x)$ and $p_{model}(x;\theta)$ or explicit density modeling [25].

The challenge of intractability in building generative models has led traditional methods to focus on designing models with tractable density functions. One branch effort is to develop learning algorithms based on computationally tractable distribution families. However, as the complexity and dimensionality of data increase, traditional methods face limitations in effectively building and solving generative models. With the emergence of high-resolution image data, video sequences, and other high-dimensional hybrid data types, traditional approaches struggle to capture the underlying data distribution accurately. The complexity of these datasets requires more sophisticated generative models to learn the representation, intricate relationships, and structures hidden in the data.

The advancement in machine learning and deep learning has enabled researchers to construct models with a vast number of parameters, such as deep neural networks, making it possible to approximate highly complex functional forms. In this context, Generative Adversarial Networks (GANs) have emerged as a powerful paradigm for data generation, which demonstrates an excellent capability for modeling complex and high-dimensional data distribution. Unlike traditional generative models that require designing a tractable density function, GANs sidestep this challenge by focusing on learning a tractable sample generation process, which is achieved through a unique framework, that consists of two neural networks: a generator and a discriminator [25].

Inspired by the game theory, GANs consist of two neural networks to play a min-max game such that both players (the generator and discriminator networks) improve each other w.r.t. their own objective functions [25]. Specifically, the first neural network is called the generator, $G(z)$, which maps some latent vector z (usually random noise from a uniform of normal distribution) to data x, and the second neural network is called discriminator, $D(x)$, which maps the data x to a scalar value that approximates how much likely the input x belong to the dataset. The purpose of $G(z)$ is to generate data that can mislead the discriminator, and the discriminator aims to distinguish whether the generated images from $G(z)$ are created by the generator or they are indeed from the real dataset. The objective of training a GAN is defined as follows [43]:

$$\min_G \max_D \mathbb{E}_{x\sim p(x)} \left[\log D(x)\right] + \mathbb{E}_{x\sim p(x)} \left[\log \left(1 - D(G(z))\right)\right] \tag{2.4.11}$$

In the context of Generative Adversarial Networks (GANs), achieving a well-performing generator is characterized by reaching a stage in training where the discriminator can no longer differentiate between generated artificial data and real data from the natural dataset [25]. This state is known as the Nash equilibrium, which implies that the generator has learned how to produce samples that closely resemble those from the natural dataset [43]. Reaching the Nash equilibrium indicates that the generator has effectively learned the data generation process. At this stage, the underlying data distribution has also been learned through training, which is similar to the real data distribution in the natural dataset. The generator can produce high-quality samples that are indistinguishable from the real data [25].

Various structures of GANs have been proposed in the literature over the past decade [82]. The design of the structure of the neural networks depends on specific applications and purposes, such as image style transferring [49], image completion [42], domain conversion [91], and so on. With the advancements in the diversity of GANs' design, GAN refers more to a training scheme in the deep learning context, that there is a discriminator-like component that tries to determine if the input is from real data or fake data and a data

generation process that aims to produce data as close to the real data as possible.

Chapter 3

Spacial Frequency Loss-VAE and Bayesian Update With Collective Voting

3.1 Introduction

Over the past several years, Deep Neural Networks (DNNs) have demonstrated remarkable performance in various computational learning tasks, including video analysis [37], image recognition [53], and audio processing [48]. These advancements have significantly contributed to the progress of deep learning and machine learning models to perform tasks that were once considered challenging for traditional approaches.

Despite their impressive capabilities, DNNs are vulnerable to adversarial attacks. Adversarial attacks exploit the sensitivity of DNNs to small, carefully crafted perturbations in input data [26]. By manipulating a few or all pixels in an image, an attacker can deceive the classifier into making incorrect predictions [105]. This susceptibility to adversarial noise raises concerns about the robustness and reliability of DNNs in real-world applications [26].

Various defense mechanisms have been proposed to mitigate the adversarial attacks on images [2, 96, 86, 61]. These mechanisms can be broadly classified into three different

genres: (1) the training process mixes adversarial images with original images as the training set to make the classifier more robust to adversarial attacks [26, 105]; (2) use high-level latent features from DNNs for clustering either to do anomaly detection or to classify the category [27, 67]; and (3) use DNNs to reconstruct the images using generative networks [72, 96, 30].

Most of the previous work assumes that only the deep learning model itself is targeted during an adversarial attack, while the whole defense framework is unknown to the adversary [2, 96, 86, 61]. But if an adversary has access to the entire defense framework, he can also take advantage of this knowledge to circumvent the defense framework.

3.2 Chapter Contribution

In order to explore the impact of an adversary who knows the defense framework and a potential solution to mitigate such attacks, our approach combines both randomization and discretization through VAE and post-VAE collective voting such that an adversary could not easily use back-propagation to attack the entire framework. We trained multiple post-VAE classifiers to independently output their prediction of those images that are reconstructed from VAE, and a majority vote is cast by a number of randomly selected post-VAE classifiers. The voting result is taken into consideration for the final classification. Given the targeted classifier's prediction on the reconstructed images and the post-VAE voting result, the final prediction is driven by combining the above voting result and the targeted model's prediction as evidence within the Bayesian Update framework.

Additionally, the quality of reconstructed images is improved by incorporating Spatial Frequency Loss (SFL) into the VAE training scheme. Traditionally, a VAE is trained by adding noise to input to reduce the effect of adversarial perturbations, but this leads to blurred reconstructed images and those blurred images negatively affect their usability for re-classification. We depart from this practice in the following ways: (1) our method guides the reconstructed images to form a sharper edge by imposing Spatial Frequency

Loss (SFL) between input and output images during training; (2) our approach forces the learned distribution at the middle layer to form a multivariate Gaussian distribution with nearly 0 mean but a relatively higher variation, allowing a higher tolerance to the adversarial noise in input.

We found that a VAE trained with our approach produces sharper reconstruction images. We also found that not only do the randomization and discretization in the collective voting process defend well against a potential second adversarial attack but they also consistently result in better performance compared to using a single post-VAE classifier.

The main contributions of the proposed work are summarized below:

- A new defense framework is introduced that takes advantage of spatial frequency loss to reconstruct sharper images from adversarial samples. The design enhances the usability of those more accurate reconstructed images for post-VAE classification.

- The proposed method is designed to integrate a post-VAE classification voting mechanism, which utilizes both randomization and discretization to defend against adversarial attacks on the targeted classifier. Furthermore, the majority voting design also shows a positive result against a second type of attack where the adversary attacks the entire defense framework.

- A collective voting strategy using multiple miscellaneous post-VAE classifiers which consistently enhances the overall accuracy. Both the voting result and the targeted classifier's prediction on reconstructed images are considered simultaneously using the Bayesian Update. This design does not only show a consistent extra improvement in classification accuracy between 3% to 5% but also highlights its portability and reusability in any defense framework that needs to incorporate multiple sources for final decision.

- To the best of our knowledge, this is the first work to compare multiple defense mechanisms under a wide spectrum of adversarial noise levels and under four differ-

ent mainstream adversarial attacks. To demonstrate the effectiveness of our defense mechanism, we conduct an extensive experimental study to compare it with two well-known existing defense mechanisms on two well-known data sets: Fashion-MNIST and CIFAR10.

3.3 The Proposed Method

The defense framework features three main components: (1) a Spatial Frequency Loss (SFL) enhanced VAE for noise reduction; (2) a group of post-VAE classifiers; (3) a Bayesian Update unit, as shown in Figure 3.1.

Let $X = \left\{x^{(i)}\right\}_{i=1}^{N}$ denote the clean unperturbed training data, and $X_{adv} = \left\{x_{adv}^{(i)}\right\}_{i=1}^{N}$ the adversarial counterparts. The VAE reduces the possible noises in the input images by reconstructing them: $f_{VAE} : X \rightarrow X^{\sim}$. This process first maps an image to its latent feature z, whose distribution is $P(z|x)$ given the input image. Then the latent feature vector is sampled from this distribution and the sampled latent features are used to reconstruct the conditional likelihood distribution $P(x|z)$ given z.

For a classifier, the ground truth is $Y = \left\{y^{(i)}\right\}_{i=1}^{N}$, and there are up to k categories in the data set. A classifier is denoted as $f_c : x \rightarrow \underset{j}{argmax}\ c_j,\ \ c_j \in [0,1],\ \ j = \{0,1,...,k\}$ and c_j is the probability output from the classifier's softmax layer for category j.

To distinguish the notations between the original classifier and the post-VAE classifiers, f_c^* is used to denote the original one and a numerical superscript to denote different post-VAE classifiers $F_c = \left\{f_c^{(i)}\right\}_{i=1}^{M}$. The same notation mark is used for the original classifier's predictions c_j^* and that of post-VAEs', c_j.

After a reconstructed image $x^{\sim}$ is fed into post-VAE classifiers, each post-VAE classifier gives its prediction j for the image. Then a majority vote is taken among the predictions from post-VAE classifiers. R out of M of the predictions will be randomly chosen from $J = \left\{f_c^{(i)}(x^{\sim})\right\}_{i=1}^{M}$. If two or more classes receive the same number of majority votes, the tie will be randomly broken.

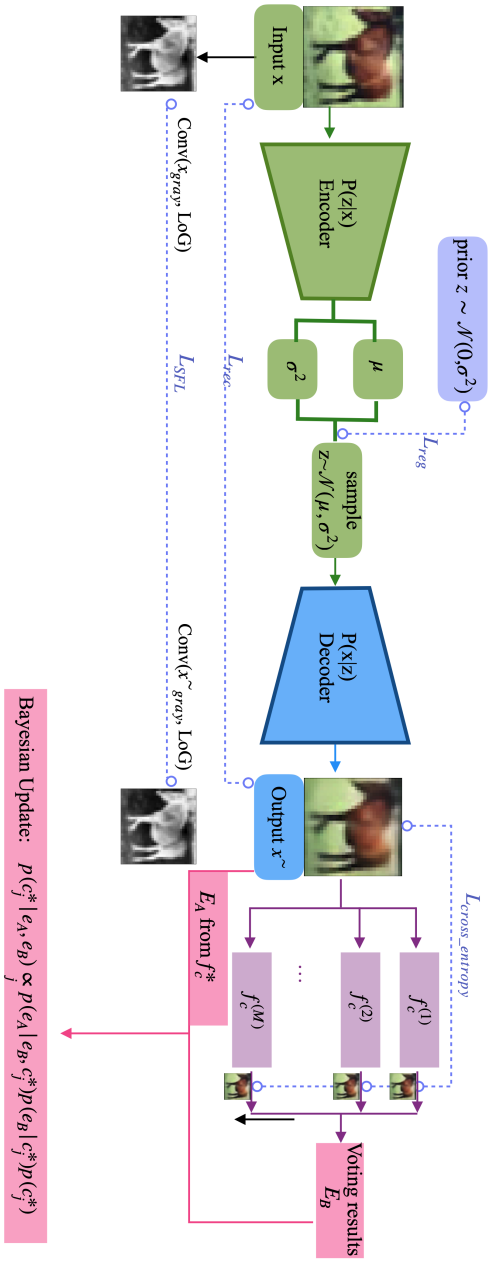


Figure 3.1: The pipeline of the defense mechanism.

To update the original classifier's prediction for all k categories, the Bayesian Update rule is applied using both the original classifier's prediction, $e_A = f_c^*(x^{\sim})$ and the majority

voting result e_B. Thus, the final prediction on a testing image is given by:

$$\underset{j}{argmax}\ \ p(c_j^*|e_A, e_B) \propto_j p(e_A|e_B, c_j^*)p(e_B|c_j^*)p(c_j^*) \tag{3.3.1}$$

3.3.1 VAE

VAE features: (1) an **Encoder** that approximates $P(Z|X)$, that given the input data, X, it outputs the parameters of the conditional probabilistic density function of the unobserved hidden variables (the latent features) Z; (2) a **Decoder** that approximates $P(X|Z)$, that given the latent feature Z (with probability distribution $P(Z)$), it converts the hidden features back to the observable data. The encoder ($Q_\theta(Z|X)$), the decoder ($P_\omega(X|Z)$), and latent feature ($P_\psi(Z)$) are specified to be some parameterizable distribution family, e.g., Gaussian, such that the deep learning models output the parameters instead of direct probabilistic density function for their distribution. By optimizing the variational lower bound of VAE (shown in Equation 2.4.6), the decoder is encouraged to reconstruct images with a similar distribution as the training input, and the encoder is encouraged to generate latent features close to the prior $P_\psi(Z)$:

$$\begin{aligned}(\theta^*, \omega^*, \psi^*) = \underset{(\theta,\omega,\psi)\in\Theta\times\Omega\times\Psi}{argmax} & \mathbb{E}_{z\sim Q_\theta(Z|X)} \log P_\omega(X|Z) \\ & - D_{KL}\left[Q_\theta(Z|X) \parallel P_\psi(Z)\right]\end{aligned} \tag{3.3.2}$$

Maximizing $\mathbb{E}_{z\sim Q_\theta(Z|X)} \log P_\omega(X|Z)$ is equivalent to generating images that are as close as the observable data, given the latent features Z which is generated by the learned encoder $Q_\theta(Z|X)$. The loss function to measure the difference is denoted as reconstruction loss, $\mathcal{L}_{\mathbf{rec}}$, and the form we adopt is the Mean Square Error (MSE). Thus, the parameters can be calculated [93]:

$$\omega^* = \underset{\omega\in\Omega}{argmin}\mathbb{E}_{z\sim Q_\theta(Z|X)}\left[\frac{(P_\omega(X|Z) - X)^2}{2}\right] \tag{3.3.3}$$

In equation (3.3.2), the marginal distribution of hidden features, $P_\psi(Z)$, is selected based on different situations of interest [65]. We adopt zero-centered independent Gaussian distribution, in which its mean and variance are the parameters. We empirically choose the dimension k for the hidden feature Z, such that the outputs from $Q_\theta(Z|X)$ (the mean vector and variance vector) have the same dimension as that of $P_\psi(Z)$. $\mathcal{L}_{\mathbf{reg}}$ is used to denote $D_{KL}\left[Q_\theta(Z|X) \parallel P_\psi(Z)\right]$ as regularization loss.

3.3.2 Spatial Frequency Loss (SFL) for VAE

Using MSE as the loss function only to enforce reconstruction of images tends to blur the images [41]. There are several causes that lead to this blurred effect on reconstructed images:

- The kernels in a convolution deep network learn to present different features, and MSE emphasizes more noticeable features, such as the orientation of lines or the color of a patch, compared to fine-grained textures or sharp edges.

- Note that in equation (3.3.3), MSE actually compares observable data X with the mean of the distribution of reconstructed output $P_\omega(X|Z)$, and the output from $P_\omega(X|Z)$ is the mean of all possible reconstructed outputs given Z. There is inherent noise in the data collection stage, which normally leaves noises in the images captured, thus, by averaging the difference through MSE during training, the learned generative network does not always reflect exact noise in texture or edge in the input image.

- Before Z is fed into the $P_\omega(X|Z)$, it is sampled based on the learned distribution $Q_\theta(Z|X)$. The decoder needs to have the capability to handle all possible sampled features for the input, therefore, during training, the decoder faces a trade-off between mirroring the exact input and reconstructing an image with lower MSE loss, which usually leads to a blurry image.

In order to effectively capture high spatial frequency features, such as sharper edges, in reconstructed images, it is crucial to incorporate an additional loss function. This loss function should emphasize the importance of edge sharpness between the input and output images, guiding the learning process to prioritize the accurate representation of such features. Laplacian of Gaussian (LoG) is an edge detection technique [70], which not only detects brightness intensity changes in an image but also detects the intensity changes at different scales. In this work, the generative network design integrates LoG into the high spatial frequency loss for driving the extraction of sharp edges for both X and $X^{\sim}$, where the LoG kernel is calculated from below:

$$LoG = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.3.4}$$

where x and y are the pixel coordinates from the kernel's center and σ is the scale that controls the fineness of the edges to be detected.

Finally, the overall loss for training the deep learning model includes: (1) a spatial frequency loss, $\mathcal{L}_{\mathbf{SFL}}$, between input X and reconstruction $X^{\sim}$, (2) reconstruction loss for decoder's output and (3) The KL Divergence for the encoder:

$$\begin{aligned}
\mathcal{L} &= \mathcal{L}_{rec} + \mathcal{L}_{reg} + \mathcal{L}_{SFL} \\
\mathcal{L}_{rec} &= \mathbb{E}_{z\sim Q_\theta(Z|X)}\left[\frac{(P_\omega(X|Z) - X)^2}{2}\right] \\
\mathcal{L}_{reg} &= D_{KL}\left[Q_\theta(Z|X) \parallel P_\psi(Z)\right] \\
\mathcal{L}_{SFL} &= \left\|Conv(X_{gray}, LoG) - Conv(X^{\sim}{}_{gray}, LoG)\right\|_2
\end{aligned} \tag{3.3.5}$$

3.3.3 Post-VAE Classification and Voting

Many white-box attacks [26, 84, 56, 68] formulate the adversarial attacks as an optimization problem. The adversarial noise is calculated by minimizing the adversarial noise while maximizing the loss function of the targeted classifier. In most cases, the structure and parameters of a classifier $f_c(x)$ provide the necessary information to craft unnoticeable and small adversarial noise.

Previously, many defense mechanisms [97, 61, 67, 86, 63] consider only the case when an adversary attacks the original classifier mainly. For example, Defense-GAN [97] uses GAN to generate a new image for the original classifier, $f_c(G(z))$ and Defense-VAE [61] retrains the original classifier using the reconstructed images that are generated by VAE, $f_c(f_{VAE}(x))$. These methods only consider the case that $f_c(x)$ is being attacked, however, they did not explore the case when the adversary has access to the entire framework, e.g., attacking $f_c(f_{VAE}(x))$ or $f_c(G(z))$ as a whole. These defense mechanisms also largely utilize deep learning models as part of the defense framework, and it is possible for the adversary to optimize their adversarial attacks by exploiting the network information in the defense framework, potentially impairing the integrity of the defense mechanism.

To defend against the case when the entire framework is being attacked, the proposed method introduces randomization and discretization in the decision-making process. All post-VAE classifiers are designed to have different structures and hyperparameters, and they are trained using reconstructed images $X^{\sim}$. After images are reconstructed from VAE, R out of M post-VAE classifiers are randomly selected to classify $X^{\sim}$. Then a majority vote is taken from the R predictions. Thus, although an adversary has access to VAE and all post-VAE classifiers and it can craft strong adversarial samples using the information in $f_c^{(i)}(f_{VAE}(x))$, $i = 1, 2, ..., M$, the adversarial sample crafted using $f_c^{(i)}(f_{VAE}(x))$ does not work as effectively as it will on $f_c^{(j)}(f_{VAE}(x))$ when $i \neq j$, because $f_c^{(i)}$ and $f_c^{(j)}$ have different network structure and parameters, which affects the noise crafted by the adversary. Although there is a possibility of transferability issue where the same adversarial sample could attack different deep networks [85], the post-VAE classifiers should be designed in a way to differentiate themselves in terms of structure, hyperparameters, optimizer, etc. to mitigate this issue.

Another advantage of using multiple post-VAE classifiers is that the different structures of classifiers tend to converge at different local optima after training. For some $X^{\sim}$, there could be disagreements of the predictions among all post-VAE classifiers due to the local optima issue and the random sampling process of getting the latent feature Z. However,

not every $f_c^{(i)}$ makes a wrong prediction for $X^{\sim}$, and if there is a majority of $f_c^{(i)}$ making a correct prediction, the collective voting process can correct the wrong predictions from the smaller group of post-VAE classifiers. We analyzed this aspect in section 3.4.

3.3.4 Combine VAE Reconstruction and Voting Result Using Bayesian Update

The image reconstructed by the VAE, $X^{\sim}$, is first fed into the original classifier for prediction, and this prediction is used as evidence e_A for later use in the Bayesian Update. Then the reconstructed image will be predicted by a group of post-VAE classifiers, and the prediction that receives the vote from the collective voting process is used as the second evidence e_B in Bayesian update.

For general multi-evidence cases, here denotes the marginal likelihood of evidence with $P(E_A = e_A)$, $P(E_B = e_B)$, ..., $P(E_M = e_K)$, where e_K is the Kth type of evidence one can find from the data. The posterior from the Bayesian update is calculated and is used as the final prediction for an unknown input image:

$$\begin{aligned} & p(c_j|e_A, e_B, ..., e_M) \\ & = \frac{p(e_A|e_B, ..., e_M, c_j)...p(e_M|c_j)p(c_j)}{\sum_j p(e_A|e_B, ..., e_M, c_j)...p(e_M|c_j)p(c_j)} \\ & \underset{j}{\propto} p(e_A|e_B, ..., e_M, c_j)...p(e_M|c_j)p(c_j) \end{aligned} \tag{3.3.6}$$

3.4 Experiments

3.4.1 Experimental Setup

To evaluate the performance and efficacy of the proposed defense mechanism, the experiments compared it with other existing methods, tested using two data sets, Fashion-MNIST [114] and CIFAR10 [6]. Fashion-MNIST contains 10 classes of hand-written digits, with 60,000 training samples and 10,000 testing samples. Each sample is a 28×28

gray-scaled image. CIFAR10 contains 10 categories of animals and objects, with 50,000 training samples and 10,000 testing samples, and each sample is an RGB image with a size of $32 \times 32 \times 3$.

For adversarial attacks, Fast Gradient Sign Method (FGSM)[26], Basic Iterative Method (BIM) [56], Projected Gradient Descent (PGD) [68], Carlini and Wagner (CW) Method [12] were used in the evaluation.

Various defense mechanisms have been proposed during recent years [2, 67, 72, 96, 86, 63, 23], and this section compares the proposed defense mechanism with two state-of-the-art methods that are more relevant to this work, **Defense-VAE** [61] and **Defense-GAN** [96].

The VAE's structure is shown in Table (A.1) in Appendix A. For post-VAE classifiers, we adopted three different types of networks (Residual-Networks [33], Wide-Residual-Networks [119] and DenseNet [40]). By adjusting the structure parameters, 4 different structures of each type were created (12 in total) and used as post-VAE classifiers for voting. The structures of all 12 post-VAE classifiers are shown in Table (B.2, B.3, B.4) in Appendix 6. The implementation of the classifiers is inspired by a public GitHub repository [60]. The Conditional Probability Tables (CPT) for $P(e_A|x^{\sim})$ and $P(e_B|e_A, x^{\sim})$ for Bayesian Update were constructed by calculating the statistics using the entire training dataset.

We conducted our experiments on a workstation equipped with an Nvidia GeForce RTX 3080 GPU, which has 8704 CUDA cores and 10 GB of GDDR6X memory, running on a Linux-based operating system (Ubuntu 20.04). The CPU equipped in the system is an Intel Core i9-10900K CPU with 10 cores and 20 threads.

3.4.2 Reconstructed Image Quality using SFL

Because the reconstructed images play a crucial role in classification tasks, it is essential to generate a close resemblance to the original images. The Spatial Frequency Loss (SFL) is introduced to enhance the training process, aiming for more detailed and rich

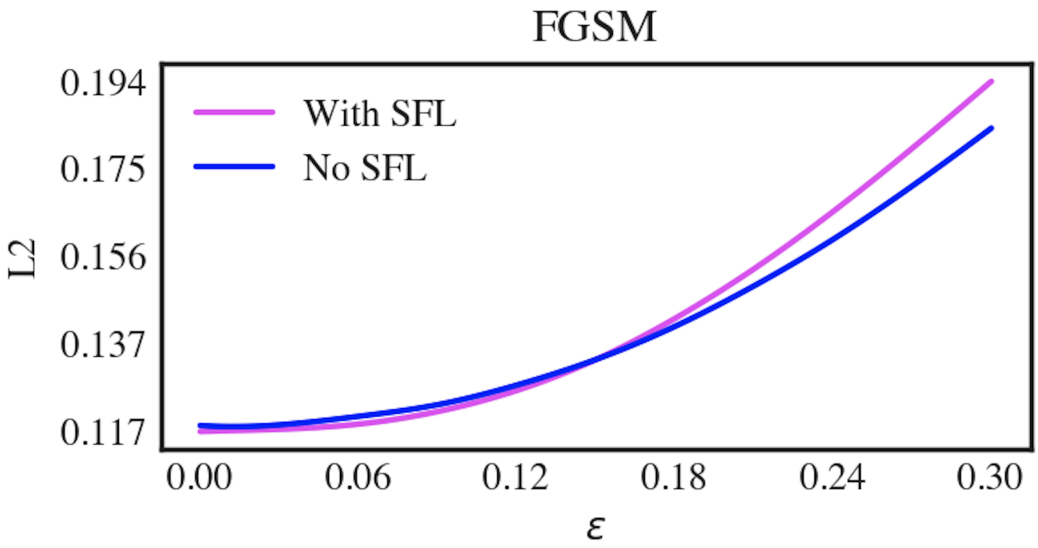


Figure 3.2: L2 difference between original training data and the reconstructed images for SFL-VAE and Non-SFL-VAE, when it's under FGSM attack.

reconstructions. This section conducts a comparative analysis of the reconstruction quality of images generated by Variational Autoencoders (VAEs) trained with and without the Spatial Frequency Loss ($\mathcal{L}_{SFL}$).

We adopt the averaged L2 norm difference as a metric for quantifying the dissimilarity between two groups of data. The experimental comparison involves measuring the L2 norm difference between the original image data and the reconstructed images generated by the VAE with and without the enhancement by SFL. This analysis provides insights into the accuracy of the reconstruction process and the effectiveness of the SFL in enhancing reconstruction quality. Figures 3.2, 3.3, 3.4, and 3.5 show the quantitative comparisons of the image quality difference between SFL-VAE and Non-SFL-VAE when it's under FGSM, BIM, PGD, and CW attacks respectively.

From the observation, across four different adversarial attacks and varying levels of perturbation, the reconstructed images from the VAE trained with the Spatial Frequency Loss ($\mathcal{L}_{SFL}$) consistently exhibit a lower L2 norm difference from the original clean

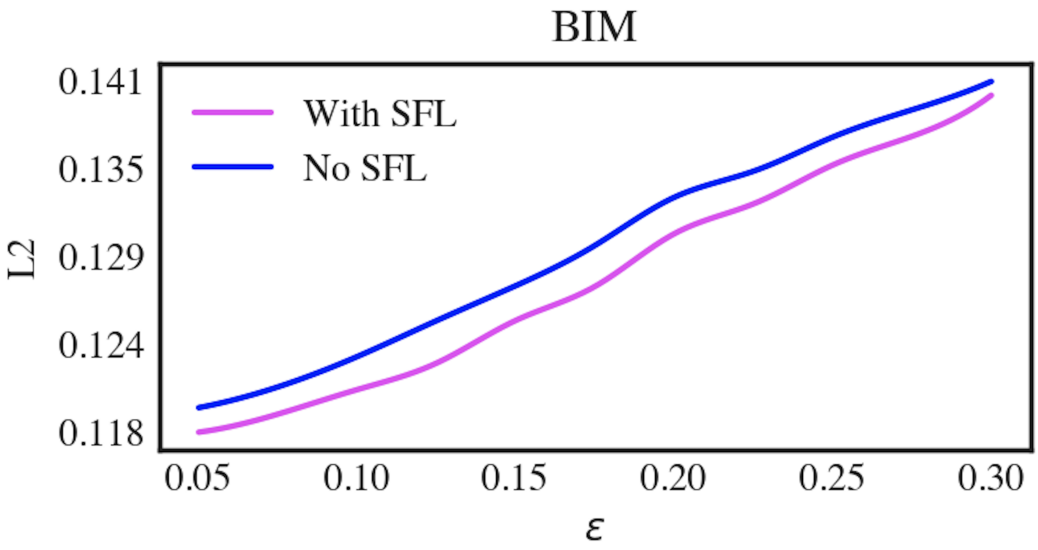


Figure 3.3: L2 difference between original training data and the reconstructed images for SFL-VAE and None-SFL-VAE, when it's under BIM attack.

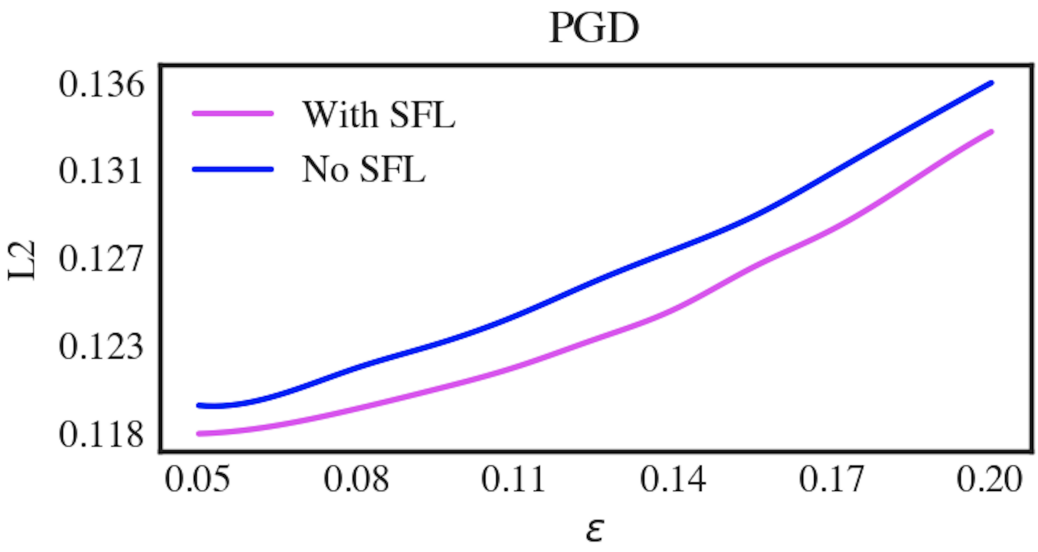


Figure 3.4: L2 difference between original training data and the reconstructed images for SFL-VAE and None-SFL-VAE, when it's under PGD attack.

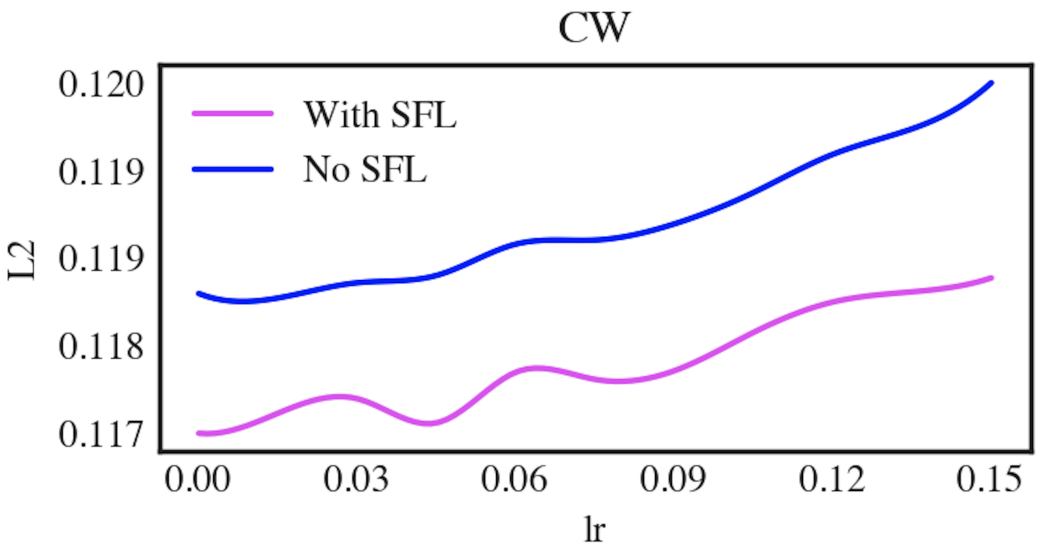


Figure 3.5: L2 difference between original training data and the reconstructed images for SFL-VAE and None-SFL-VAE, when it's under CW attack.

images. This lower averaged L2 norm difference indicates greater similarity between the reconstructed images and the original images when the generative network is trained with Spatial Frequency Loss (SFL-VAE).

By observing Figures 3.2, 3.3, 3.4, and 3.5, this trend holds across the board, with the exception that when $\epsilon \geq 0.15$ in the Fast Gradient Sign Method (FGSM) attack. Notably, this threshold also coincides with a discernible decrease in the accuracy of post-VAE classifiers following the $\epsilon = 0.15$ perturbation level in the FGSM attack, as illustrated in Figure 3.6. These findings underscore the efficacy of the $\mathcal{L}_{SFL}$ training strategy in enhancing the reconstruction quality of SFL-VAE, regularizing the impact of adversarial noise under various perturbation levels.

3.4.3 White-box Attacks on Classifiers

Out experiments conducted evaluations on the effectiveness of the proposed defense mechanism and its performance against other state-of-the-art noise-reducing defense meth-

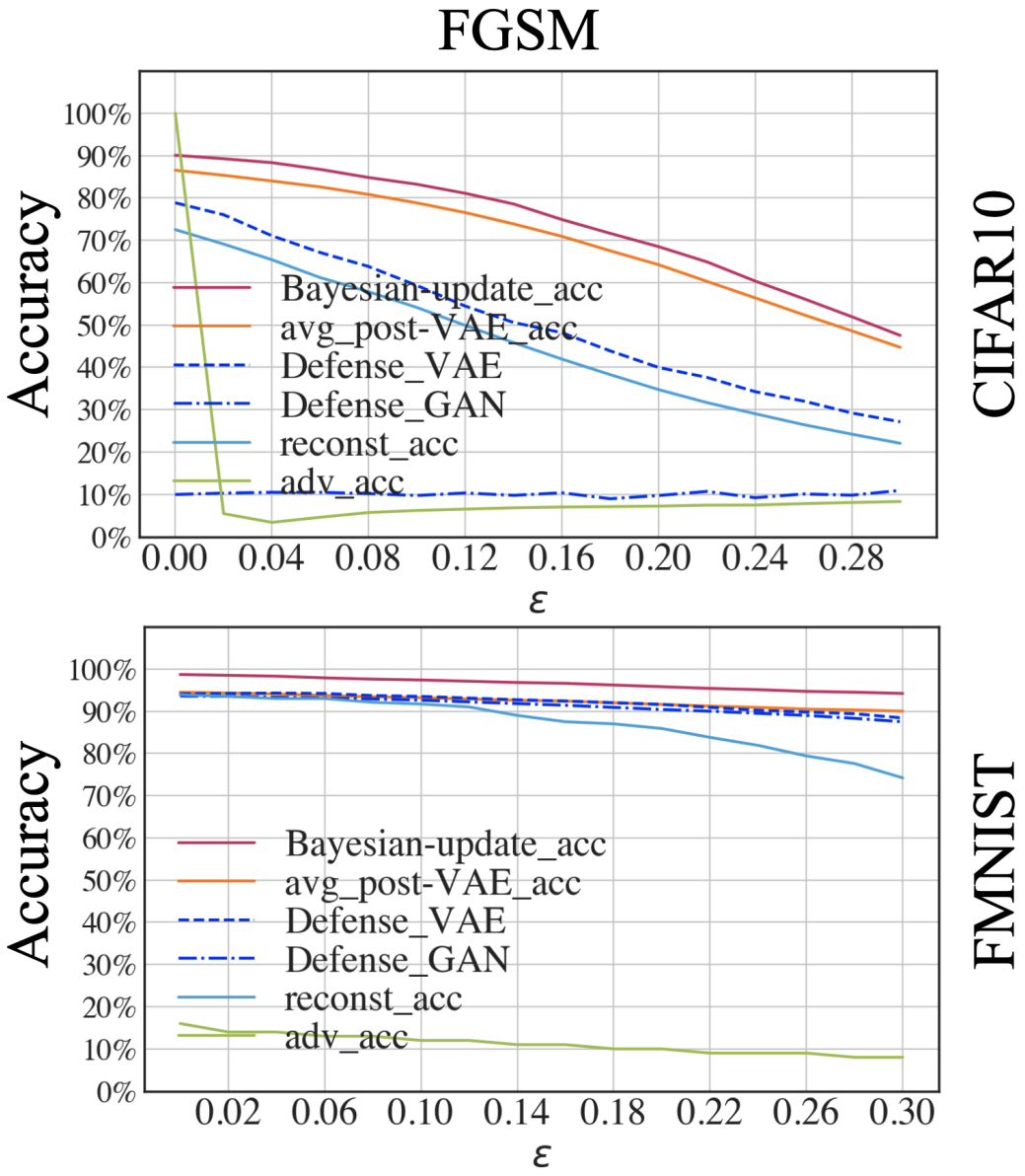


Figure 3.6: Comparison of different defense methods under FGSM attack on CIFAR10 and Fashion-MNIST datasets.

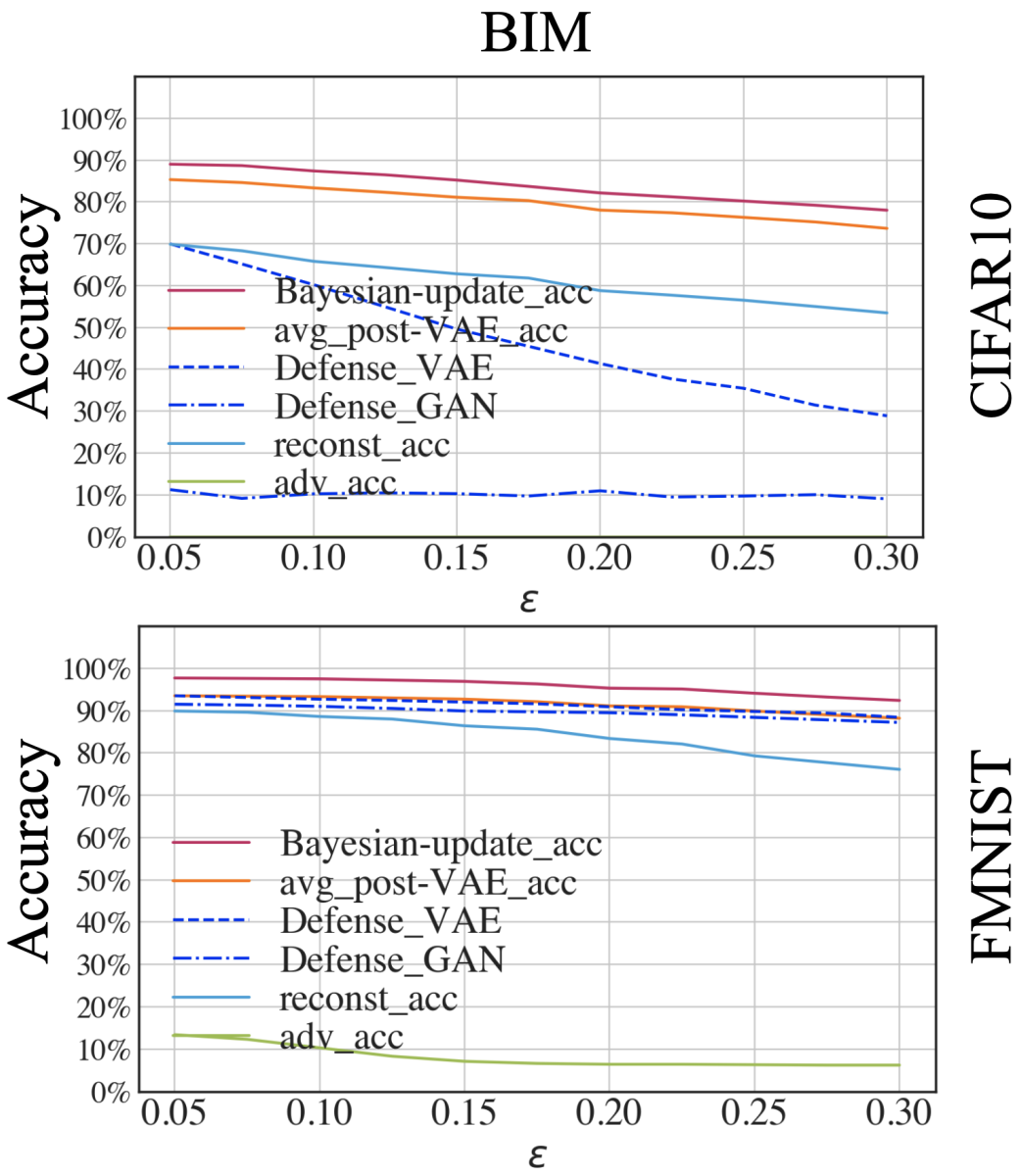


Figure 3.7: Comparison of different defense methods under BIM attack on CIFAR10 and Fashion-MNIST datasets.

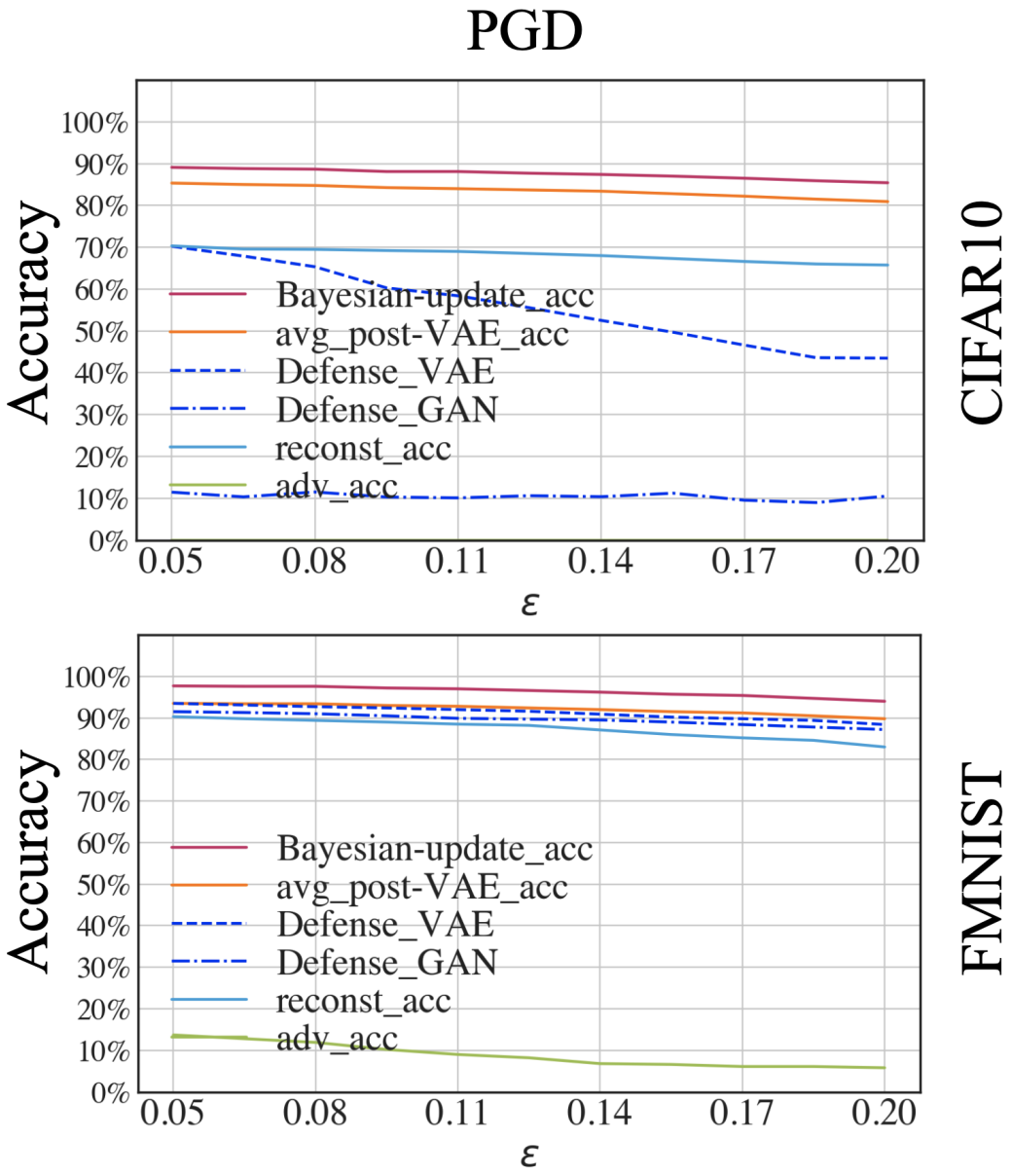


Figure 3.8: Comparison of different defense methods under PGD attack on CIFAR10 and Fashion-MNIST datasets.

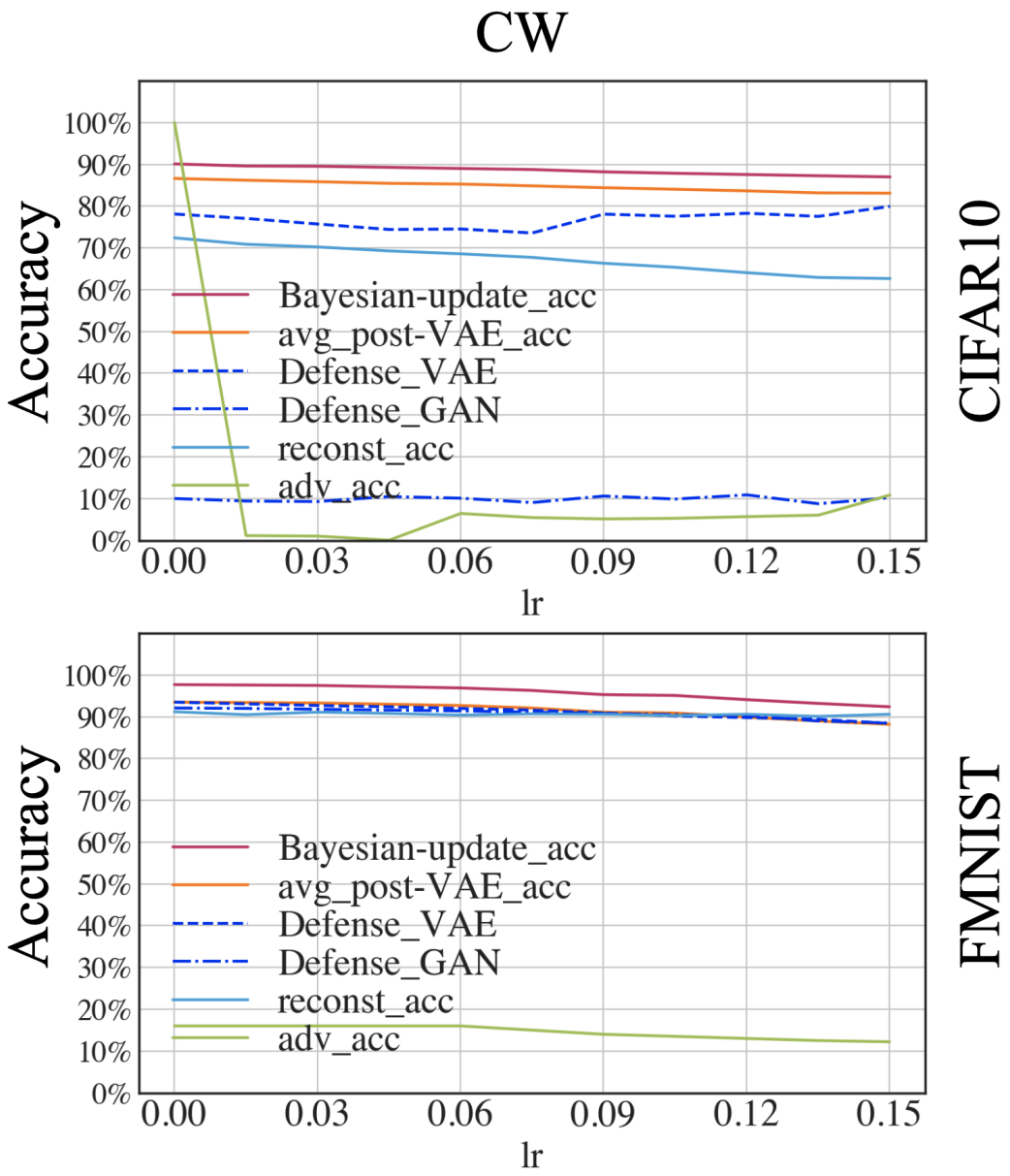


Figure 3.9: Comparison of different defense methods under CW attack on CIFAR10 and Fashion-MNIST datasets.

ods under a wide range of adversarial perturbations. The experiment compared the defense mechanisms with three other methods. For each test scenario, 5000 adversarial samples were generated using the testing dataset. The accuracy of the defense methods was then computed as the average accuracy achieved by three targeted classifiers A, B, and C.

Specifically, the comparison was conducted against white-box attacks, including the Fast Gradient Sign Method (FGSM), Carlini-Wagner (CW) attack, Basic Iterative Method (BIM), and Projected Gradient Descent (PGD) attack, at various perturbation levels. The results of these experiments are presented in Figures 3.6, 3.7, 3.8, and 3.9 when the models are under FGSM, BIM, PGD, and CW attacks respectively. This evaluation setup offers comprehensive insights into the robustness of the proposed defense mechanism and other existing state-of-the-art defense methods when the models are under diverse adversarial contexts.

The results depicted in Figures 3.6, 3.7, 3.8, and 3.9, demonstrate a superior performance of the proposed defense method compared to existing approaches across a broad spectrum of adversarial perturbation levels for both Fashion-MNIST and CIFAR10. Specifically, the proposed method achieves an average improvement of 25% in accuracy compared to other existing methods. Similarly, for Fashion-MNIST, the proposed defense method exhibits a superior performance, surpassing existing methods by an average margin of 5% in accuracy. Notably, the accuracy of the proposed method on Fashion-MNIST remains above 93% for various adversarial attacks and perturbation levels, indicating its resilience and robustness against adversarial noise for simpler datasets like Fashion-MINIST.

Moreover, the results indicate that the proposed defense method maintains a high level of accuracy on clean images in the absence of an adversarial attack, achieving approximately 91% accuracy for CIFAR10 and 99% accuracy for Fashion-MNIST. These findings underscore a minimal compromise on the original model's performance. The proposed method offers superior robustness for classifiers against adversarial attacks while maintaining high accuracy in the absence of adversarial attacks.

3.4.4 Defense When Adversary Has An Access to Entire Defense Mechanism

The experiments conducted in this section compared the proposed method with other existing methods in the context where the attacker targets the entire defense framework. In the case of the proposed method, this type of attack targets both the noise reduction unit (Variational Autoencoder, VAE) and the post-VAE classifiers. For other existing defense methods, some necessary adjustments were made in order for this attack to be employed in their setup.

For Defense-GAN, we connected the trained generator to a well-trained classifier, denoted as $f_c(f_{gen}(z))$. This configuration allows an adversary to access the knowledge for the entire framework to tamper the input z.

In the case of Defense-VAE, the adjustment connected a well-trained VAE to a classifier that had been trained using the reconstructed images from the same VAE. This setup enables the adversary to utilize the information $f_c(f_{VAE}(x))$ to launch a second wave of attacks.

In contrast, for the proposed method, each post-VAE classifier is connected to the well-trained VAE to form an end-to-end network. In this setup, the adversary utilizes this end-to-end network to craft adversarial samples. During the testing phase, adversarial samples generated by different end-to-end networks are randomly selected and fed into the proposed defense framework for evaluation.

This experimental setup allows for a thorough comparison of the proposed method with existing approaches under the scenario that the entire defense framework can be accessed by an adversary, providing valuable insights into its robustness for this type of attack. Results and performance comparison when the entire defense framework is being attacked are shown in Figures 3.10, 3.11, 3.12, and 3.13, for FGSM, BIM, PGD, and CW, respectively.

The results depicted in Figures 3.10, 3.11, 3.12, and 3.13 highlight the superior performance of the proposed defense mechanism compared to existing methods. Specifically, the

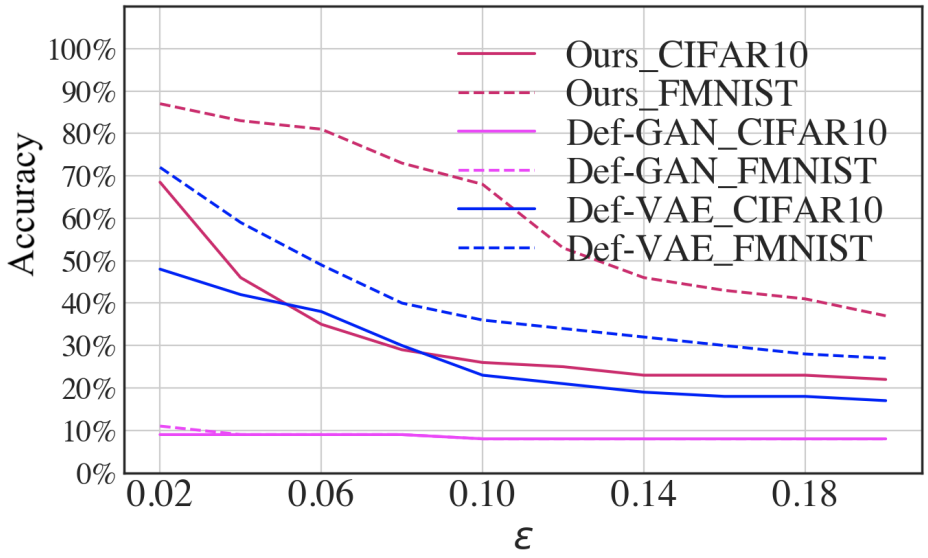


Figure 3.10: Performance comparison of different defense methods when the entire defense framework is under FGSM attack.

proposed method achieves an additional 10% improvement in accuracy on the CIFAR10 dataset and an additional 10% improvement in accuracy on the Fashion-MNIST dataset compared to existing methods under the FGSM and CW attacks. It is important to note that all three defense mechanisms experience a decrease in performance under this new attacking mode. This indicates that attackers with knowledge of the entire defense framework can significantly impede its integrity. These findings underscore the importance of developing defense frameworks that can also handle the case when both the deep learning systems and their defense framework are under attack.

Additionally, a notable observation on the comparison between the proposed method and Defense-GAN is that despite the adversary using similar information, $f_c(f_{VAE}(x))$, to generate attacks on both methods, the proposed method demonstrates superior performance compared to Defense-GAN. This improvement in accuracy can be attributed to the use of collective voting results, which will be elaborated upon in the subsequent subsection.

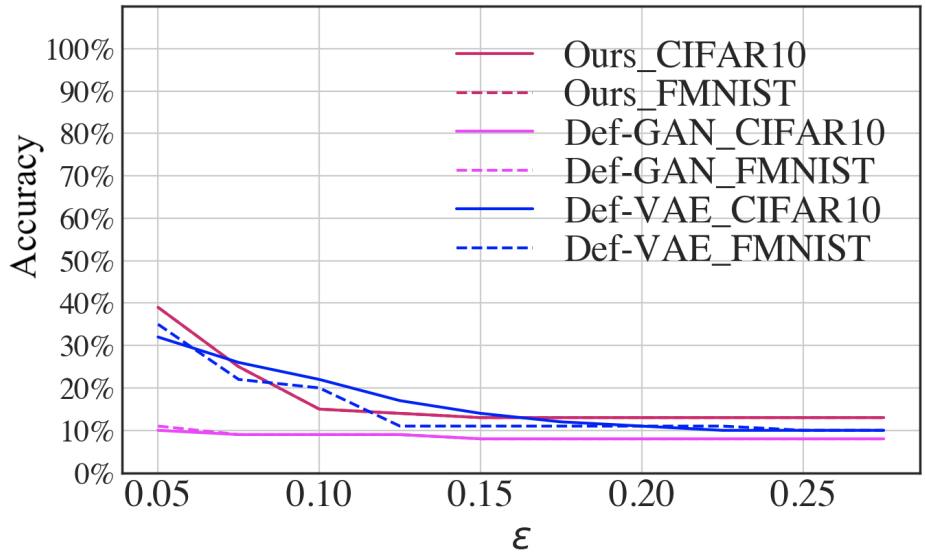


Figure 3.11: Performance comparison of different defense methods when the entire defense framework is under BIM attack.

3.4.5 Collective Voting

From the experimental analysis, the inclusion of a voting mechanism results in a notable increase in accuracy, ranging from 3% to 5%, superior to the performance achieved by averaging individual VNNs, as illustrated in Figures 3.6, 3.7, 3.8, and 3.9.

To take a closer look at how voting benefits the classification, the experiment measures how much percentage of the mispredictions are recovered through the voting process. Note that for simplicity in notation, we use VNNs to represent the voting classifiers. Among each VNN, the total wrong prediction is n_{wrong_pred}, which contains both recoverable predictions and non-recoverable predictions: $n_{wrong\_pred} = n_{can\_be\_recovered} + n_{not\_recovered}$. The recovery rate is: $\alpha_{recovery} = \frac{n_{can\_be\_recovered}}{n_{wrong\_pred}}$. Figure 3.14 illustrates the recovery rate across varying numbers of VNNs employed for voting. Each color corresponds to a distinct perturbation level of the FGSM noise on CIFAR10 (note that a similar pattern is observed

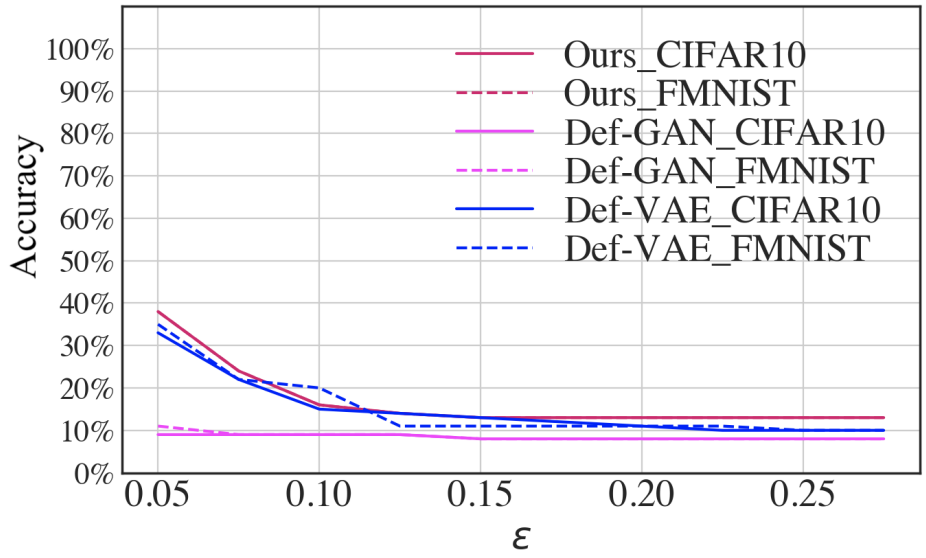


Figure 3.12: Performance comparison of different defense methods when the entire defense framework is under PGD attack.

for other adversarial attacks). The results indicate that as the number of VNNs increases, the incremental gain in recovery rate diminishes for each additional VNN utilized. This observation suggests that, while adding more VNNs to the voting process improves the recovery rate, the rate of improvement decreases with each additional VNN, indicating a diminishing gain in the number of VNNs used.

To elucidate the diminishing returns in extra gain for accuracy, Figure 3.15 illustrates the overlapping rate of wrong predictions among VNNs. This rate quantifies the percentage at which the wrong predictions of each VNN overlap with those given by other VNNs. Figure 3.15 reveals that as the number of VNNs increases, the overlapping rate of VNNs' wrong predictions decreases. This decrease indicates that there is greater diversity in the wrong predictions as more VNNs are employed. This phenomenon can be attributed to the fact that each VNN, with its unique structure and random parameter initialization, forms a slightly different classification boundary. However, as the rate of decrease in overlapping

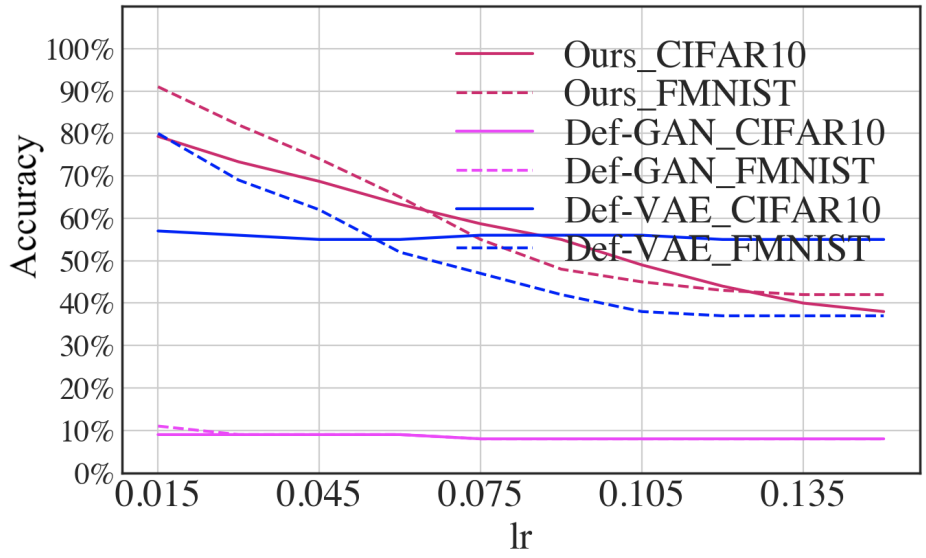


Figure 3.13: Performance comparison of different defense methods when the entire defense framework is under CW attack.

diminishes as the number of VNNs increases, it explains why the additional recovery gain achieved by employing multiple VNNs diminishes as more VNNs are used for voting.

The diminishing gain in additional recovery can also be understood by examining the overlapping rate of correct predictions among VNNs as plotted in Figure 3.16. As more VNNs are incorporated, there is a decreasing slope rate in the overlapping rate of correct predictions. This indicates that the agreement among VNNs on those correct predictions becomes more pronounced. This decreasing slope rate suggests that, with an increasing number of VNNs, the consensus among models regarding correct predictions strengthens.

3.5 Chapter Summary

A novel adversarial defense approach was introduced, incorporating randomization and discretization processes. Spatial Frequency Loss (SFL)-enhanced VAE was designed to

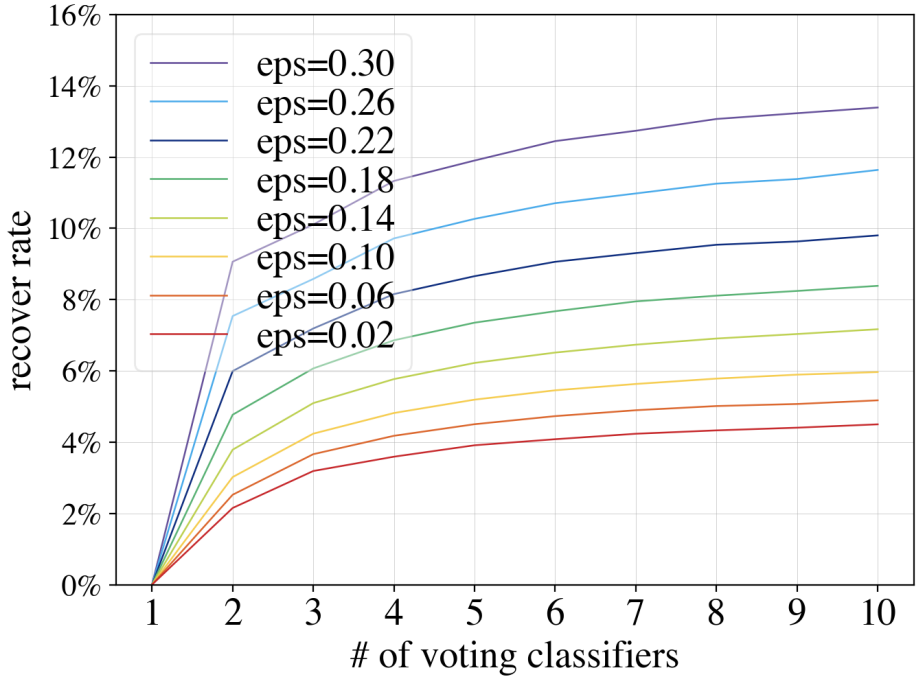


Figure 3.14: Recovery rate of mispredictions that are recovered through the voting process.

produce sharper reconstructed images. Both the prediction on the reconstructed images and a collective voting result of the reconstructed images were integrated into the final prediction using Bayesian Updating. Extensive experimentation was conducted to thoroughly evaluate the method's performance, comparing it against two prominent existing defense mechanisms across a range of challenging scenarios. The results demonstrated the method's superiority, showcasing an additional 25% increase in accuracy on the CIFAR10 dataset and a notable 5% increase on Fashion-MNIST.

Moreover, the proposed method exhibited a superior consistency in maintaining an accuracy exceeding 93% on Fashion-MNIST even under strong adversarial perturbation levels. Notably, when subjected to attacks targeting the entire defense framework, our method consistently outperformed other existing approaches by approximately 10% in

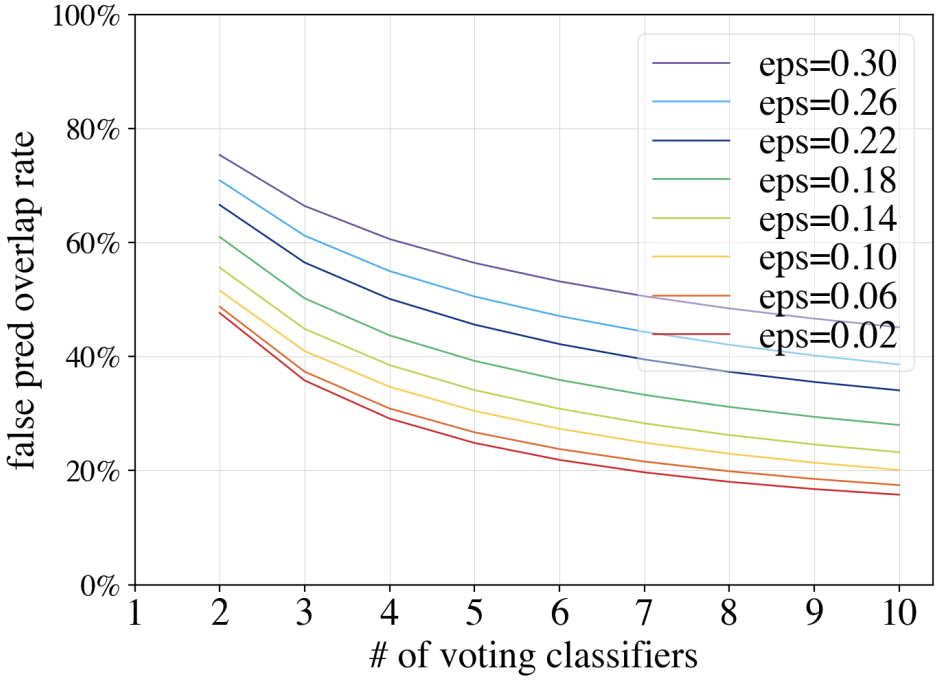


Figure 3.15: The overlap rate between different voting classifiers for the same wrong predictions.

accuracy for Fashion-MNIST and an average of 5% for CIFAR10. These findings underscored the robustness of the proposed defense framework under various adversarial attack contexts. The experimental results demonstrated the efficacy of the proposed techniques to enhance the security of deep learning systems under adversarial attacks.

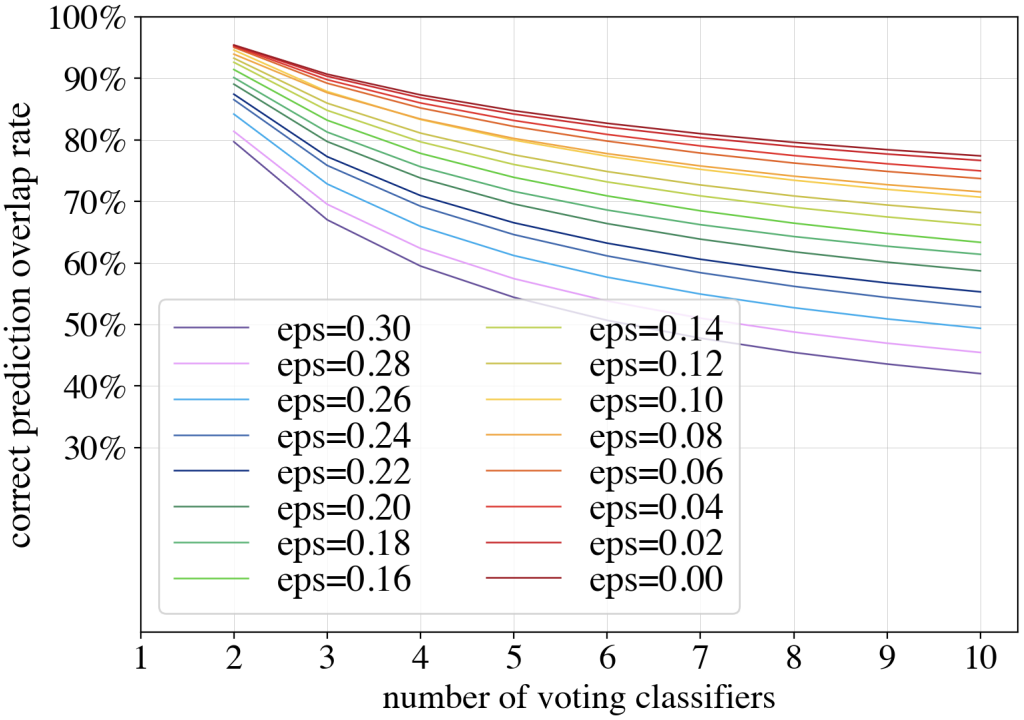


Figure 3.16: The overlap rate between different voting classifiers for the same correct predictions.

Chapter 4

Defense-CycleGAN: Multi-Adversarial Noise Filtering with CycleGAN

4.1 Introduction

Deep Neural Network (DNN)-based applications have emerged as a new computational paradigm that has transformed a wide range of fields, including video analysis [37], image recognition [53], natural language processing [78], and audio processing [48]. Despite their remarkable success, these methods are susceptible to adversarial attacks [105]. Adversarial attacks involve manipulating the input of deep learning applications, such as images, to deceive DNNs into making wrong predictions. For example, attackers can perturb a single pixel [102] or a group of pixels [26] in an image, causing the DNN to misclassify it. These perturbations are often imperceptible to the human eye, making it challenging to spot the difference between the original and adversarial images [92]. While the ability of DNNs to achieve excellent performance, their vulnerability to adversarial attacks has raised concerns about their reliability and security in real-world applications [26]. Addressing these challenges requires developing robust defense mechanisms that can protect DNNs from the impact of adversarial attacks without compromising the performance of DNNs

on legitimate inputs [105].

Various defense mechanisms have been developed to enhance the robustness of deep learning applications against adversarial attacks [35, 2, 96, 61, 112, 16, 89, 27, 67]. Among different categories of defense mechanisms, adversarial noise reduction has been an effective tactic to tackle adversarial problems. Generative networks, such as generative adversarial networks (GANs) or variational autoencoders (VAEs), are often the computational paradigm used for filtering adversarial noise in the input data or transforming the input data to a new domain that is similar to that of the clean data [72, 96, 35].

Variational autoencoders (VAEs) as the generative networks for reducing adversarial noise have shown significant promise in prior research [35, 23, 72]. However, a notable trade-off exists between noise reduction and the quality of reconstructed images from VAEs. In the previous works [35, 72], a decrease in the quality of those images reconstructed using VAEs has been observed, such as blurriness in images and lack of clear edges for objects. This blurriness is not limited to the noise reduction process when handling the adversarial samples but also exists in reconstructed images that are transformed from clean original data. Consequently, if the deep learning application wants to use those filtered images for classification, the accuracy will be negatively impacted by the blurry effect, even in the absence of adversarial attacks.

Generative Adversarial Networks (GANs) represent a powerful training paradigm for high-quality and high-resolution data generation [26, 49, 124]. The generated samples do not need to mirror the input data visually because the generator G aims to capture the underlying distribution of the original training data in order to generate realistic data from the learned distribution.

A fundamental distinction between VAEs and GANs lies in their architectures and training objectives. VAEs typically comprise an Encoder that compresses input data into a lower-dimensional intermediate vector and a Decoder that reconstructs data solely from this compressed vector. In contrast, GANs do not emphasize the compression and decompression process. Rather, the generative networks in GANs are designed to mimic

the data distribution rather than replicate the training data exactly, while VAEs aim to generate data identical to the input provided to the network. This difference stems from the training objectives: GANs strive to generate samples that confuse their discriminator networks, whereas VAEs focus on minimizing the Euclidean distance between input and output data and reducing the distribution difference between the prior and posterior in the designated latent feature space.

Several adversarial noise reduction approaches [23, 72, 61] attempt to enhance adversarial noise reduction by filtering adversarial samples multiple times using the same VAE. However, this iterative process can exacerbate the inherent blurry effect of the VAE. To tackle both the blurry effect inherent in the generative network and the potential deterioration of the noise during multiple passes of filtering, we propose a method called Defense-CycleGAN. This approach involves multiple passes of noise reduction with high-fidelity data reconstruction, which aims to mitigate the blurry effect while improving the efficacy of the adversarial noise reduction process.

4.2 Chapter Contribution

The proposed method integrates a multi-depth latent feature extraction approach for high-fidelity data reconstruction within a GAN-based training framework. Additionally, to address the issue of blurriness caused by multiple passes of the generative network, our method introduces a two-stage generative framework for noise filtering. This framework consists of two GAN-based generative networks: the first network reconstructs the data into a noise-reduced image domain, while the second network adds sharp details back to the image. Both generative networks are trained end-to-end within an adversarial training framework. To ensure the category of the object in the image remains consistent throughout the transformation process, we employ a cycle consistency loss to maintain object category consistency.

The innovation in the proposed method lies in several key aspects:

- Unlike previous approaches [23, 72, 61] that reuse the same generative network for multiple passes of noise filtering, our approach reframes multiple noise reduction passes as a domain adaptation problem. Our method involves training two distinct generative networks that learn both noise reduction and detail reproduction simultaneously by maintaining a cycle consistency.

- In contrast to prior GAN-based methodologies [96, 59], which typically focus on noise reduction from a single input space, and directly utilize the noise-reduced output for classification, our approach involves passing unknown inputs through multiple levels of feature extraction and reconstruction for both noise reduction and detail reproduction using two GANs. Additionally, the proposed method considers both the predictions from classifiers re-trained using the reconstructed images and the predictions from the targeted model in the final decision-making process, thereby enhancing the prediction accuracy.

The proposed method provides notable enhancements in data reconstruction quality, particularly in terms of fidelity to the original data. This improvement is reflected in increased accuracy when utilizing the reconstructed images for re-training targeted deep learning models. Furthermore, we leverage the performance of these re-trained models by incorporating their results to assist final predictions. This is achieved by combining the accuracy results with those of the targeted model's original predictions within a Bayesian Update framework. This integration leads to a further enhancement in prediction accuracy, adding extra accuracy gain in a range of 3% to 5% under various adversarial attacks.

Results from the experimental analyses demonstrate that the proposed defense mechanism surpasses state-of-the-art adversarial defense methods. Notably, even in scenarios where no adversarial noise is present in the input data, the proposed noise reduction framework produces nearly identical images as the clean data. Consequently, the targeted model can maintain nearly equivalent accuracy performance to that achieved by the original classifier. The defense performance of the proposed method exhibits robustness and resilience

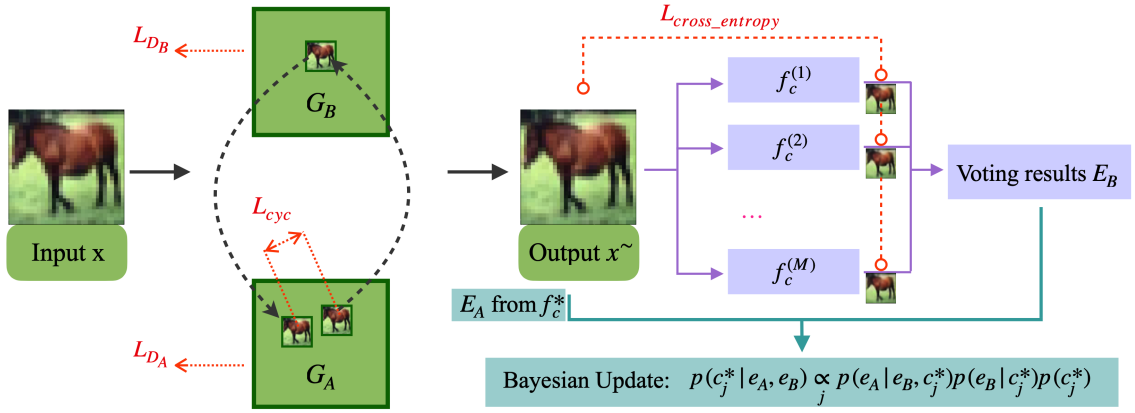


Figure 4.1: The network structure and data flow of the proposed defense mechanism.

against adversarial threats under diverse conditions.

4.3 The Proposed Method

The network structure and the data flow of the proposed method are shown in Figure 4.1. After training, the input images $X = \{x^{(i)}\}_{i=1}^{N}$ are first fed into the CycleGAN to reconstruct their noise-reduced counterparts: $f_{CycleGAN} : X \rightarrow X^{\sim}$. The reconstructed images $X^{\sim}$ are classified by the original classifier: $f^* : x \rightarrow \underset{j}{argmax}\ c_j$, where $c_j \in [0,1]$, $j = \{0,1,...,k\}$ is the index of the category, where k is the total number of categories, and c_j is the softmax layer output from the classifier for each class j. And $X^{\sim}$ are also fed into a group of M post-CycleGAN classifiers $F = \{f^{(i)}\}_{i=1}^{M}$ for their predictions, $O = \{f^{(i)}(x^{\sim})\}_{i=1}^{M}$. For the final decision, the output from the original classifier, $f^*(x)$, and the outputs from M post-CycleGAN classifiers, O, will be jointly considered through the Bayesian update process that is discussed later in section 4.3.3. The loss used for training all classifiers is categorical cross entropy loss $L_{cross\_entropy}$.

4.3.1 CycleGAN

Adversarial Loss [25] is used as the objective to guide the two identical generative networks, G_A and G_B, to generate samples that are indistinguishable from the target domain. If we denote the training samples as X, the generated samples as $X_g = G_A(X)$, and the two discriminator networks as D_A and D_B (for G_A and G_B, respectively), the training loss for D_A (similar to D_B) is the Least Squares [69] between X and X_g, so the loss for D_A, L_{D_A} (similar for D_B) is:

$$\begin{aligned}\mathcal{L}_{D_A} =& \mathbb{E}_{x\sim P(X)}\left[\sqrt{(1-D_A(x))^2}\right] + \\ & \mathbb{E}_{x\sim P(X)}\left[\sqrt{(D_A(G_A(x)))^2}\right]\end{aligned} \tag{4.3.1}$$

where $D : x \rightarrow [0, 1]$, and $G : x \rightarrow x_g$

The generative network G's objective is to generate samples X_g that are similar to the domain targets X. The loss for training the generative network G_A, L_{G_A}, (similar to G_B) is as below:

$$\mathcal{L}_{G_A} = \mathbb{E}_{x\sim P(X)}\left[\sqrt{(1-D_A(G_A(x)))^2}\right] \tag{4.3.2}$$

Cycle Consistency Loss. Cycle consistency [47] has been utilized as a unique regularizing strategy to promote constancy in the forward-backward loop, enabling deep neural networks to learn mappings between source and target domains. Various deep learning applications have attempted to address correspondence problems between different domains, such as transformation between 3D projections and 2D pixels [54], natural images and synthetic datasets [122], natural language and 3D models [54, 98], style translations [125, 15] and so on. In our method, we utilize cycle consistency to address the adversarial noise reduction problem, by formulating the noise filtering process as an image-to-image translation problem.

A generative network with sufficient capacity has the ability to map an input image to a variety of images similar to those from the target domain. However, this versatility

poses a challenge for classification tasks, as the generative network may transform an image into a different category. This issue arises because the adversarial training does not inherently ensure the category consistency between input and output, instead, the GAN training scheme aims to mimic a similar data distribution.

To address this challenge and to reliably train the generative network G to produce images within the same category as the input images, this work proposes to enforce a cycle consistency loss. Specifically, the proposed method introduces a cycle consistency loss between the input images, denoted as X, and the reconstructed images, represented as $G_A(G_B(X))$. Additionally, a cycle consistency loss is enforced between X and $G_B(G_A(X))$, ensuring that the generated images maintain their original category during the image-to-image transformation process. Additionally, not only does such a design improve the quality of the generated images by organically filtering the adversarial twice, but also enforces a closely resembled image reconstruction such that the final output images fall within the same category as the input images. Formally:

$$\begin{aligned}\mathcal{L}_{cyc} =& \mathbb{E}_{x\sim P(X)}\left[|x - G_A(G_B(x))|\right] + \\ & \mathbb{E}_{x\sim P(X)}\left[|x - G_B(G_A(x))|\right]\end{aligned} \tag{4.3.3}$$

The integration of cycle consistency loss in the training process facilitates the network to perform a predictable transformation from X to $G_B(G_A(X))$ and vice versa, from X to $G_A(G_B(X))$. By introducing the cycle consistency loss $\mathcal{L}_{cyc}$, the training encourages GB to generate $G_B(G_A(X))$ that closely resembles the input X. This is achieved when G_A focuses primarily on the distribution of the output while maintaining the content of the images relatively stable, as closely resembling X as possible. Consequently, G_B can also prioritize the output's distribution, enabling it to use the intermediate result $G_A(X)$ to recreate images that are nearly identical to X. This approach ensures that the generated images not only adhere to the desired distribution but also retain the essential content as appeared in the input images.

4.3.2 U-Net

The U-Net framework [94] is a powerful tool used in the image generation process. VAEs compress the input data's dimension to a much smaller dimension followed by the use of this compressed latent feature as the sole information for reconstructing the data to its original dimension. In contrast, U-Net takes a different approach, where it incorporates hidden features from each depth level in the network into the image generation process. This is achieved by iteratively concatenating early hidden features with those from deeper layers. This design allows U-Net to leverage primitive features from the early layers, which contributes to the generation of detailed, rich images that closely resemble the images from the target domain.

The use of U-Net is particularly advantageous in comparison to VAEs in terms of image reconstruction, because it helps overcome the limitations of VAEs for producing high-fidelity reconstructed data. VAEs often create a noticeable blurry effect on the generated images [123, 23, 35], which can limit the classification accuracy of those images. In contrast, U-Net's ability to incorporate features from multiple levels of abstraction allows it to generate images with greater detail and clarity, which can lead to improved classification performance. The visual comparison of network structure between VAE and U-Net is shown in Figure 4.6.

This capability of U-Net is further explored and discussed in detail in section 4.4.4, which provides a comprehensive analysis and experimental results about how U-Net's unique architecture contributes to the advantage in defending against adversarial attacks.

4.3.3 Collective voting and Bayesian update

Combining the results from multiple classifiers not only helps mitigate adversarial attacks but also enhances the overall accuracy of the final results [35]. Once a group of post-CycleGAN classifiers provides their predictions, denoted as O, a majority voting mechanism is taken. This voting result serves as one piece of evidence in the Bayesian

update and is labeled as E_B. Another crucial piece of evidence for the Bayesian update is the prediction from the original classifier on the reconstructed image, denoted as $f^*(x^{\sim})$, which is labeled as E_A.

The prediction from the original classifier on the unknown input image, $f^*(x)$, acts as the prior in the Bayesian process. The prior undergoes the Bayesian Update processing using $P(E_A = e_A)$ and $P(E_B = e_B)$, where e_A and e_B represent the category index. By leveraging these two pieces of evidence, the posterior (i.e., the final prediction result) for an unknown input is calculated using the Bayesian update [35] as follows:

$$\begin{aligned} & p(c_j|e_A, e_B) \\ & = \frac{p(e_A|e_B, c_j)p(e_B|c_j)p(c_j)}{\sum_j p(e_A|e_B, c_j)p(e_B|c_j)p(c_j)} \\ & \underset{j}{\propto} p(e_A|e_B, c_j)p(e_B|c_j)p(c_j) \end{aligned} \tag{4.3.4}$$

4.3.4 Data Augmentation

Data augmentation is a well-established technique for enriching the available training data, contributing to improving the generalization of neural networks, addressing overfitting problems, and enhancing the training dataset's diversity [100]. This technique has been deployed in various applications, including color space transformations [73], geometric transformations [100], and noise injection.

During the training for voting classifiers when using the generated noise-reduced data, we found that geometric transformations and noise injection effectively contribute to the network's resilience to adversarial perturbations. Therefore, during the training for post-CycleGAN classifiers, the data augmentation includes: vertical and horizontal random flips, Gaussian noise $\mathcal{N}(\mu = 1, \sigma = 0.2)$ injection, vertical and horizontal random translation by 20%, and vertical and horizontal random stretch for 10%.

4.4 Experiments

4.4.1 Experimental Setup

To evaluate the proposed defense mechanism, comprehensive experiments were conducted on the CIFAR10 [52] and Fashion-MNIST [115] datasets.

Fashion-MNIST dataset comprises 10 classes of hand-written digits, with 60,000 training samples and 10,000 testing samples. Each sample is a 28×28 grayscale image. On the other hand, CIFAR10 dataset consists of 10 categories of animals and objects, with 50,000 training samples and 10,000 testing samples. Each sample in CIFAR10 is an RGB image with dimensions $32 \times 32 \times 3$.

The networks were trained using the training samples, and all experiments were evaluated using the full 10,000 testing samples from each dataset.

The experiments compared the proposed method with three other existing defense methods: High Frequency Loss VAE (denoted as HFL-VAE, and note that its post-VAE classifiers are denoted as VAE_vnn) [35], Defense-VAE [61] and Defense-GAN [96]. All these defense mechanisms belong to a similar category, in that the defense mechanism includes a generative network to reduce the adversarial noise or to reconstruct images for classification. It tested all defense methods under four adversarial attacks: Projected Gradient Descent (PGD) [68], Carlini and Wagner Method (CW) [12], Fast Gradient Sign Method (FGSM) [26], Basic Iterative Method (BIM) [56]. For the implementation of generating adversarial samples of the four different attacks, the Cleverhans package [83] was used. Except that the perturbation level ϵ of the four adversarial attacks was varied in order to generate adversarial samples with various noise levels, the rest of the parameters of those attack algorithms were set to their default values.

For the post-CycleGAN classifiers, three different types of structures were adopted: (Residual-Networks [33], Wide-Residual-Networks [119] and DenseNet [40]). The implementation details of those classifiers were inspired by the public GitHub repository [60].

During the Bayesian update process, the Conditional Probability Tables (CPT) for $P(e_B|c_j)$ and $P(e_A|e_B, c_j)$ were computed using the entire training set. This step is crucial for accurately updating the posterior probability distribution based on the evidence provided by (1) the voting results from post-CycleGAN classifiers and (2) the original classifier's prediction on the reconstructed images.

There is one modification made for the Bayesian Update process. Given that the original classifier's predictions can be unreliable under adversarial attacks, our experiment assumed a uniform distribution for the prior $P(C = c_j)$ across all categories. This assumption simplifies the Bayesian update process while still allowing for meaningful usage of the voting result as evidence for correcting the distribution of the prior. The final prediction for each test image was determined using the posterior probability distribution calculated according to equation 4.3.4.

4.4.2 White-box Attacks on Classifiers

In order to obtain a comprehensive insight into the robustness of different defense methods under various adversarial attacks and noise perturbation levels, the analysis of defense performance under white-box attacks is conducted. In the context of white-box attacks, where adversaries possess knowledge about the classifier's architecture and parameters, the performance of various defense methods is evaluated against a diverse range of adversarial perturbation levels, as depicted in Figures 4.2, 4.4, 4.3, and 4.5 for Fast Gradient Sign Method (FGSM), Carlini and Wagner Method (CW), Projected Gradient Descent (PGD), and Basic Iterative Method (BIM) respectively.

It is worth noting that there are some major differences between the two datasets used in the experiments. The Fashion-MNIST dataset, comprising grayscale images of simple items like shoes, T-shirts, and pants, exhibits limited variation in size, orientation, and background. In contrast, CIFAR10's RGB-colored images depict a diverse array of objects such as cats, birds, and airplanes. CIFAR10 presents greater complexity than Fashion-

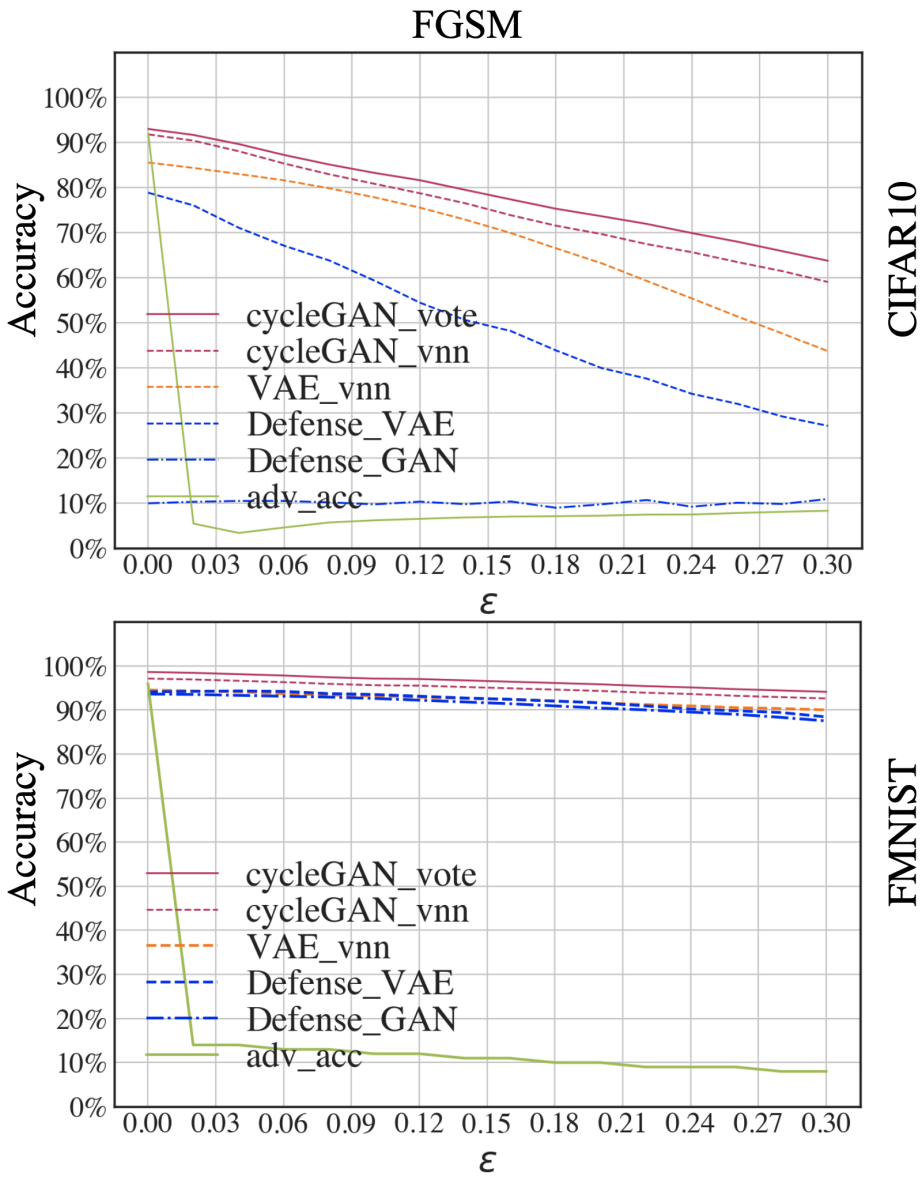


Figure 4.2: The comparison of different adversarial defense mechanisms against FGSM for CIFAR10 and Fashion-MNIST datasets.

MNIST due to the extra color channels that can produce substantially more variations in object outlines, color shades, and patterns within each category.

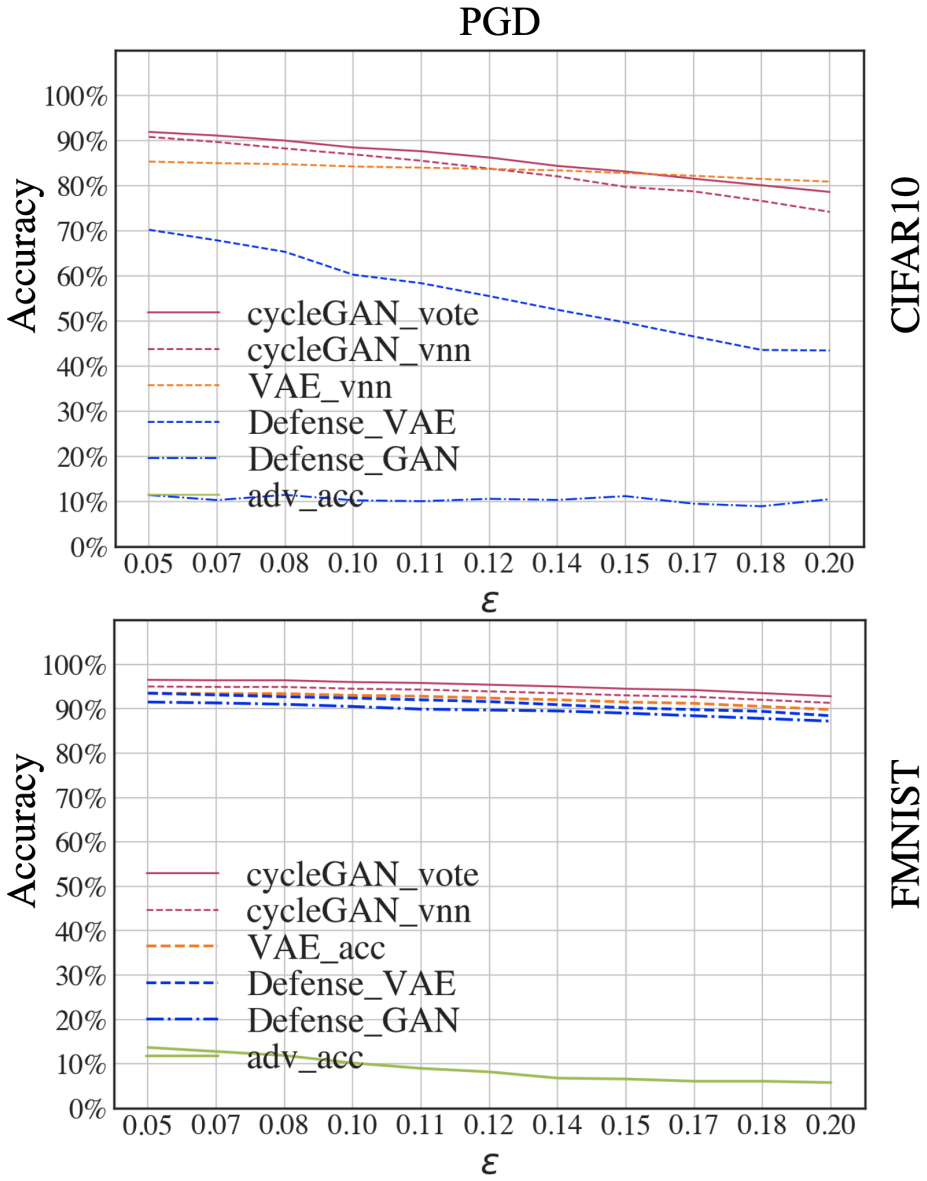


Figure 4.3: The comparison of different adversarial defense mechanisms against PGD for CIFAR10 and Fashion-MNIST datasets.

The difference in the complexity between the two datasets impacts the performance of defense mechanisms against adversarial attacks. The defense methods all exhibit superior

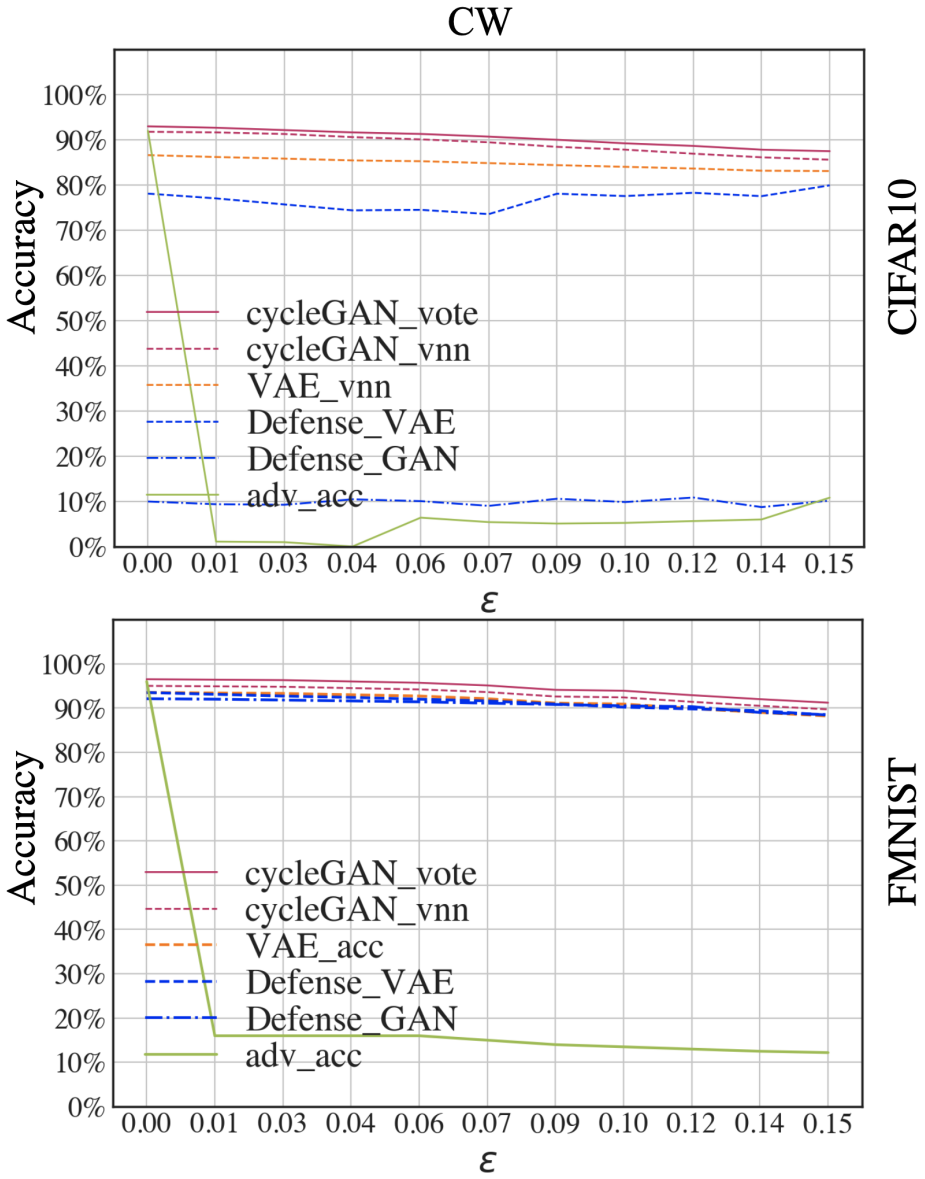


Figure 4.4: The comparison of different adversarial defense mechanisms against CW for CIFAR10 and Fashion-MNIST datasets.

performance on Fashion-MNIST, achieving accuracy exceeding 90% across a wide range of adversarial attacks, which underscores their effectiveness in handling simpler image

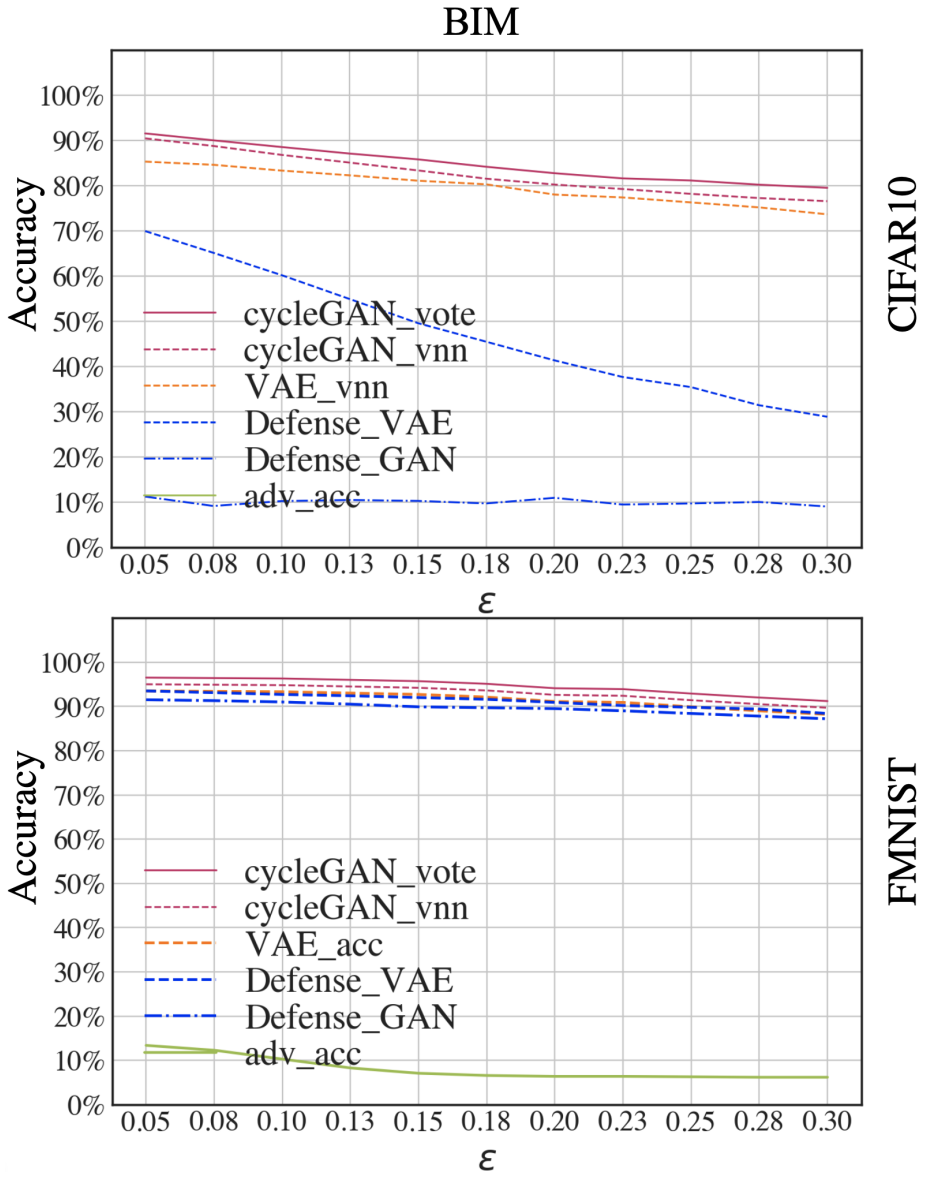


Figure 4.5: The comparison of different adversarial defense mechanisms against BIM for CIFAR10 and Fashion-MNIST datasets.

datasets.

The complexity of CIFAR10 presents a more significant challenge for defense methods

to subdue various adversarial attacks across the board. The performance of all four defense mechanisms shows a notable decrease in accuracy, especially when the level of adversarial noise increases. Among these methods, the CycleGAN approach demonstrates a superior performance compared to Defense-VAE and Defense-GAN, achieving an average accuracy margin of nearly 30% better than the existing methods. Furthermore, CycleGAN outperforms the HFL-VAE method by 5% to 13% across various levels of adversarial perturbations.

Moreover, by integrating the Bayesian Update with the collective voting results from multiple classifiers $f^{(i)}$ into the final classification, this add-on provides an additional accuracy boost of 3% to 5% compared to that achieved by a single classifier. This enhancement highlights the portability and efficacy of using extra add-on defense techniques on different defense methods, such as the majority voting followed by integrating the voting result into the final decision using Bayesian Update.

4.4.3 Model Performance Degradation after Noise Reduction on Clean Data

A robust noise-reduction-based defense mechanism should not compromise the original model's performance after the noise-reduction process. It should maintain a close performance of the targeted model especially in the absence of adversarial attacks. To show the analytic comparison, our experiments also compare different defense methods' performance with and without the presence of adversarial samples, as shown in Figure 4.2. It illustrates that the original classifier achieves approximately 97% accuracy on clean testing images for Fashion-MNIST and 91% for CIFAR10.

During the test for the Fashion-MNIST dataset, the proposed method demonstrates a nearly identical performance as the original classifier, with a notable 97% accuracy for the test. In contrast, Defense-GAN and HFL-VAE achieve around 92% accuracy for the Fashion-MNIST dataset.

Under the context of the CIFAR10 dataset, the proposed method also shows its efficacy in handling a more complex dataset, achieving 91% in accuracy in the absence of adversarial samples, which is nearly the same level of accuracy as that given by the original model. In comparison, the performance of the other three defense methods falls short for such cases, with the Defense-GAN and the Defense-VAE achieving under 80% in accuracy, and the HFL-VAE reaching approximately 86% in accuracy. This indicates that the proposed method does not incur a notable trade-off in the original model's performance on clean data while providing noise reduction defense against adversarial attacks.

4.4.4 Image Reconstruction

In evaluating the effectiveness and image quality driven by the data reconstruction capability of CycleGAN compared to VAE, a detailed analysis of the average L1 norm for the difference between original images and their reconstructed counterparts in the absence of adversarial attacks was conducted. This metric provides a quantitative measure of the fidelity of the reconstructed data from both generative models. By examining this aspect, we can better understand the inherent advantages of CycleGAN over VAE in accurately reconstructing images.

The key structural difference between VAE and CycleGAN lies in their network architecture. VAE is characterized by a fully sequential arrangement of network layers, whereas CycleGAN's generative network (U-Net) incorporates a unique concatenation approach for different layers. Specifically, U-Net concatenates the latent feature outputs from early layers to those from deeper layers, as illustrated in Figure 4.6. This design choice is rooted in the well-known understanding among the computer vision community that the early layers of deep neural networks (DNNs) tend to capture primitive features in objects, such as edges, corners, and colors [77]. Those primitive features allow for generating detail-rich replicas based on the inputs, contributing to a high-fidelity data reconstruction process.

To delve deeper into the influence of early latent features on the quality of reconstructed

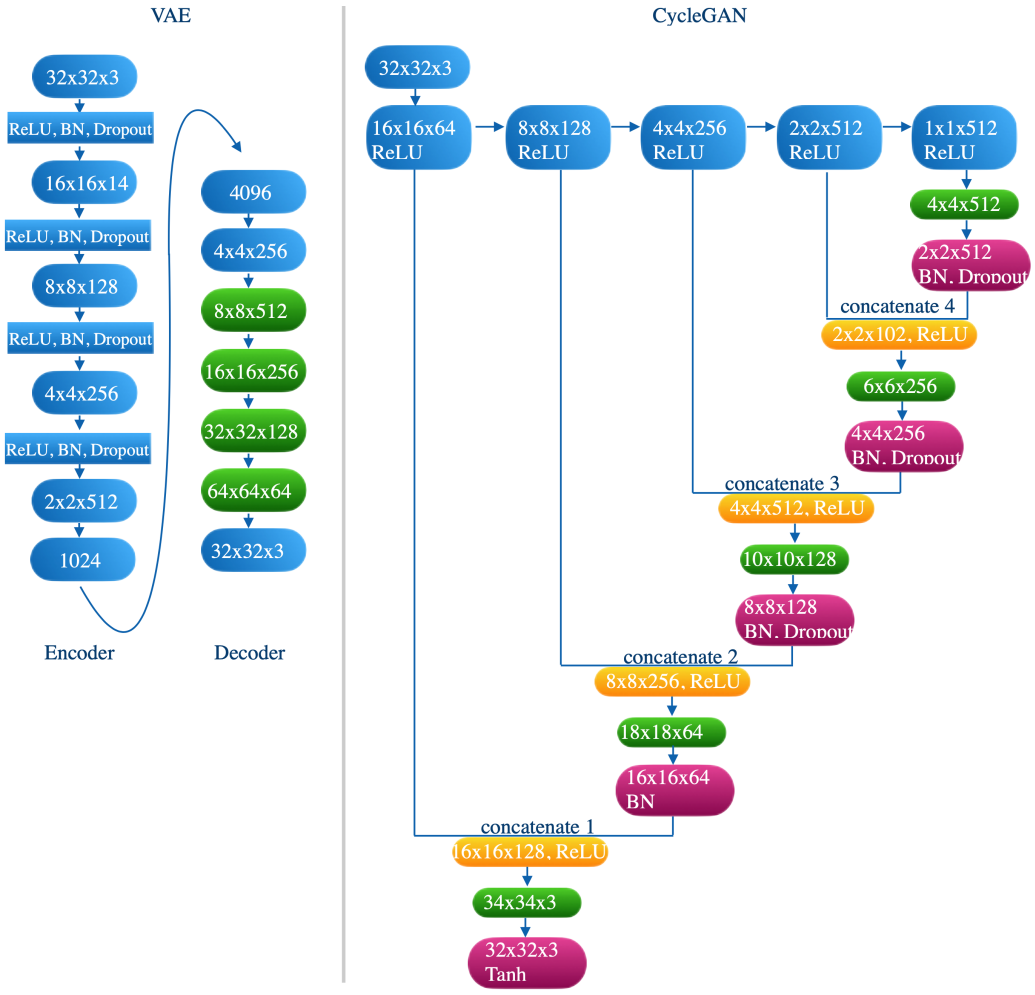


Figure 4.6: The generative networks' structures of VAE (left) and CycleGAN (right). For both networks, all the blue boxes with dimension output are Conv2D layers, and they all use kernel size 4, strides 2, and ReLU activation as their layer parameters; all the green boxes are Con2DTranspose layers, and kernel size 4 and strides 2 are used as their layer parameters. For the CycleGAN, the red boxes represent the Crop operation to trim the output dimension before they are fed into the yellow boxes which denote a Concatenation operation and their output dimension.

images, a detailed experimental analysis was conducted involving the gradual detachment of connections between early and deep latent features in the CycleGAN architecture.

The experimentation process starts by isolating *concatenate 1* in Figure 4.6 while keeping *concatenate 2, 3,* and *4* intact in the network structure. The newly created structure is denoted as Cyc_1. In order to maintain the output dimension of the network, adjustments were made to the sequential layers following *concatenate 2* by doubling the number of filters. This adjustment ensured that the output dimension matched the dimension of the output from *concatenate 1*.

In a similar methodology, a sequence of networks with various levels of detachment of the connections between early and deep latent layers was built for comparison. The experiment isolated *concatenate 1&2*, *concatenate 1&2&3*, and *concatenate 1&2&3&4*, and we denoted the newly created networks as Cyc_2, Cyc_3, and Cyc_4, respectively. Each of these networks represented a variation in the connection design between early and deep latent features, which in turn controlled the level of access to primitive features during the data reconstruction process. By doing so, it allows for a detailed examination of their impact on image reconstruction quality.

In our experiments, the L1 norm between the original images and reconstructed images in the absence of attacks was computed and averaged over 10,000 testing samples. The quantitative results of this measurement for various newly created networks are shown in Table 4.1. These experiments aimed to provide insights into the specific contributions of early latent features to the overall image reconstruction quality in CycleGAN.

Network	VAE	Cyc	Cyc_1	Cyc_2	Cyc_3	Cyc_4
L1 Norm	126.0	54.3	103.9	225.9	358.6	518.8

Table 4.1: L1 Norm of the difference between original images and reconstructed images from various generative networks.

Note that the L1 norm serves as a crucial metric indicating the divergence between

the reconstructed images and their original counterparts. As depicted in Table 4.1, the L1 norm for images generated by the original CycleGAN stands at 54.3, which is significantly lower than the 126.0 observed for VAE-generated images. This discrepancy sheds light on why the targeted classifier achieves higher accuracy on those data that are generated by CycleGAN compared to that from VAE.

In transitioning from Cyc_1 to Cyc_4, which is represented as the access to various levels of primitive features in the network, there is a notable increase in the L1 norm, escalating from 103.9 to 518.8. Since Cyc_2, the L1 norm already surpasses that of VAE-generated images, indicating a stronger divergence between reconstructed data and the inputs.

Notably, Cyc_4 adopts a fully sequential network structure after disengaging all concatenation layers, resembling VAE's sequential layer configuration. Both Cyc_4 and VAE share a common characteristic, a bottleneck layer in the network where the output dimension is substantially smaller than that of the input and output. However, there is a notable difference between Cyc_4 and VAE. While VAE imposes regularization on the bottleneck layer's output, for example, it imposes a multivariate Gaussian distribution for the bottleneck layer, Cyc_4 does not. This lack of regularization in Cyc_4's bottleneck layer adversely affects the accuracy of image reconstruction.

The escalating L1 norm from Cyc_1 to Cyc_4 underscores the benefit of early layer latent features in image reconstruction. The diminishing access to these primitive features in Cyc_4 results in a lack of availability of rich details necessary for generating high-fidelity images similar to the input images.

4.5 Chapter Summary

A new adversarial defense mechanism, Defense-CycleGAN, was introduced to mitigate adversarial noise in unknown input samples and to generate clean samples suitable for classification. Unlike traditional methods, Defense-CycleGAN leveraged various levels of

latent features, both primitive and highly abstract features in a network, enabling more accurate image reconstruction.

To evaluate the efficacy of Defense-CycleGAN, extensive experiments were conducted, comparing it against three existing adversarial defense methods. These defense methods were tested under four different adversarial attacks with a wide range of perturbation levels, on the CIFAR10 and Fashion-MNIST datasets. The results demonstrated that images reconstructed using CycleGAN were notably more accurate than those reconstructed using the VAE-based approach.

Across various attacks, Defense-CycleGAN consistently achieved high accuracy rates ranging from 90% to 98% on Fashion-MNIST. During the test on the CIFAR10 dataset, Defense-CycleGAN surpassed other existing methods, consistently maintaining above 80% in classification accuracy under CW, PGD, and BIM attacks. In the absence of an adversarial attack, the three other methods showed a 6% to 10% drop in classification accuracy, whereas the proposed method maintained a nearly identical accuracy to the original classifier's performance.

Overall, the proposed Defense-CycleGAN method outperformed the three comparative methods by a substantial margin of 5% to 30% across different attacks under a wide range of noise perturbation levels. This highlighted the effectiveness of the proposed method in defending against adversarial attacks, underscoring its reliability without compromising the original model's performance.

Chapter 5

VQUNet: Vector Quantization U-Net for Defending Adversarial Attacks

5.1 Introduction

Modern DNNs have made their success in achieving excellent performance on a wide range of tasks in various domains, e.g., images [34, 43, 53], audio [48, 103] and videos [32, 18], etc. However, it's been widely observed that DNN-based approaches are sensitive to samples perturbed by adversarial attacks [3, 13]. Such adversarial samples can be indistinguishable from real data but they can cause DNN models to give wrong predictions. Because each benign sample can be used to craft various adversarial counterparts, it's challenging for a defense mechanism to handle all possible adversarial samples, and at the same time, without introducing any decrease in a model's performance on benign data when there is no adversarial attack.

The research community has developed a range of techniques and methodologies to address the challenges posed by adversarial attacks. Some of the examples include:

- adversarial training, which involves augmenting the training dataset with adversarially perturbed samples, such that the deep learning models are exposed to the polluted

samples early [11, 26].

- Provable defense, is a branch of defense approaches that aims to establish a certified guarantee that, for any given adversarial sample, as long as certain conditions are met, the model will remain resistant to any type of adversarial sample. For example, in the work from [89], if an unknown input does not deviate from its benign counterpart more than a L_p-norm-ball, the model will be guaranteed to have a correct prediction regardless of the attack algorithm employed to craft it.

- Gradient obfuscation, a category in defense that seeks to obscure adversaries from calculating the adversarial noise such that the adversarial attacks are mitigated [86, 9].

- Detection, another branch of adversarial defense that focuses on identifying anomalies in the model inputs [117, 20].

- Adversarial noise reduction, which aims to clean unwanted noise from unknown inputs [50, 51].

In the realm of adversarial noise reduction, the complexity of the dataset affects the robustness of the noise reduction and the image quality produced by the generative networks. For instance, reconstructing RGB images from CIFAR10 is more challenging than reconstructing grayscale images from Fashion-MNIST. As a result, the performance of noise-reduction-based methods can vary significantly across different datasets [36, 35, 46, 45].

Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs) are the two widely studied generative paradigms for training noise reduction neural networks. While both generative networks aim to improve the generation process for realistic data, they differ in their training objectives and methodologies. VAEs emphasize the encoding and decoding process, aiming to compress the data while retaining the resemblance between input-output pairs. The compression stage produces a latent representation for the input, which is an important part of the VAEs' learning objectives for reconstruction and noise reduction. On the other hand, GANs focus on training a discriminator to distinguish real

and generated data. The discriminator in turn is used as guidance to help the generator to learn the underlying data distribution and reproduce more realistic data.

It has been observed in the literature that during noise reduction, VAEs tend to introduce blurry effects into the reconstructed image data [61, 72, 23, 123]. This blurriness stems from information loss from data compression and a lack of emphasis on reconstruction for sharp details incurred when using the mean square error loss for training [35]. In contrast, GANs depart from the data compression task, allowing for greater flexibility in the design of their generator networks. This flexibility offers GANs the capability to achieve a more realistic data reconstruction quality compared to VAEs.

Despite GAN-based generators benefit from the flexibility in their data generation pipeline, challenges persist. One such challenge is that adversarial noise can persist even after the noise reduction process. This occurs when the noise perturbation is so pronounced that a significant portion of the information becomes distorted and it can no longer be recovered by the generators. Consequently, while the primary objective of the generators is to reduce adversarial noise, excessively strong noise can introduce unwanted features into the image object. These features may then persist even after the noise reduction process.

5.2 Chapter Contribution

In this chapter, we present a novel noise regularization strategy to mitigate and limit the impact of the strong perturbation introduced by adversarial samples. The method combines multi-scale hierarchical representation learning with discrete latent feature learning using Vector Quantization (VQ). The goal is to reconstruct high-fidelity images and enhance the model's tolerance to adversarial noise.

The key concept of the proposed noise regularization lies in the vector quantization mechanism during the data forward pass, where it maps the continuous vectors of latent features in the network to a set of learnable discrete vectors. Even with largely increasing adversarial noise, many of these discrete vectors will still remain the same for those affected

continuous vectors. This approach effectively mitigates the impact of adversarial noise, particularly when the perturbation level is large, compared to existing noise reduction defense methods.

The process of mapping from continuous vectors to discrete vectors can result in information loss, potentially lowering the quality of data reconstruction. To address this issue, the generative network is designed to employ hierarchical latent features for data generation. In this hierarchical approach, latent features at shallower depths provide more primitive image features, while deeper levels retain more global information. At each depth, there is an independent vector quantization process dedicated to the discretization transformation specific to that depth level.

In particular, the proposed work contributes to the field in the following ways:

- It introduces Vector Quantization U-Net (VQUNet), a novel generative model that is designed to improve data fidelity during noise reduction. It introduces minimal performance degradation ($< 1\%$) compared to the original model's performance after the data noise reduction process. VQUNet achieves this by leveraging multi-scale hierarchical latent feature learning for data reconstruction.

- VQUNet significantly enhances the robustness of the defense framework against a range of adversarial attacks, particularly when confronted with high levels of adversarial perturbation. Its performance surpasses that of current state-of-the-art methods by a considerable margin. This improvement is primarily attributed to the vector quantization process employed by VQUNet, which involves the mapping of latent features from a continuous space to a discrete space. This mapping effectively mitigates unwanted value outbursts in the latent features caused by substantial adversarial noise.

- To the best of our knowledge, this work is a pioneering effort to provide unique insights into how adversarial noise impacts internal value shifts dynamically as the network is being attacked by adversarial samples. Our approach offers a unique

perspective to observe how noise reduction is mitigated by our design of hierarchical learning and quantization mechanisms. The novel design and the compelling quantitative results contribute to the advancement of adversarial defense methodologies.

- Via rigorous experiments, we show that VQUNet outperforms state-of-the-art adversarial noise reduction defense methods across a wide spectrum of adversarial noise levels and various adversarial attacks on two benchmark datasets (CIFAR10 [52] and Fashion-MNIST [115]). Importantly, the de-noising process implemented by VQUNet does not compromise the prediction accuracy of clean data after the de-noising network processes them. The proposed method incurs a minimal impact on the original deep learning model's performance when there is no adversarial attack.

- Our comprehensive experimental investigation delves into the individual components of our approach, shedding light on their respective contributions and efficacy to the overall performance in defending against adversarial attacks.

5.3 The Proposed Method

In this study, we introduce a new adversarial noise reduction network, Vector Quantization U-Net (VQUNet), which features an effective noise reduction procedure to regularize both adversarial and clean data. VQUNet enhances deep learning models' robustness against various adversarial attacks. VQUNet features a U-shaped latent feature forward pass to combine both the high-level latent features and low-level information, such as texture, localized features, and geometry of objects. Both high-level and low-level continuous latent representations will be regularized to a set of discrete representations through vector quantization, mitigating the disturbance from adversarial noise. The combination of the information from different hierarchies of the network allows the decoding blocks to utilize the local discrete latent features to yield a more precise recreation of the input.

VQUNet Structure

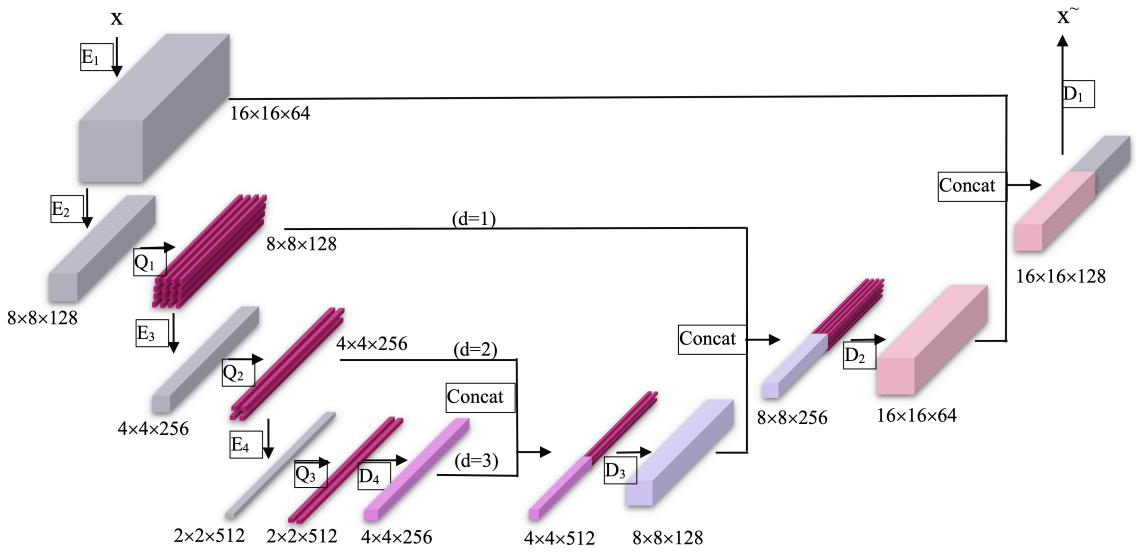
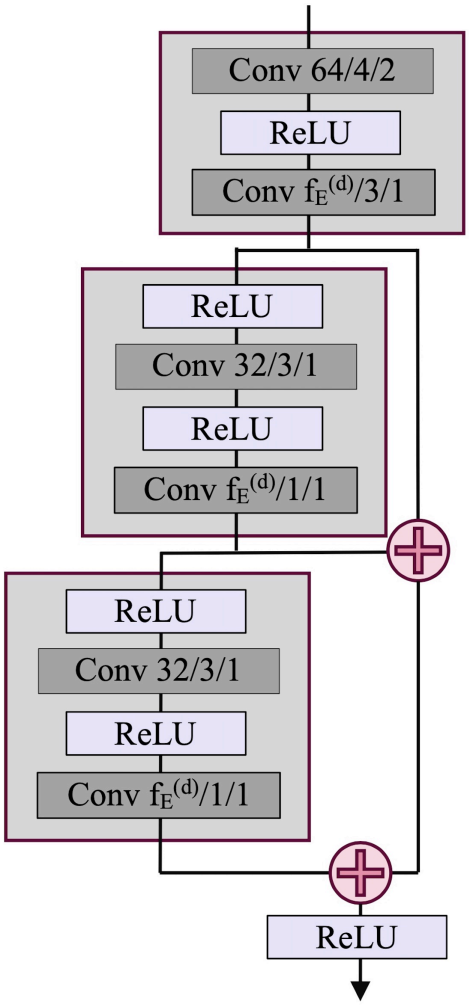


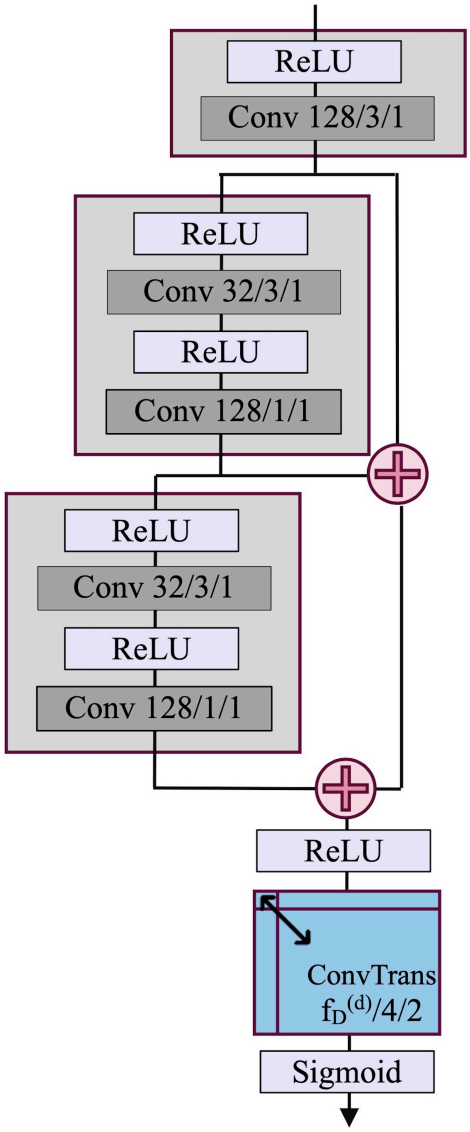
Figure 5.1: The structure and data flow of VQUNet, where E denotes the encoding block (in Figure 5.2a), Q denotes the quantization block and D denotes the decoding block (in Figure 5.2b). The depth level is indicated by d, which controls the parameters in E and D blocks.

The structure of VQUNet is illustrated in Figure 5.1, where the input is an image, and the output is its noise-reduced counterpart. The non-linear forward pass shares a touch of resemblance to the traditional U-net [94], where the feature extraction and data reconstruction path are symmetric at different depth levels as opposed to traditional linear structure. However, the design of the proposed network, training process, and application differ largely from the traditional U-net [94].

For simplicity, we annotate the encoding, decoding, and vector quantization process with E_d, D_d, and Q_d respectively, where d represents their depth level. The single arrows in Figure 5.1 indicate the data flow in the forward pass. The internal network structures in detail for the encoding blocks E_d and decoding blocks D_d are illustrated in Figure 5.2a and Figure 5.2b, respectively.



(a) The encoding block structure, denoted as “E" in Figure 5.1.



(b) The decoding block structure, denoted as “D" in Figure 5.1.

Figure 5.2: The detailed network structure for encoding block and decoding block.

The vertical direction (from top to bottom) in Figure 5.1 is the encoding process in various depth levels, which extracts more globally shared latent features as the forward

pass moves deeper vertically. A common practice for latent feature extraction is to apply a sequence of Convolutional Layers to the feature outputs. However, when the forward pass is too deep, the network struggles to use the highly abstract latent features for data reconstruction. Therefore, we depart from this design and apply residual blocks to the encoding process as shown in Figure 5.2a, which helps retain the information from earlier layers, avoiding the information degradation problems.

In VQUNet, every encoded feature vector from the encoder block is matched to one of the quantized vectors in the codebook. This process is denoted as Q_d in Figure 5.1, where d represents the depth level for the vector quantization process. For a comprehensive understanding of this mapping mechanism, we will provide a detailed explanation and full discussion in Section 5.3.1, where we will delve into its intricacies and implications.

Upon completion of all encoding processes, at the deepest level, the latent features have been transformed to the dimension of $2 \times 2 \times 512$. From there, the decoding process starts to expand this highly abstract information. This process unfolds in two stages: (1) the features initially pass through a convolutional block and two residual blocks; (2) following this, a ConvTranspose process is employed to expand the size with a stride of two, as illustrated in Figure 5.2b.

After decoding, the decoded features are concatenated to the quantized features from the immediate previous depth level. This process generates hybrid features, which subsequently undergo further expansion by the next decoding block, as depicted in the data pipeline illustrated in Figure 5.1. As the decoding and expansion process approaches completion, the output size matches that of the original data, and adjustments are made to the output's channel number to ensure consistency with the domain of the original data.

5.3.1 Vector Quantization

In the forward pass, vector quantization employs a non-linear mapping process that converts the continuous outputs of each encoding block into discrete codes at various

hierarchical levels. These discrete codes are selected from a learnable codebook that contains a group of discrete vectors. In this process, a discrete vector is chosen and assigned to replace the continuous vector based on its similarity to the continuous vector.

Particularly, the vector quantization model [108] includes a learned codebook that contains up to K 1-D vectors. Following each encoding process, each of the continuous feature vectors will be mapped to one of the quantized vectors in the codebook. The non-linear mapping is decided by: (1) calculating the Euclidean distance between the natural feature vector and the quantized vectors in the codebook; (2) the quantized vector with the closest distance will be selected as the replacement for the natural feature vector. Formally [90], with the input data, x, to quantize the natural feature vectors $E(x)_i$, the quantized feature $Q(E(x)_i)$ is calculated as below:

$$Q(E(x)_m) = e_k \quad \text{where} \quad k = \arg\min_j \left\| E(x)_m - e_j \right\|_2 \tag{5.3.1}$$

where $k \in 1, 2, \cdots, K$ (K is the total number of quantized vectors in the codebook), and $m \in 1, 2, \cdots, M$ where M is the total number of natural feature vectors in $E(x)$. The quantized feature vectors $Q(E(x))$ will be used for data reconstruction during the forward pass.

Note that each depth level in the network has its own quantization process and unique codebook. This design enables the codebook to learn the discrete latent feature representations specific to a depth level. The encoding process at each depth level extracts the latent features that relate to a particular granularity level of information. As a result, the depth-specific codebook can learn a more coherent group of quantized vectors within itself.

The input quantization process has a regularization effect on both small and large perturbations of adversarial noise. It's worth noting that a continuous vector is mapped to the most similar discrete vector based on their Euclidean distance. In cases where the perturbation level is small, although it may cause slight value shifting in the continuous vector, as long as the Euclidean distance between the perturbed continuous vector and the current discrete vector still remains the smallest compared to other discrete vectors in the

codebook, the same discrete vector is still used for reconstructing the same data point, which mitigates the impact of the adversarial noises.

In scenarios where the adversarial perturbation level is substantial, vector quantization (VQ) also plays a crucial role in providing regularization within the network. When the noise perturbation of the input is large, as the noise keeps increasing, the values in the perturbed continuous vector also shift in certain directions. However, because the continuous vectors are always mapped to their discrete counterparts, the discrete vectors themselves do not undergo value shifting. Instead, they are replaced by a different set of discrete vectors selected from the same codebook.

The encoding process is specifically trained to reconstruct data using the discrete vectors stored in the codebook. Even though the adversarial noise might cause value changes in the continuous vectors, no matter how drastic those changes are, the VQ process always replaces the perturbed continuous vector with one of the discrete vectors. This mechanism effectively mitigates the disruptive effects of large adversarial noise. For example, if the increasing adversarial noise pushes a continuous latent feature vector further away from all the discrete vectors in the codebook, regardless of how far the vector has been displaced, only the one discrete vector with the closest Euclidean distance will ultimately be used for data reconstruction. This behavior exhibits a regularization effect on largely increasing adversarial noise. While the values in the continuous latent vector continue to shift due to adversarial noises, the discrete vectors maintain their value range once the training is complete.

Although the VQ provides a distinctive approach to managing the impact of adversarial noise, there are challenges associated with learning a high-quality set of vectors in the codebook. There are two prerequisites that need to be met for VQ to achieve better results:

- Representative Codebook: It is ideal for most of the discrete vectors to be utilized by the entire dataset, instead of only a small group of discrete vectors being selected from the codebook, which can cause a lack of diversity in the discrete features representation.

- Effective Training: The training procedure for VQ should be carefully designed to ensure that the discrete vectors in the codebook are well spread in the high-dimensional space instead of clogging together. This goal ensures that the set of discrete vectors can represent more locations in the space, bringing better diversity in discrete feature representation.

A comprehensive explanation of the training process for the codebook is provided in Section 5.3.2. This section delves into the specifics of how the codebook is trained to ensure that it contains a representative set of vectors that effectively capture the essential features of the data.

5.3.2 Training

For the N images $X = \left\{x^{(i)}\right\}_{i=1}^{N}$, the VQUNet regularizes the input to its noise-reduced counterpart $X^{\sim}$: $f_{VQUNet} : X \rightarrow X^{\sim}$. The adversarial version of the data is denoted as $X^{adv} = \left\{x^{adv(i)}\right\}_{i=1}^{N}$.

Reconstruction Loss

For the data reconstruction, the loss to guide the network to recreate data that are similar to the clean samples is the mean square error between the inputs and network outputs:

$$L_{reconst} = \mathbb{E}_{x \in X}\left[\left|x - f_{VQUNet}(x))\right|^2\right] \tag{5.3.2}$$

Training the Encoder Blocks with Reconstruction Loss

The encoding process is denoted as $f_E : x \rightarrow a$. a_d is used to denote the output from an encoding block at depth level d. The vector quantization process is denoted as $f_Q : a \rightarrow q$, and the quantized vector (discrete vector) at depth level d is denoted as q_d. Note that the implementation of the codebook is through the Embedding Layer in Tensorflow, in which a 2-D matrix is created as the codebook parameters. The size of a 2-D matrix is $K \times G$, where K is the total number of the 1-D quantized vector in the codebook, and each quantized vector has G entries. The first step of the quantization process is to compare an input vector

against all the K discrete vectors, such that the index of the most similar discrete vector can be found, as shown in Equation 5.3.1. Then this index is used in Embedding Layer's look-up process to output the corresponding discrete vector.

However, this look-up operation that compares the input vector against K discrete vectors and then finds the index of the most similar discrete vector, is a discrete process. Because of the discrete nature of the comparison operation, in the Tensorflow computation graph of the neural network, there is not a differentiable path during gradient back-propagation from the Embedding Layer to its previous layer. In such case, the gradient of $L_{reconst}$ w.r.t. the parameters of those layers before the Embedding Layer (e.g., the parameters in the Encoding blocks) will not be available. During implementation, Tensorflow automatically assigns "None" or "Zero" to the gradient of loss w.r.t. those variables because they are regarded as unconnected variables, caused by the comparison operation which is not differentiable.

To bypass the non-differentiable path issue above, an intermediate variable is used to bridge the encoder blocks' parameters to the Embedding Layer, such that the encoders' parameters are connected to the rest of the computation graph during the gradient back-propagation. First, note that a_d and q_d are already available during each forward pass, then, during the implementation a new variable o_d is created in the computation graph and assigned with the value: $o_d = q_d - a_d$. The value of o_d is treated as constant. During each forward pass, a new variable is created and defined as $q_d^* = a_d + o_d$. This new variable q_d^* will be used as the input for the layer immediately after the quantization block at depth d.

It is worth noting that q_d^* holds the same value as q_d. However, by treating o_d as constant and using q_d^* in the forward pass instead of q_d, the path in the Tensorflow computational graph from a_d to q_d^* becomes a linear operation. This linear relationship makes the back-propagation path from q_d^* to a_d to be differentiable in the neural network. Thus, the parameters in encoder blocks can be optimized with the gradient descent of $L_{reconst}$ during training.

Training the Codebooks with Reconstruction Loss

The optimization of quantized vectors in the codebook with respect to $L_{reconst}$ is achieved through standard back-propagation. This is possible because of the fact that the quantized vectors, q_d, already form a seamless connection to the layers that come immediately after them. This direct connection allows for gradient flow during back-propagation, enabling the codebook vectors to be optimized effectively based on the reconstruction loss.

Encoder&Quantization Loss

Because there is no constraint on the value range for a_d and q_d, as training goes, a_d and q_d can end up having large differences in their elements' value magnitude.

This value range divergence can make it difficult for the training to converge. For example, if the value of the elements in a_d is much bigger than that of q_d, then during the discrete mapping, a small change in a_d can cause a very different set of quantized codes to be selected between each training step. Therefore, two extra loss functions are imposed to bring the value range of a_d and q_d closer.

First, to guide the parameters of encoder blocks to be closer to the codebook's value range, the intermediate quantized vectors q_d^* are reused. The mean square error between q_d^* and a_d is used as the loss function in training to guide the encoder blocks to give output closer to the quantized vectors in the codebook. The encoder loss is defined as:

$$L_E = \mathbb{E}_{x \in X} \left[\left| q_d^* - a_d \right|^2 \right] \tag{5.3.3}$$

Second, in order for the quantized vectors in the Embedding Layer to form a closer value range as the encoder blocks' output, another intermediate variable, a_d^*, is created in the computation graph for training. The variable a_d^* is created to hold the value of the encoder blocks' output, a_d. Then the mean square error between a_d^* and q_d is used as the quantization loss to guide the codebook vectors' training, denoted as L_Q:

$$L_Q = \mathbb{E}_{x \in X} \left[\left| q_d - a_d^* \right|^2 \right] \tag{5.3.4}$$

Optimization for Discrete Vectors in Codebooks

Although the encoder&quantization loss above can help narrow the value range between pairs of continuous and discrete vectors, it does not guarantee that the entire dataset will

be represented by a diverse set of discrete vectors from the codebook. In some cases, only a small subset of discrete vectors may end up being used to represent the entire dataset. In such a scenario, it will inevitably result in a lack of diversity in the representation of discrete features.

This issue can be attributed to several factors. First, during training, it is possible that some discrete vectors in the codebook gradually move towards the same point in high-dimensional space, effectively clustering together.

Second, training discrete vectors using $L_{reconst}$ can be inefficient. This inefficiency arises because the training rule for the Embedding Layer only updates the discrete vectors that are selected during each forward pass. Consequently, if only a small subset of discrete vectors is utilized by the entire training dataset, many other discrete vectors may never have the opportunity to learn more meaningful values.

Third, the random initialization of discrete vectors in the codebook may not align with the characteristics of the actual dataset. As a result, only a small group of discrete vectors is frequently selected during training, further reducing the diversity of the codebook's representation.

To overcome these challenges, an additional update rule is introduced for the discrete vectors in the codebook, which is complemented after the training using the encoder&quantization loss. This extra update rule aims to disperse the clogged discrete vectors, enabling them to cover a broader range of locations in the high-dimensional space. As a result, a larger number of discrete vectors in the codebook are effectively utilized to represent the entire dataset.

Note that at depth level d in the VQUNet, there are M continuous vectors, which are the outputs from the encoder block, $f_E(x)$. The codebook contains a total of $K\ 1-D$ discrete vectors. Each of the M continuous vectors will be mapped to one of the $K\ 1-D$ discrete vectors based on their Euclidean distance. To provide a concrete example, let's define C as the number of categories in the dataset. For instance, in the CIFAR10 dataset, there are $C = 10$ categories.

The number of the $1-D$ discrete vectors in the codebook Embedding Layer is set such that $K = C * M$. This configuration results in C groups of discrete vectors in the codebook, with each group denoted as K_c, where $c \in 1, 2, \cdots, C$. Within each group K_c, there are M discrete vectors.

In each training step, the values of discrete vectors are optimized based on different categories. To elaborate, let's consider a scenario where each training step involves a batch of samples with a $batch\_size = 128$. Our method divides this batch into sub-batches, denoted as $batch_c$, each corresponding to a specific category c. The output of the vector quantization process for each category, denoted as $Q_c = f_Q(batch_c)$, comprises the discrete representations for a group of samples that all belong to category c.

The reason to separate discrete representations into different sub-batches based on their categories is to facilitate the learning of discrete values that are specific to each category. Let's consider a scenario where n_c represents the number of samples belonging to category c within the entire batch.

Within the samples sub-batch of category c, there are n_c sets of discrete vectors, with each set representing the discrete representation for an individual data sample in the batch. For each data sample, its set of discrete vectors comprises M discrete vectors. We denote an individual discrete vector within this set as $e_{c,m,i}$, where $m \in 1, 2, \cdots, M$. Note that value c refers to the category, value i refers to the index of a data sample in sub-batch n_c, and value m refers to the index of an individual discrete vector in the discrete representation for one data sample.

The parameters for the codebook, represented by the Embedding Layer, are structured as a 2-D matrix with dimensions $K \times G$, where there are K 1-D vectors, with each vector having an entry size of G. Since we set $K = C * M$, where C is the number of categories and M is the number of discrete vectors in a discrete representation, we use $\theta_{c,m}$ to denote the one vector indexed at: $(c-1) * M + m$ in the sequence of K vectors in the codebook.

The update rule for the codebook is described below in Algorithm 1, where $\theta_{c,m}^{[t-1]}$ refers to the discrete vector $\theta_{c,m}$ at previous training step $t-1$. ω in Algorithm 1 refers to

the hyper-parameter to control the learning speed for the exponentially weighted moving average when updating $\theta_{c,m}^{[t]}$ at current training step t. In the implementation, we set $\omega = 0.2$.

Algorithm 1 Extra update rule for the parameters θ in the codebook

$\theta \leftarrow$ random initialization
for $c = 1, 2, \ldots, C$ **do**
  for $m = 1, 2, \ldots, M$ **do**
    $e_{avg,c,m} = \frac{1}{n_c} \sum_{i=1}^{n_c} e_{c,m,i}$
    $\theta_{c,m}^{[t]} = (1 - \omega) * \theta_{c,m}^{[t-1]} + \omega * e_{avg,c,m}$
  end for
end for

The rationale behind the additional codebook update rule, as outlined in Algorithm 1, is to effectively distribute the K discrete vectors in the codebook into distinct groups, where each group is responsible for reconstructing the data for one category.

When updating the discrete vector $\theta_{c,m}$ in the codebook, it first averages the discrete vectors $e_{c,m,i}$ over the sub-batch samples for one category, then this averaged value $e_{avg,c,m}$ is used to update $\theta_{c,m}$ with the exponentially weighted moving average.

The use of an averaged value over a sub-batch for each category to guide the codebook update aims to move each codebook discrete vector towards a centroid for multiple samples. This approach helps mitigate the issue of clogging among discrete vectors, thereby promoting a more diverse and representative set of discrete representations for the entire dataset.

Entire Training

Each training step for the VQUNet has two sub-steps.

The first sub-step is to train the network with regular gradient decent with the loss as follows:

$$L = \alpha * L_{reconst} + \beta * L_E + L_Q \tag{5.3.5}$$

where α and β are two weight parameters that can be adjusted to give different priority to the individual loss. During the empirical study, the value of α and β affect the convergence speed of $L_{reconst}$, L_E and L_Q only around the first 30 epochs, but after that, the convergence of loss does not vary much for different values of α or β. Therefore, for the rest of the experiments, $\alpha = 1$ and $\beta = 1$ are used.

The second sub-step is to correct the codebook values with operations described in Algorithm 1

5.4 Experiments

5.4.1 Experimental Setup

Once VQUNet is trained, the de-noised version of data, $X^{\sim}$, shall be used to retrain the target deep learning model. During the testing phase, VQUNet acts as an adversarial noise filter for the inputs, and then the de-noised outputs are fed into the target deep learning model for testing. In our experiments, two datasets are used for evaluation: CIFAR10 [52] and Fashion-MNIST [115]. Each sample in CIFAR10 is a $32 \times 32 \times 3$ RGB-colored image, and each sample belongs to 1 of the 10 categories of objects. Individual sample in Fashion-MNIST is a $32 \times 32 \times 1$ gray-scaled image, and there are also 10 categories of fashion objects in the dataset. CIFAR10 contains 50,000 training samples and 10,000 testing samples, and Fashion-MNIST contains 60,000 training samples and 10,000 testing samples.

For comparison, the proposed method and three other existing adversarial noise reduction methods were evaluated: the proposed VQUNet, Defense-VAE [61], High-Frequency Loss VAE (FHL_VAE) [35], and Defense-CycleGAN (CycleGAN) [36]. Following a common practice for adversarial defense evaluation, all defense methods were tested under four different adversarial attacks: Fast Gradient Sign Method (FGSM) [26], Basic Iterative Method (BIM) [56], Projected Gradient Descent (PGD) [68], Carlini and Wagner Method

(CW) [12]. All noise reduction networks were only trained once and their performance was evaluated against all four adversarial attacks for a wide range of noise levels ε. Note that there is a post-training add-on feature that can be attached to most of the noise reduction process to further boost the target model's accuracy as mentioned in [35], but to test the data reconstruction properties, the experiments did not integrate the add-on feature but just focus on the noise reduction properties.

The implementation of generating adversarial samples for all four different attacking algorithms was through Cleverhans [83] package. White-box attack refers to the case when the adversary has full access to the target models' information, e.g., parameters, structure, and hyper-parameters, to generate adversarial samples. For a black-box attack, in comparison, the attacker did not have any knowledge about the target network, but the attacker could test the target model through trial and error, such as collecting the results using adversarial inputs computed from other networks. In the experiment, 50% of the testing samples were created using white-box attack algorithms on the target model, which was a Wide Residual Network (with parameters $depth = 28$ and $k = 10$), and the other 50% were created with 3 trained DenseNet [40] with different structures.

When creating adversarial samples using different attack algorithms, we kept most of the parameters to their default values, except the noise perturbation level ε. For classification accuracy, the target model being tested was a Wide-Residual-Network [119].

5.4.2 Accuracy Degradation from Noise Reduction Process

Note that the noise reduction defense methods filter every input before the de-noised output is used by the target model, even when there is no adversarial attack. We use "filtered" and "unfiltered" to distinguish the data that have been processed by the noise reduction networks as opposed to those data that have not. We call those data that have not been modified by an attack algorithm as "benign data", and those that are crafted by an attack algorithm as "adversarial samples".

It is important to see if the filtering process would change the target model's original performance, hence, we tested the target model's accuracy on the benign data before and after the filtering process, and the result is shown in Table 5.1.

Compared with the other three defense methods in Table 5.1, "VQUNet" is the only one that has $< 1\%$ accuracy degradation for both CIFAR10 and Fashion-MNIST datasets, which means "VQUNet" incurs the least alteration to the target model's original performance.

model's acc on benign CIFAR10: 94.03%		
Methods	acc (filtered benign)	acc degradation
VQUNet	**93.08%**	**-0.95%**
Cycle_GAN	91.78%	-2.25%
HFL_VAE	86.51%	-7.52%
Defense_VAE	78.84%	-15.19%
model's acc on benign Fashion-MNIST data: 94.31%		
Methods	acc (filtered benign)	acc degradation
VQUNet	**93.39%**	**-0.92%**
Cycle_GAN	93.13%	-1.18%
HFL_VAE	92.81%	-1.50%
Defense_VAE	92.83%	-1.48%

Table 5.1: The target model's accuracy before and after the noise reduction process of different defense methods on CIFAR10 (top) and Fashion-MNIST (bottom).

5.4.3 Defense against White/Black-Box Attacks

We conducted a comparative analysis of the four noise reduction defense methods against a variety of adversarial attacks on the CIFAR10 and Fashion-MNIST datasets. The experiments measure the targeted model's classification accuracy on the adversarial samples, which are crafted in the way described in the section 5.4.1. The results of the

comparative analysis for various adversarial defense mechanisms are illustrated in Figures 5.3, 5.4, 5.5, and 5.6, for the case of being attacks by FGSM, BIM, PGD, and CW, respectively.

It is worth noting that the accuracy of the target model significantly declines to below 10% in the absence of any defense protection under all considered adversarial attacks on both datasets.

CIFAR10. From the results shown in Figure 5.3 (top), the performance of "VQUNet" maintains significantly higher accuracy than the other three defense methods across the entire spectrum of the adversarial noise levels. This trend is also observed in the comparative analysis under the adversarial attacks from BIM, PGD, and CW, as illustrated in Figures 5.4, 5.5, and 5.6, respectively. It's also worth noting that across the four adversarial attacks, as the magnitude of the noise perturbation increases, the defense performance of "VQUNet" demonstrates a slower decrease in classification accuracy, as indicated in a flatter curve in Figures 5.3, 5.4, 5.5, and 5.6. This indicates that the proposed method offers better robustness in protecting the deep learning model from diverse adversarial attacks compared to other existing defense mechanisms.

Fashion-MNIST. Under the defense analysis for the Fashion-MNIST dataset, the performance of "VQUnet" also outperforms all other three defense methods by a noticeable margin. In the comparison, as illustrated in Figures 5.3, 5.4, 5.5, and 5.6, for the defense performance under FGSM, BIM, PGD, and CW, respectively, "VQUnet" exhibits a more stable accuracy trend as the level of adversarial perturbation is increased, as a flatter curve is observed for "VQUnet" in contrast to other three defense methods. This observation implies that the proposed method exhibits a stronger resilience against adversarial noise for gray-scale image data, such as Fashion-MNIST. More importantly, it is worth noting that among all the defense methods for experiments, "VQUnet" is the only one that maintains at least 90% accuracy under all adversarial attacks across the entire noise spectrum, while other existing methods drop their defense performance below this threshold as the noise level reaches a certain point.

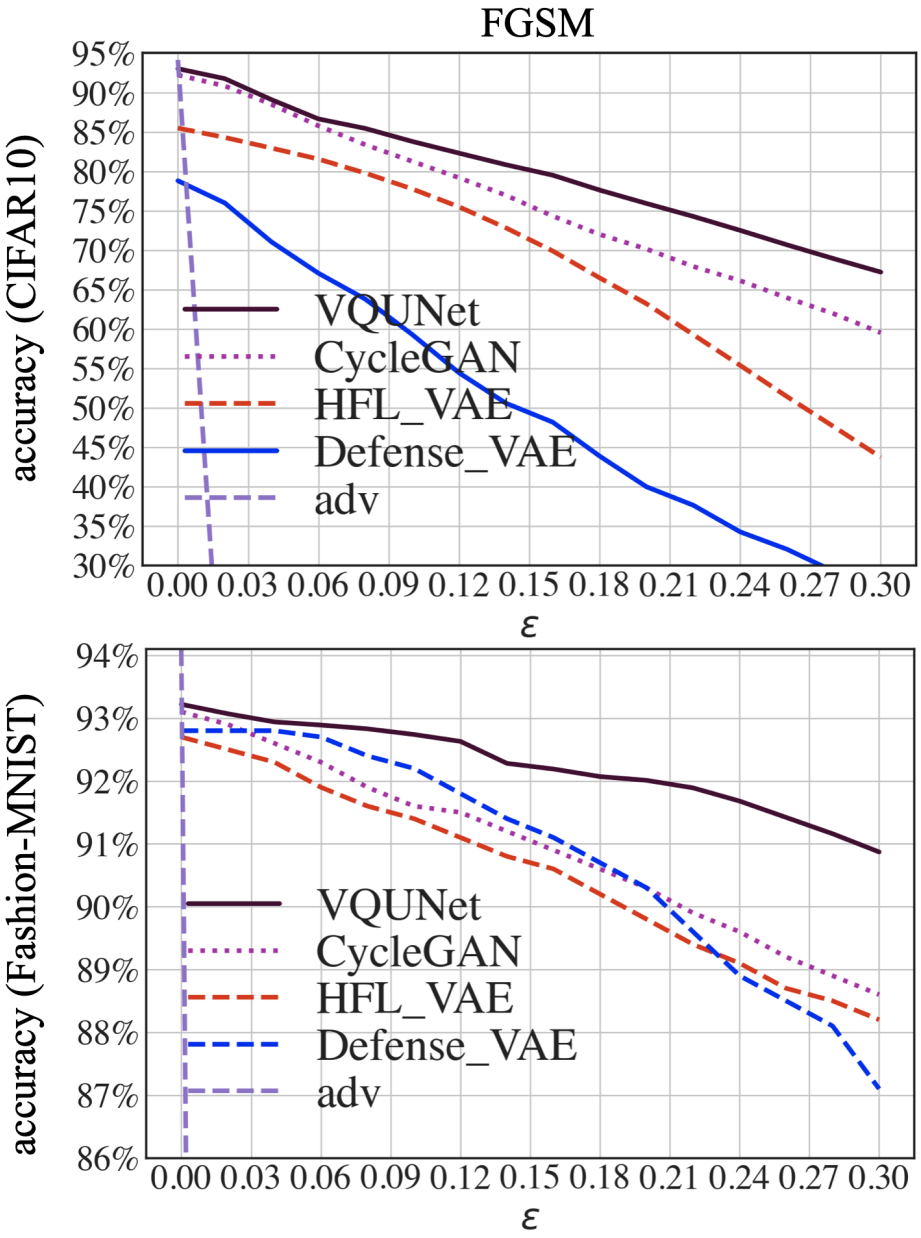


Figure 5.3: The comparison of different adversarial defense mechanisms against FGSM for CIFAR10 and Fashion-MNIST datasets.

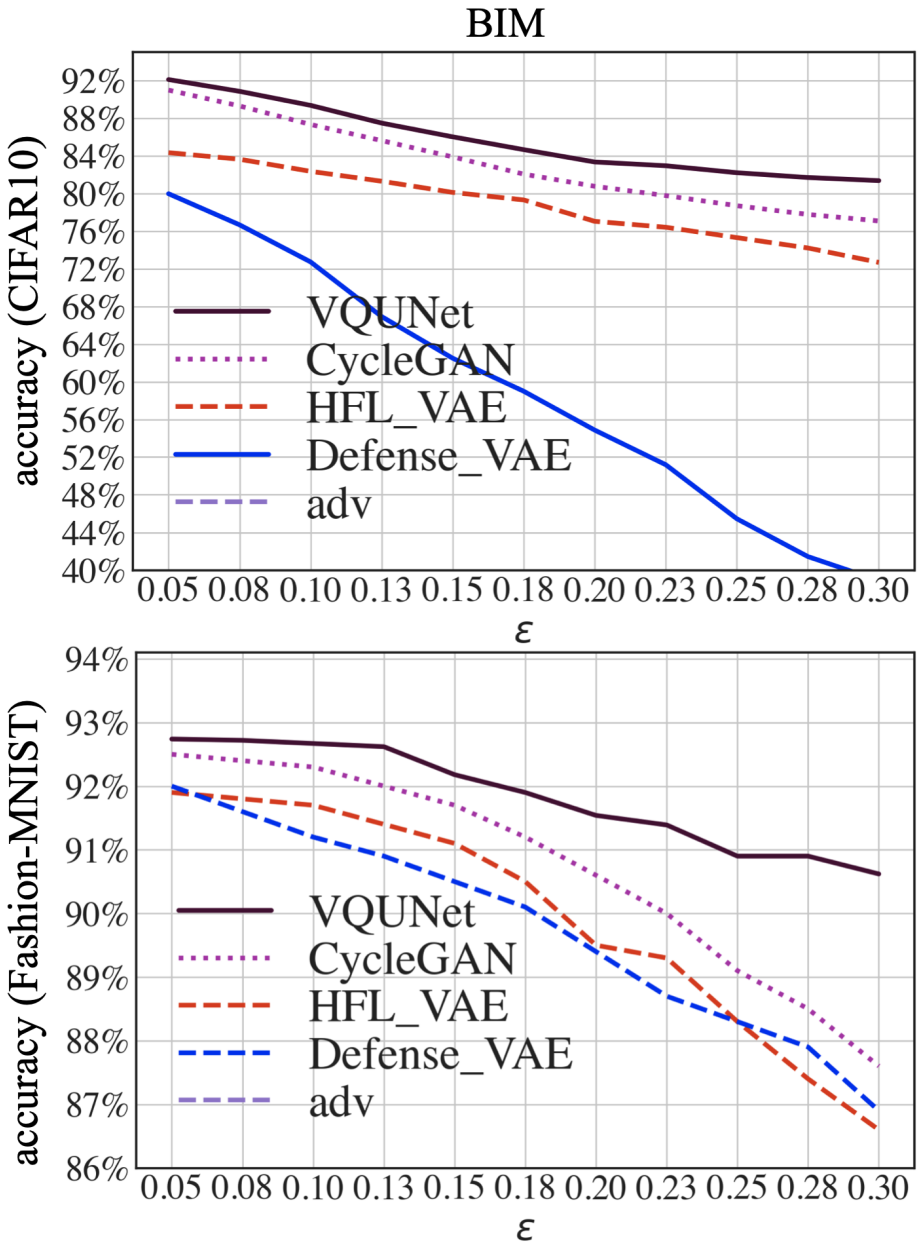


Figure 5.4: The comparison of different adversarial defense mechanisms against BIM for CIFAR10 and Fashion-MNIST datasets.

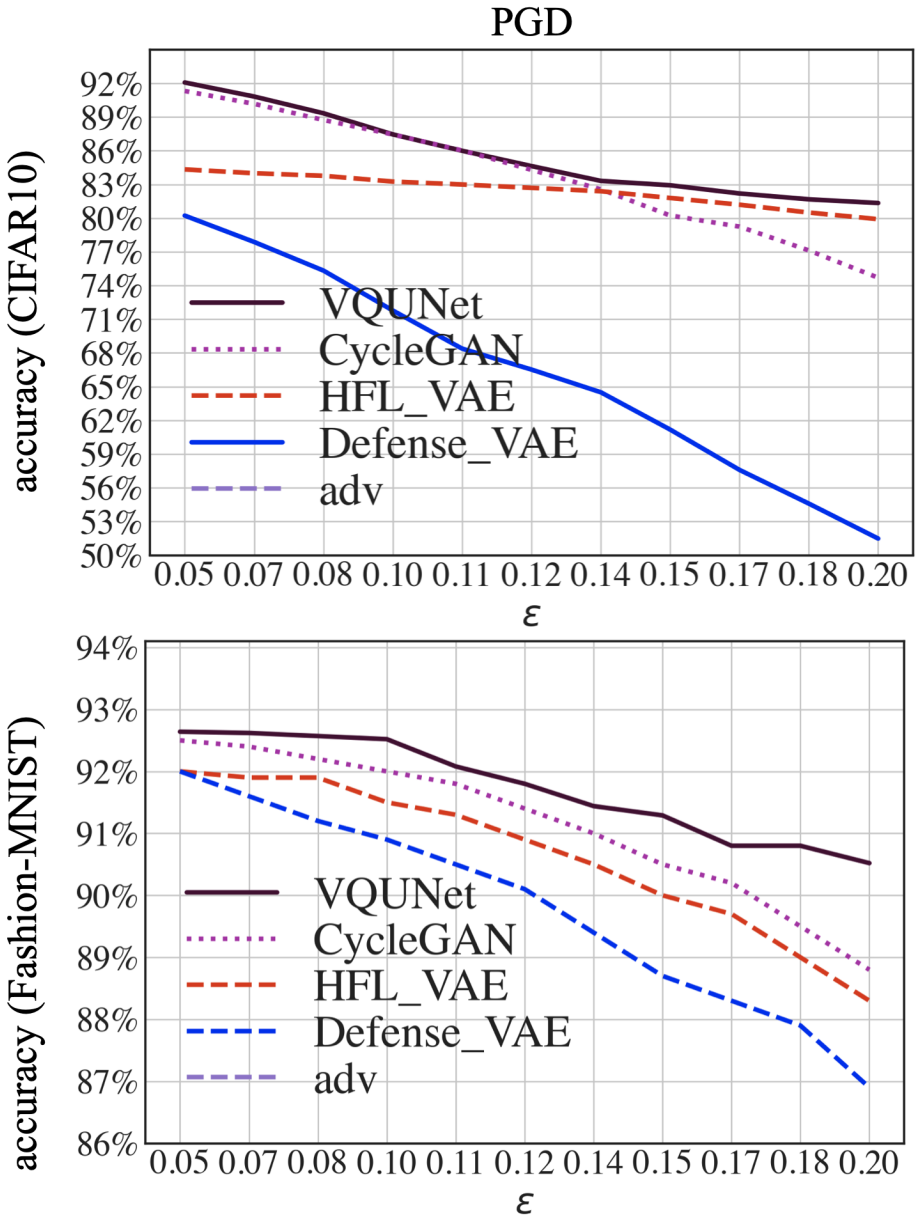


Figure 5.5: The comparison of different adversarial defense mechanisms against PGD for CIFAR10 and Fashion-MNIST datasets.

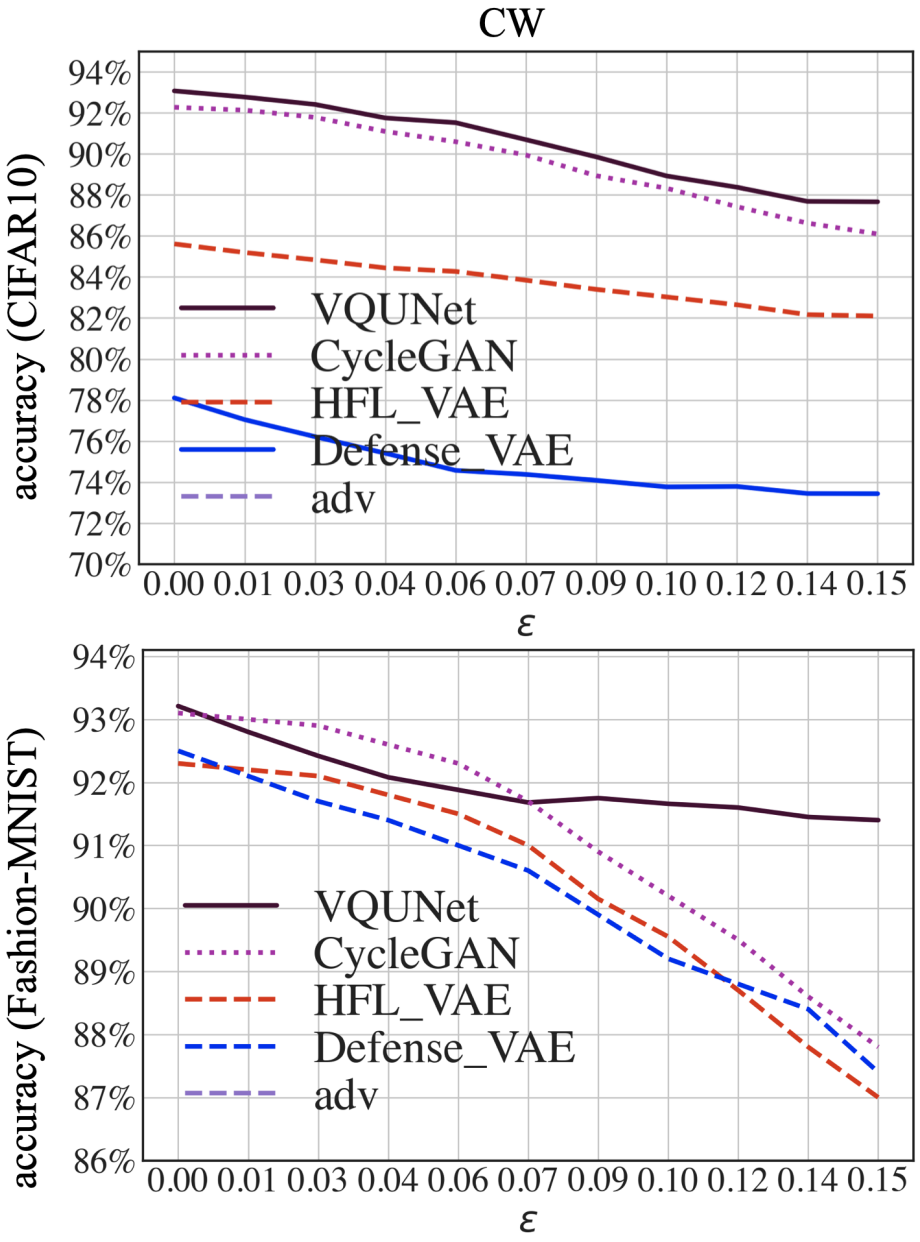


Figure 5.6: The comparison of different adversarial defense mechanisms against CW for CIFAR10 and Fashion-MNIST datasets.

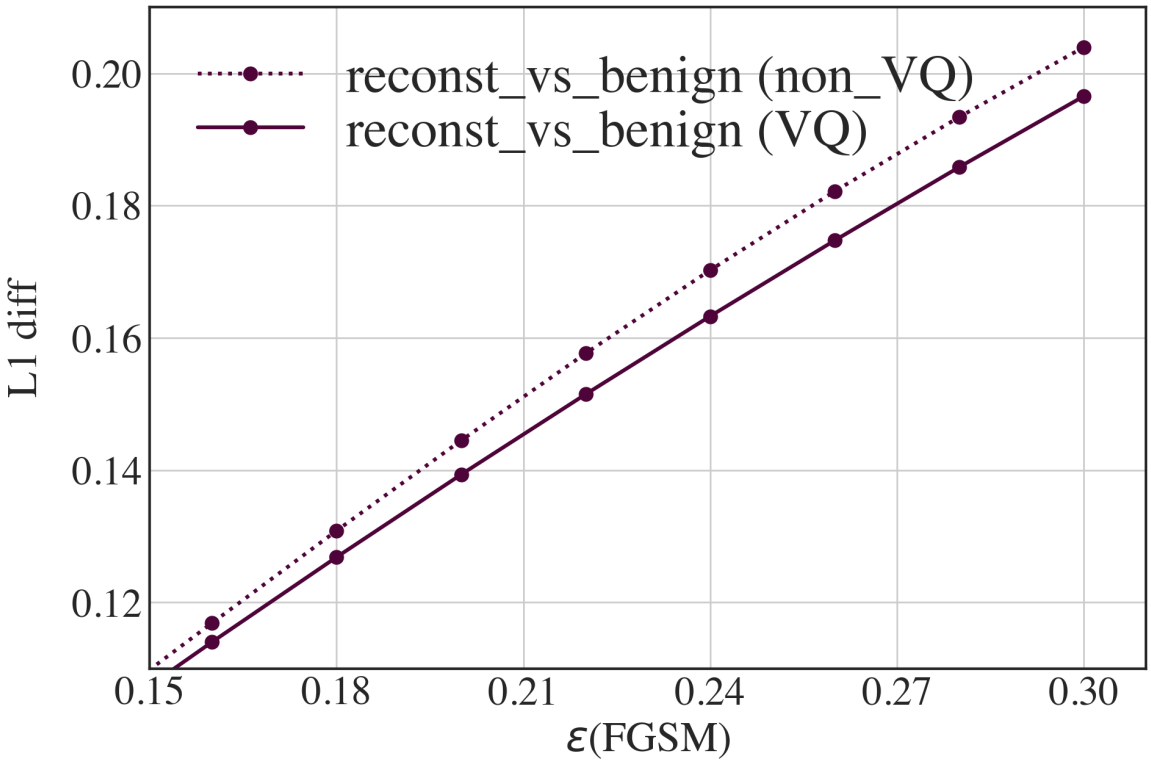


Figure 5.7: The $L1$ difference between benign images and the reconstructed images that are generated from VQUNet and Non-VQUNet for various adversarial (FGSM) noise levels.

5.4.4 Regularization from Vector Quantization

VQ v.s. Non-VQ. One of the main motivations for using vector quantization is to regularize the impact of adversarial noise on latent features. To study the regularization effect from VQ, we compared VQUNet to a second generative network called "non-VQUNet", which is nearly identical to the VQUnet except that there is no VQ block in it.

We first compared the averaged $L1$ difference between the original images and the reconstructed images from the generative networks when they were under FGSM attack. The results for both VQUNet and non-VQUNet are shown in Figure 5.7. As the adversarial perturbation increases, the reconstructed images from VQUNet demonstrated less

divergence from the original images compared to those of the non-VQUNet.

To take a closer look, our experiments also examine the behavior of encoding blocks for both VQUnet and non-VQUNet when they are under FGSM attack. Figure 5.8 shows the $L1$ difference between the encoding blocks' output before and after the network is under FGSM attack at various depth levels (d=1, d=2, and d=3) for both of the VQUNet and non-VQUNet. The value deviation of the encoders' output before and after the attack for the non-VQUNet is much larger than that of VQUNet, which means the encoding features in the non-VQUNet are much more susceptible to adversarial noise than those in VQUNet. This helps explain why the VQUNet achieves better fidelity in image reconstruction as shown in Figure 5.7.

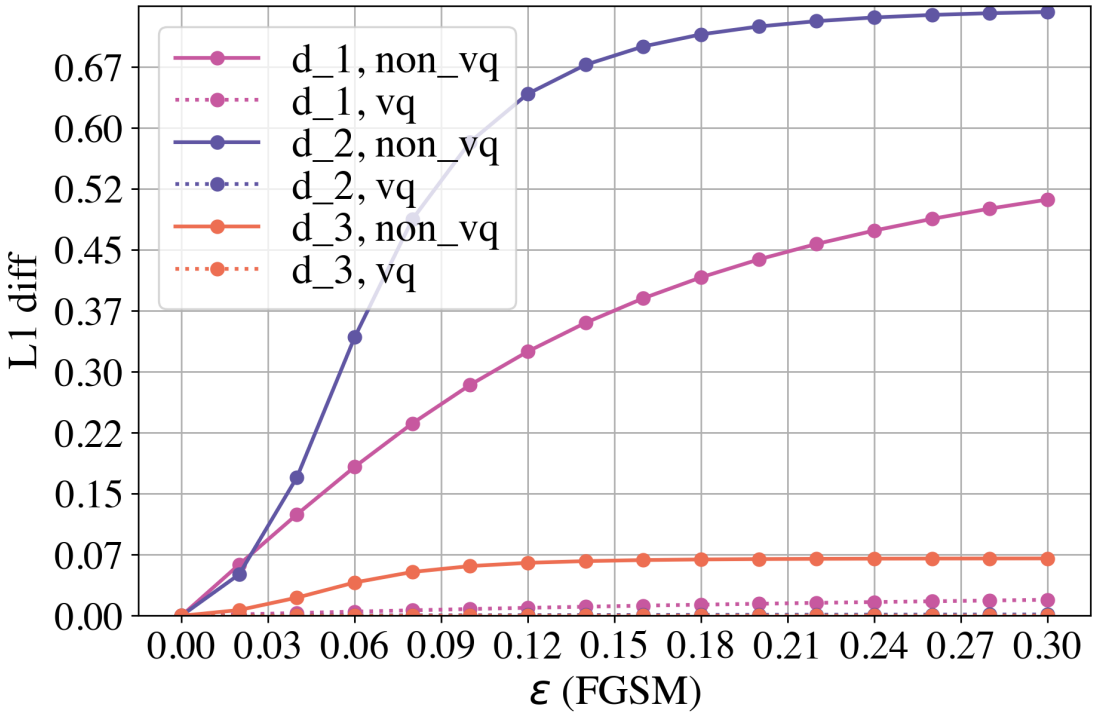


Figure 5.8: The $L1$ difference of pre_vq vectors before and after FGSM attack at various depth levels.

Pre_VQ v.s. Post_VQ. Note that during the VQ process, the pre_vq features are mapped to post_vq features through the VQ layer. In order to see the regularizing effect of VQ on the adversarial noise, we compared the $L1$ difference of the features before and after the adversarial attack (FGSM) for both pre_vq vectors and post_vq vectors at different depth levels, as shown in Figure 5.9. It's observed that both pre_vq and post_vq vectors show an increase in the divergence of their features before and after the attack as the adversarial noise level becomes larger. But for each depth level, the $L1$ difference in post_vq vectors is noticeably smaller than that of pre_vq. In other words, the adversarial noise has less negative impact on the post_vq vectors after the VQ process, compared to those of pre_vq vectors. Comparing different depth levels, both pre_vq and post_vq at deeper depth show less susceptibility to adversarial noise compared to those at shallower depth.

Code Index Change under Adversarial Attack. Note that the VQ process maps the encoder blocks' outputs to a new set of discrete codes, where each code has its unique index in the codebook. To study the effect of adversarial noise on the discretization process, we compared the how much percentage change of the selected codes' indices before and after the adversarial attack (FGSM). Figure 5.10 shows the percentage change of selected codes' indices at different depth levels (d=1, d=2, and d=3) of VQUNet when it's under the FGSM attack. From the observation, the VQ layer at a deeper level maintains its original codes' indices relatively better than that at a shallower depth. As the adversarial noise increases, the change of indices rises more rapidly at the beginning, but this change slows down as the adversarial noise level becomes larger. This means that the VQ process shows more susceptibility to noise perturbation when the noise level is small. However as the adversarial level becomes larger, the VQ process maintains a similar selection of the discrete codes when it's under the attack.

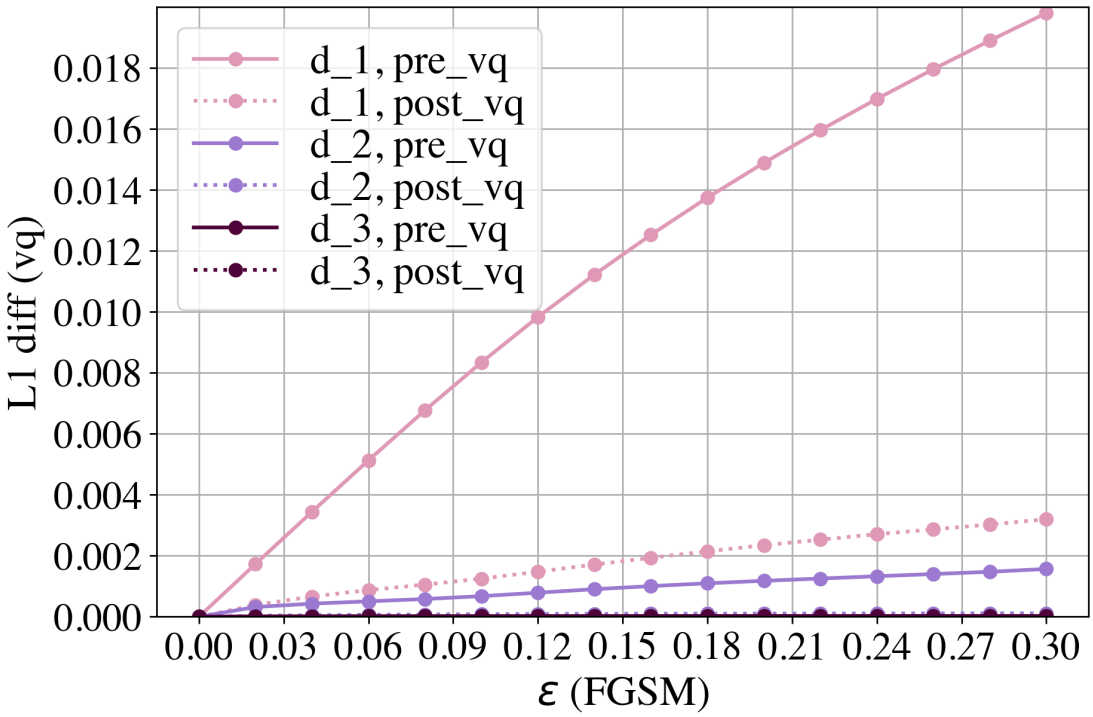


Figure 5.9: L1 difference of features before and after the FGSM attack for pre_vq and post_vq vectors at various depth levels.

5.5 Chapter Summary

We introduced a new noise-reduction model, VQUNet, for adversarial defense. The features include a hierarchical structure for high-fidelity image reconstruction and a vector quantization mechanism that helps regularize the impact from the adversarial noise for better data reconstruction. We showed that under various adversarial attacks and a wide range of noise perturbation levels, VQUNet outperforms other state-of-the-art defense methods on both CIFAR10 and Fashion-MNIST datasets with a noticeable margin. In addition, in the absence of an adversarial attack, only VQUNet controlled its accuracy degradation $< 1\%$ from the target model's original accuracy performance compared to other methods.

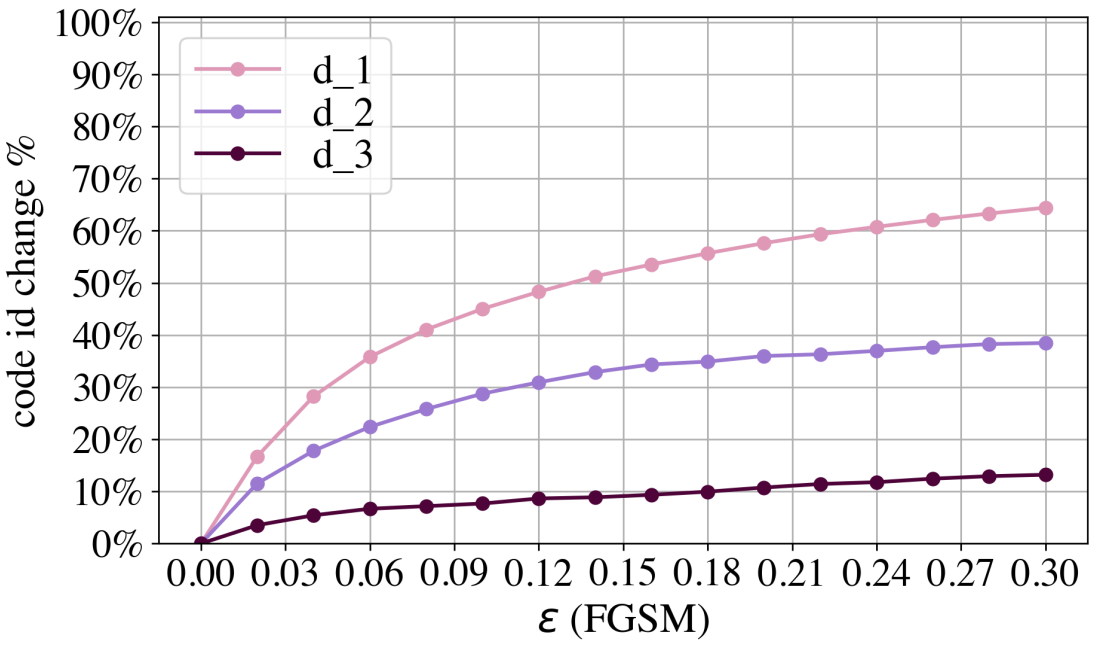


Figure 5.10: The percentage change of the indices of selected discrete codes before and after the FGSM attack.

Chapter 6

Summary

In this dissertation, we discussed the overall advancements in adversarial attacks and defense mechanisms. Despite the recent progress, there are still challenges inherent in designing effective and robust adversarial defense strategies, such as the reduced model performance on clean data, limited robustness against sophisticated attacks, and the lack of consistency over various adversarial noise levels. To address those challenges, the dissertation proposed three innovative defense methods.

Chapter 3 introduced a novel adversarial defense method, focusing on enhancing data reconstruction quality through a Spacial Frequency Loss-enhanced VAE. This method improved the reliability of deep learning models by integrating multiple forms of supporting evidence, using Bayesian Updates to enhance confidence in final decisions. A key aspect was the inclusion of a collective voting result from randomly selected post-VAE classifiers, which consistently improved the accuracy by a notable margin. The findings illustrated the method's superiority over other contemporary defense methods.

In Chapter 4, an innovative adversarial noise reduction method, Defense-CycleGAN, was proposed to enhance the robustness of deep learning systems against adversarial attacks. This approach entailed the creation of an end-to-end training framework comprising two GANs, guided by cycle consistency loss, to independently reconstruct high-fidelity data

while simultaneously removing harmful noise from each other's outputs. This novel framework resulted in high-fidelity data reconstruction, thereby facilitating the overall performance of targeted models against various adversarial attacks while mitigating the performance degradation issue encountered by previous defense methods.

Chapter 5 introduced a novel framework, VQUNet, that aims to reduce adversarial noise and enhance data reconstruction fidelity. Central to reproducing accurate data was the hierarchical structure of the data reconstruction pipeline, which integrates both primitive and highly abstract latent features to reproduce realistic data. Moreover, the proposed approach incorporated a dynamically learned vector quantization mapping from continuous vectors to quantized codes to effectively regularize the impact of adversarial noise. Our study demonstrated that VQUNet outperformed existing state-of-the-art defense methods across diverse adversarial attacks and noise perturbation levels on CIFAR10 and Fashion-MNIST datasets. Even in the absence of adversarial attacks, VQUNet demonstrated remarkable stability by maintaining its accuracy within a marginal 1% degradation from the original model's performance, a level of robustness not achieved by other existing methods.

In conclusion, the dissertation presented three innovative adversarial defense methods aimed at improving the reliability and security of deep learning systems against adversarial attacks. Extensive experimental evaluations exhibited the efficacy and robustness of the proposed methods. The analytical results demonstrated the proposed methods' superiority over existing defense methods under a broad range of adversarial attacks and perturbation levels. Through meticulous analysis, it offered valuable insights into the mechanisms of adversarial attacks and defense strategies, facilitating the ongoing efforts in the field to develop more potent and efficient defense methods.

Appendix A

VAE Architecture

Encoder	Decoder
Conv(64,4,2)+BN+ReLU	Dense(4096)+ReLU
Dropout(0.2)	Reshape(4,4,256)
Conv(128,4,2)+BN+ReLU	ConvTrans(512,4,2)+ReLU
Dropout(0.2)	ConvTrans(256,4,2)+ReLU
Conv(256,4,2)+BN+ReLU	ConvTrans(128,4,2)+ReLU
Dropout(0.2)	ConvTrans(64,4,2)+ReLU
Conv(512,4,2)+BN+ReLU	ConvTrans(3,4,2)+ReLU
Dropout(0.2)	
Flatten	
Dense1(1024), Dense2(2014)	

Table A.1: Network structure of VAE used for de-noising and image reconstruction.

The VAE's architecture used in the proposed defense method is shown in Table A.1. Except for the Decoder needs to be adjusted for the dimension difference between Fashion-MNIST and CIFAR10, the rest of the structure is the same for the experiments on both data sets.

Appendix B

Classifier Structures

The classifier models that were used in comparison experiments between different defense mechanisms are shown in Table B.1.

The post-VAE classifiers are created using the implementation from the GitHub repository [60]. Residual-Networks [33], Wide-Residual-Networks [119] and DenseNet [40] were used to create the post-VAE classifiers. Table B.2, B.3 and B.4 show the parameters that were used to create the 12 different post-VAE classifiers (the rest of the parameters were set with their default values).

A		B		C	
F-MNIST	CIFAR10	F-MNIST	CIFAR10	F-MNIST	CIFAR10
Conv(64,5,1)	Conv(32,3,1)	Dropout(0.2)	Conv(32,3,2)	Conv(128,3,1)	Conv(32,3,1)
ReLU	ELU,BN	Conv(64,8,2)	ReLU,BN	ReLU	ELU,BN
Conv(64,5,2)	Conv(32,3,1)	ReLU	Conv(32,3,2)	Conv(64,5,2)	Conv(32,3,1)
ReLU	ELU,BN	Conv(128,6,2)	ReLU,BN	ReLU	ELU,BN
Flatten	Pooling(2)	ReLU	Pooling(2)	Flatten	Pooling(2)
Dropout(0.25)	Dropout(0.2)	Conv(128,5,1)	Dropout(0.2)	Dropout(0.25)	Dropout(0.2)
Dense(128)	Conv(64,3,1)	ReLU	Conv(64,3,2)	Dense(0.25)	Conv(64,3,1)
ReLU	ELU,BN	Flatten	ReLU,BN	ReLU	ELU,BN
Dropout(0.5)	Conv(64,3,1)	Dropout(0.5)	Conv(64,3,2)	Dropout(0.5)	Conv(128,3,1)
Dense(10)	ELU,BN	Dense(10)	ReLU,BN	Dense(10)	ELU,BN
Softmax	Pooling(2)	Softmax	Pooling(2)	Softmax	Pooling(2)
	Dropout(0.2)		Dropout(0.2)		Dropout(0.2)
	Conv(128,3,1)		Conv(128,3,1)		Dense(10)
	ELU,BN		ReLU,BN		Softmax
	Conv(128,3,1)		Conv(128,3,1)		
	ELU,BN		ReLU,BN		
	Pooling(2)		Pooling(2)		
	Dropout(0.2)		Dropout(0.2)		
	Dense(10)		Dense(10)		
	Softmax		Softmax		

Table B.1: Detailed network structures of different classifiers used for white-box attacks.

Params	stack_n	epochs	batch_size
ResNet A	3	50	128
ResNet B	5	50	128
ResNet C	10	100	128
ResNet D	18	100	128

Table B.2: Four different ResNet architectures used throughout the experiments.

Params	depth	growth_rate	epochs	batch_size
DenseNet A	50	12	100	128
DenseNet B	50	24	100	128
DenseNet C	100	12	100	128
DenseNet D	100	24	80	256

Table B.3: Four different DenseNet architectures were used throughout the experiments.

Params	depth	wide	epochs	batch_size
WResNet A	8	8	80	128
WResNet B	16	8	80	128
WResNet C	16	10	80	256
WResNet D	28	10	80	256

Table B.4: Four different Wide-ResNet architectures were used throughout the experiments.

Bibliography

[1] Alankrita Aggarwal, Mamta Mittal, and Gopi Battineni. Generative adversarial network: An overview of theory and applications. *International Journal of Information Management Data Insights*, 1(1):100004, 2021.

[2] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018.

[3] Naveed Akhtar, Ajmal Mian, Navid Kardan, and Mubarak Shah. Advances in adversarial attacks and defenses in computer vision: A survey. *IEEE Access*, 9:155161–155196, 2021.

[4] Mislav Balunovic and Martin Vechev. Adversarial training and provable defenses: Bridging the gap. In *International Conference on Learning Representations*, 2019.

[5] Thomas Bayes. Lii. an essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfr s. *Philosophical transactions of the Royal Society of London*, 0(53):370–418, 1763.

[6] Benchmarks.ai. Cifar-10 benchmarks, 2020.

[7] Hervé Bourlard and Yves Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, 59(4):291–294, 1988.

[8] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[9] Jacob Buckman, Aurko Roy, Colin Raffel, and Ian Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. In *International Conference on Learning Representations*, 2018.

[10] Lei Cai, Hongyang Gao, and Shuiwang Ji. Multi-stage variational auto-encoders for coarse-to-fine image generation. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 630–638. SIAM, 2019.

[11] Nicholas Carlini, Guy Katz, Clark Barrett, and David L Dill. Ground-truth adversarial examples. *openreview.net*, 2018.

[12] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[13] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*, 2018.

[14] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. Ead: elastic-net attacks to deep neural networks via adversarial examples. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[15] Yun-Chun Chen, Po-Hsiang Huang, Li-Yu Yu, Jia-Bin Huang, Ming-Hsuan Yang, and Yen-Yu Lin. Deep semantic matching with foreground detection and cycle-consistency. In *Asian Conference on Computer Vision*, pages 347–362. Springer, 2018.

[16] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR, 2019.

[17] Richard O Duda, Peter E Hart, et al. *Pattern classification and scene analysis*, volume 3. Wiley New York, 1973.

[18] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. Temporal cycle-consistency learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1801–1810, 2019.

[19] Scott E Fahlman, Geoffrey E Hinton, and Terrence J Sejnowski. Massively parallel architectures for al: Netl, thistle, and boltzmann machines. In *National Conference on Artificial Intelligence, AAAI*, 1983.

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

[21] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333, 2015.

[22] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine learning*, 29(2-3):131–163, 1997.

[23] Partha Ghosh, Arpan Losalka, and Michael J Black. Resisting adversarial attacks using gaussian mixture variational autoencoders. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 541–548, 2019.

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

[25] 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*, 27, 2014.

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

[27] Gaurav Goswami, Akshay Agarwal, Nalini Ratha, Richa Singh, and Mayank Vatsa. Detecting and mitigating adversarial perturbations for robust face recognition. *International Journal of Computer Vision*, 127(6-7):719–742, 2019.

[28] Chirag Goyal. Deep understanding of discriminative and generative models in machine learning, 2021.

[29] Kathrin Grosse, Praveen Manoharan, Nicolas Papernot, Michael Backes, and Patrick McDaniel. On the (statistical) detection of adversarial examples. *arXiv preprint arXiv:1702.06280*, 2017.

[30] Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068*, 2014.

[31] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

[32] GM Harshvardhan, Mahendra Kumar Gourisaria, Manjusha Pandey, and Siddharth Swarup Rautaray. A comprehensive survey and analysis of generative models in machine learning. *Computer Science Review*, 38:100285, 2020.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[34] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[35] Zhixun He and Mukesh Singhal. Adversarial defense through high frequency loss variational autoencoder decoder and bayesian update with collective voting. In *2021 17th International Conference on Machine Vision and Applications (MVA)*, pages 1–7. IEEE, 2021.

[36] Zhixun He and Mukesh Singhal. Defense-cyclegan: A defense mechanism against adversarial attacks using cyclegan to reconstruct clean images. In *2022 3rd International Conference on Pattern Recognition and Machine Learning (PRML)*, pages 173–179, 2022.

[37] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

[38] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[39] Geoffrey E Hinton and Russ R Salakhutdinov. A better way to pretrain deep boltzmann machines. *Advances in Neural Information Processing Systems*, 25, 2012.

[40] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/-1608.06993, 2016.

[41] Naoyuki Ichimura. Spatial frequency loss for learning convolutional autoencoders. *arXiv preprint arXiv:1806.02336*, 2018.

[42] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Transactions on Graphics (ToG)*, 36(4):1–14, 2017.

[43] Abdul Jabbar, Xi Li, and Bourahla Omar. A survey on generative adversarial networks: Variants, applications, and training. *ACM Computing Surveys (CSUR)*, 54(8):1–49, 2021.

[44] Xiaojun Jia, Xingxing Wei, Xiaochun Cao, and Hassan Foroosh. Comdefend: An efficient image compression model to defend adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6084–6092, 2019.

[45] Lingyun Jiang, Kai Qiao, Ruoxi Qin, Linyuan Wang, Wanting Yu, Jian Chen, Haibing Bu, and Bin Yan. Cycle-consistent adversarial gan: The integration of adversarial attack and defense. *Security and Communication Networks*, 2020, 2020.

[46] Guoqing Jin, Shiwei Shen, Dongming Zhang, Feng Dai, and Yongdong Zhang. Ape-gan: Adversarial perturbation elimination with gan. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3842–3846. IEEE, 2019.

[47] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Forward-backward error: Automatic detection of tracking failures. In *2010 20th international conference on pattern recognition*, pages 2756–2759. IEEE, 2010.

[48] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[49] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[50] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[51] Diederik P Kingma, Max Welling, et al. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019.

[52] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). -, 2009.

[53] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[54] Nilesh Kulkarni, Abhinav Gupta, and Shubham Tulsiani. Canonical surface mapping via geometric cycle consistency. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2202–2211, 2019.

[55] S Kullback and RA Leibler. 10.1214/aoms/1177729694. *Ann. Math. Stat*, 22:79–86, 1951.

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

[57] Yann LeCun. Phd thesis: Modeles connexionnistes de l'apprentissage (connectionist learning models). -, 1987.

[58] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[59] Hyeungill Lee, Sungyeob Han, and Jungwoo Lee. Generative adversarial trainer: Defense to adversarial perturbations with gan. *arXiv preprint arXiv:1705.03387*, 2017.

[60] Wei Li. cifar-10-cnn: Play deep learning with cifar datasets. `https://github.com/BIGBALLON/cifar-10-cnn`, 2017.

[61] Xiang Li and Shihao Ji. Defense-vae: A fast and accurate defense against adversarial attacks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 191–207. Springer, 2019.

[62] Yao Li, Minhao Cheng, Cho-Jui Hsieh, and Thomas CM Lee. A review of adversarial attack and defense for classification methods. *The American Statistician*, 76(4):329–345, 2022.

[63] Qi Liu, Tao Liu, Zihao Liu, Yanzhi Wang, Yier Jin, and Wujie Wen. Security analysis and enhancement of model compressed deep learning systems under adversarial attacks. In *2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 721–726. IEEE, 2018.

[64] Xuanqing Liu and Cho-Jui Hsieh. Rob-gan: Generator, discriminator, and adversarial attacker. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11234–11243, 2019.

[65] Xuanqing Liu, Yao Li, Chongruo Wu, and Cho-Jui Hsieh. Adv-bnn: Improved adversarial defense through robust bayesian neural network. In *International Conference on Learning Representations*, 2018.

[66] Zihao Liu, Qi Liu, Tao Liu, Nuo Xu, Xue Lin, Yanzhi Wang, and Wujie Wen. Feature distillation: Dnn-oriented jpeg compression against adversarial examples. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 860–868. IEEE, 2019.

[67] Jiajun Lu, Theerasit Issaranon, and David Forsyth. Safetynet: Detecting and rejecting adversarial examples robustly. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 446–454, 2017.

[68] 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.

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

[70] David Marr and Ellen Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 207(1167):187–217, 1980.

[71] Edward James McShane. Jensen's inequality. *Bulletin of the American Mathematical Society*, 43(8):521–527, 1937.

[72] Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 135–147. ACM, 2017.

[73] Agnieszka Mikołajczyk and Michał Grochowski. Data augmentation for improving deep learning in image classification problem. In *2018 international interdisciplinary PhD workshop (IIPhDW)*, pages 117–122. IEEE, 2018.

[74] Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning. *J. Mach. Learn. Res.*, 21(132):1–62, 2020.

[75] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[76] Andrew Ng. Introduction to machine learning, 2011.

[77] Andrew Ng, Jiquan Ngiam, Chuan Yu Foo, and Y Mai. Deep learning. *CS229 Lecture Notes*, pages 1–30, 2014.

[78] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. Multimodal deep learning. In *ICML*, 2011.

[79] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[80] Achraf Oussidi and Azeddine Elhassouny. Deep generative models: Survey. In *2018 International Conference on Intelligent Systems and Computer Vision (ISCV)*, pages 1–8. IEEE, 2018.

[81] Art B. Owen. *Monte Carlo theory, methods and examples*. Art B. Owen, 2013.

[82] Zhaoqing Pan, Weijie Yu, Xiaokai Yi, Asifullah Khan, Feng Yuan, and Yuhui Zheng. Recent progress on generative adversarial networks (gans): A survey. *IEEE Access*, 7:36322–36333, 2019.

[83] 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.

[84] Nicolas Papernot, Ian Goodfellow, Ryan Sheatsley, Reuben Feinman, and Patrick McDaniel. cleverhans v2. 0.0: an adversarial machine learning library. *arXiv preprint arXiv:1610.00768*, 10, 2016.

[85] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

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

[87] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51(5):1–36, 2018.

[88] Shilin Qiu, Qihe Liu, Shijie Zhou, and Chunjiang Wu. Review of artificial intelligence adversarial attack and defense technologies. *Applied Sciences*, 9(5):909, 2019.

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

[90] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. In *Advances in neural information processing systems*, pages 14866–14876, 2019.

[91] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International conference on machine learning*, pages 1060–1069. PMLR, 2016.

[92] Kui Ren, Tianhang Zheng, Zhan Qin, and Xue Liu. Adversarial attacks and defenses in deep learning. *Engineering*, 6(3):346–360, 2020.

[93] Joseph Rocca. Understanding variational autoencoders (vaes), Sep 2019.

[94] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[95] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017.

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

[97] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *CoRR*, abs/1805.06605, 2018.

[98] Meet Shah, Xinlei Chen, Marcus Rohrbach, and Devi Parikh. Cycle-consistency for robust visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6649–6658, 2019.

[99] Shiwei Shen, Guoqing Jin, Ke Gao, and Yongdong Zhang. Ape-gan: Adversarial perturbation elimination with gan. *arXiv preprint arXiv:1707.05474*, 2017.

[100] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019.

[101] 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.

[102] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

[103] Guangzhi Sun, Yu Zhang, Ron J Weiss, Yuan Cao, Heiga Zen, Andrew Rosenberg, Bhuvana Ramabhadran, and Yonghui Wu. Generating diverse and natural text-to-speech samples using a quantized fine-grained vae and autoregressive prosody prior. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6699–6703. IEEE, 2020.

[104] Lichao Sun, Yingtong Dou, Carl Yang, Ji Wang, Philip S Yu, Lifang He, and Bo Li. Adversarial attack and defense on graph data: A survey. *arXiv preprint arXiv:1812.10528*, 2018.

[105] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2013.

[106] Pooya Tavallali, Vahid Behzadan, Azar Alizadeh, Aditya Ranganath, and Mukesh Singhal. Adversarial label-poisoning attacks and defense for general multi-class models based on synthetic reduced nearest neighbor. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 3717–3722. IEEE, 2022.

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

[108] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning, 2017.

[109] Jason Van Tuinen, Aditya Ranganath, Goran Konjevod, Mukesh Singhal, and Roummel Marcia. Novel adversarial defense techniques for white-box attacks. In *2022 21st*

IEEE International Conference on Machine Learning and Applications (ICMLA), pages 617–622. IEEE, 2022.

[110] Zhengwei Wang, Qi She, and Tomas E Ward. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Computing Surveys (CSUR)*, 54(2):1–38, 2021.

[111] Will Williams, Sam Ringer, Tom Ash, John Hughes, David MacLeod, and Jamie Dougherty. Hierarchical quantized autoencoders. *arXiv preprint arXiv:2002.08111*, 2020.

[112] Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5286–5295. PMLR, 2018.

[113] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610*, 2018.

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

[115] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[116] Han Xu, Yao Ma, Hao-Chen Liu, Debayan Deb, Hui Liu, Ji-Liang Tang, and Anil K Jain. Adversarial attacks and defenses in images, graphs and text: A review. *International Journal of Automation and Computing*, 17(2):151–178, 2020.

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

[118] Xitong Yang. Understanding the variational lower bound. *variational lower bound, ELBO, hard attention*, 13:1–4, 2017.

[119] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *CoRR*, abs/1605.07146, 2016.

[120] Junhai Zhai, Sufang Zhang, Junfen Chen, and Qiang He. Autoencoder and its various variants. In *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 415–419. IEEE, 2018.

[121] Lei Zhou, Chunlei Cai, Yue Gao, Sanbao Su, and Junmin Wu. Variational autoencoder for low bit-rate image compression. In *CVPR Workshops*, pages 2617–2620, 2018.

[122] Tinghui Zhou, Philipp Krahenbuhl, Mathieu Aubry, Qixing Huang, and Alexei A Efros. Learning dense correspondence via 3d-guided cycle consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 117–126, 2016.

[123] Jinlin Zhu, Guohao Peng, and Danwei Wang. Dual-domain-based adversarial defense with conditional vae and bayesian network. *IEEE Transactions on Industrial Informatics*, 17(1):596–605, 2020.

[124] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *European conference on computer vision*, pages 597–613. Springer, 2016.

[125] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.