Diffusion Inspired Image Watermarking against Adversarial Attacks

R. R. Walgama

2025

Diffusion Inspired Image Watermarking against Adversarial Attacks

Ramindu Walgama
Index No: 20001959

Supervisor: Dr. Ajantha Atukorale
Co-Supervisor: Mr. Yasas Mahima

May 2025

Submitted in partial fulfillment of the requirements of the
B.Sc. (Honours) in Computer Science Final Year Project

Abstract

Recent years have seen Deep Learning Neural Networks emerging in the Computer Vision domain across various industries like medical, autonomous vehicles, robotics, and the defense industry, with applications in classification, object detection, and many other tasks. Despite these improvements, these networks have been exposed to adversarial attacks. Even a small amount of perturbation can fool deep learning networks. In the context of classification tasks, this can lead to incorrect class predictions. This research aims to solve this problem by introducing a novel diffusion-inspired model which adds noise on top of the image as a watermark before transmitting it through the network. From the receiver's end, a deployed diffusion-inspired denoiser extracts those noise layers, aiming to purify the perturbations added by the attacker, and the deployed classifier aims to classify whether an adversarial attack exists or not based on the purified image. The extensive experiments showcase that these models can achieve up to **99.9%** uniform accuracy across different attacks and above **94%** accuracy across different datasets. The performance evaluations prove that the proposed solution can identify adversarial samples in **less than half a second**.

Keywords— Adversarial Attacks, Adversarial Defenses, Deep Learning, Diffusion Model

Subject descriptors:

- **Computing methodologies** → Artificial intelligence → Computer vision → Computer vision representations → Appearance and texture representations
- **Computing methodologies** → Machine learning → Machine learning approaches → Neural networks
- **Computing methodologies** → Machine learning → Machine learning approaches → Bio-inspired approaches → Generative and developmental approaches

Preface

I certify that this dissertation does not incorporate, without acknowledgment, any material previously submitted for a degree or diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, be made available for photocopying and for interlibrary loans, and for the title and abstract to be made available to outside organizations.

Candidate Name: Ramindu Walgama

... [illegible] ......

Signature of Candidate Date: ...25th of Jun, 2025...

This is to certify that this dissertation is based on the work of

Mr. Ramindu Walgama

under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Supervisor's Name: Dr. D. A. S. Atukorale

[illegible]
....

Signature of Supervisor Date: ...25th of Jun, 2025...

Co-Supervisor's Name: Mr. Yasas Mahima

... [illegible] ...

Signature of Supervisor Date: ...25th of Jun, 2025...

Contents

List of Figures

List of Tables

List of Acronyms

DNN Deep Neural Network

GAN Generative Adversarial Networks

CNN Convolutional Neural Network

RNN Recurrent Neural Network

MitM Man-in-the-Middle

FGSM Fast-Gradient Sign Method

PGD Projected Gradient Descent

BIM Basic Iterative Method

C&W Carlini and Wagner

HGD High-level representation Guided Denoiser

VAEs Variational Auto-Encoders

DDPM Denoising Diffusion Probabilistic Models

DDIM Denoising Diffusion Implicit Models

SGM Score-based Generative Models

SDE Stochastic Differential Equations

KL Kullback-Leibler

MAE Mean Absolute Error

SSIM Structural Similarity Index Measure

PSNR Peak Signal-to-Noise Ratio

NLP Natural Language Processing

CHAPTER 1: INTRODUCTION

1.1 Chapter Overview

This chapter presents the background and motivation for the research, followed by a discussion of the identified research gap. The research gap is emphasized through the formulation of a hypothesis aimed at addressing the problem while achieving the proposed research objectives. These objectives are aligned with research questions. The chapter also outlines the research methodology and the publication plan.

1.2 Introduction and Motivation

In recent years, Deep Neural Network (DNN) techniques developed for computer vision have been an emerging area in various application domains such as object recognition (He et al., 2016; Zeng et al., 2021), image fusion (Ma et al., 2020), image processing (Dai et al., 2019), and image segmentation (Liu et al., 2019). These Deep Learning (DL) techniques have been used in various industries such as medical, autonomous vehicles, social networks, defense, robotics, etc. Various generative models have been proposed for these tasks, such as Variational Auto-Encoders (VAEs), Generative Adversarial Networks (GAN)s, and diffusion models. However, recent research reveals that DNNs like Convolutional Neural Network (CNN)s are susceptible to intentionally crafted adversarial perturbations (Wang et al., 2021). Attackers have exploited this limitation to alter the network's predictions by adding small perturbations to the network's inference or training data.

With the rapid growth of emerging generative models such as diffusion models, image generation has become popular within Computer Vision (CV) research communities (Ho et al., 2020). This research is based on the concepts of diffusion models and explores the idea of extracting noise layers from the noisy image through the diffusion inspired reverse process. This approach has been combined with the watermarking concepts of Quiring et al. (2017) research and Fei et al. (2022) studies to implement a reactive defense system against adversarial attacks using noise as a watermark.

Attackers may compromise images during the source image generation, such as with a camera, or perform attacks on stored data as well as during data transmission to execute a Man-in-the-Middle (MitM) adversary (Moon et al., 2022). The proposed solution is capable of detecting adversarial attacks on both training and inference images before downstream processing to DNNs. Thus, adversarial samples can be rejected before causing erroneous predictions.

1.3 Research Gap

In recent years, DNN networks have become popular, but they are highly sensitive to adversarial noise in images, which can lead to incorrect predictions even with a small amount of noise (Dai et al., 2022; Apostolidis and Papakostas, 2021; Zhang et al., 2021). There have been research conducted on image watermarking to hide extra information in an image, such as StegaStamp by Tancik et al. (2020), but lacks of hiding the image for privacy while concerning about the adversaries. Further, noise-based watermarking has received considerably less attention. To the best of the authors' knowledge, no solutions based on image watermarking have been proposed that use noise to identify where an image has been poisoned by external adversaries.

1.4 Research Aim, Research Questions and Objectives

1.4.1 Research Hypothesis

The diffusion model concept is based on a Markov chain, where it adds a set of noise layers iteratively ($1 \rightarrow T$) to an image through the forward process. The model is based on auto-encoders based on UNets (Ho et al., 2020), where it tries to predict the noise layer added through the Markov chain during each iteration ($T \rightarrow 1$). Thus, subtracting the noise image, x_t, from the predicted noise gives x_{t-1} (Figure 1.1) (Ho et al., 2020). The diffusion models are mainly used for generative tasks such as image-generation. Inspired by this process, the author has developed the concept of watermarking using noise and iteratively denoising the added noise to identify whether the image data has been poisoned before being passed into other image tasks/models.

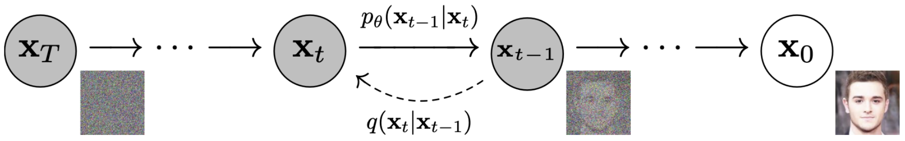

$\mathbf{x}_T$
$\mathbf{x}_t$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$
$\mathbf{x}_{t-1}$
$\mathbf{x}_0$

Figure 1.1: Diffusion Model process (Ho et al., 2020)

Lorenz et al. (2024) stated that when an input image x, goes through the forward process of a diffusion model, it maps it to a noise vector of x_T in the noise space $\mathcal{N}(0, I)$. Then, after applying the backward process of the diffusion model, the predicted noise should be closer to the vector space of $\mathcal{N}(x_T; 0, I)$ samples received. But if the input is altered by adversarial perturbation, the output of the reverse diffusion process will lie on a different manifold compared to benignly transformed samples as visualized in the Figure 1.2 (Lorenz et al., 2024).

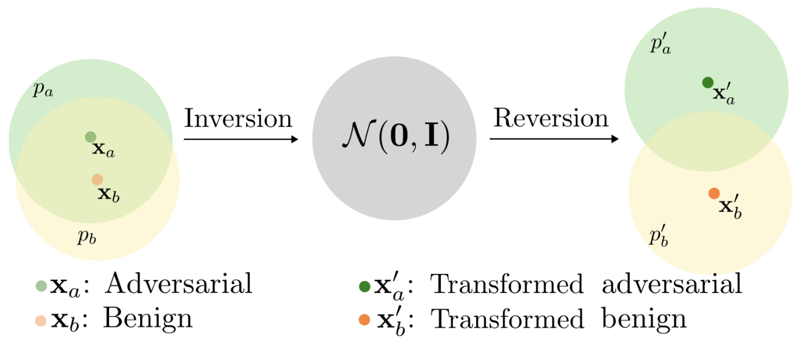

p_a
$\mathbf{x}_a$
$\mathbf{x}_b$
p_b
Inversion
$\mathcal{N}(\mathbf{0},\mathbf{I})$
Reversion
p'_a
$\mathbf{x}'_a$
$\mathbf{x}'_b$
p'_b
● $\mathbf{x}_a$: Adversarial
● $\mathbf{x}_b$: Benign
● $\mathbf{x}'_a$: Transformed adversarial
● $\mathbf{x}'_b$: Transformed benign

Figure 1.2: Lorenz et al. (2024)'s illustration of diffusion model leads to different distributions in benign and adversarial images.

1.4.2 Research Aim

The aim of this research is to design and develop a novel architecture that embeds a set of predefined random noise layers into an image as a watermark and enables the extraction of this noise to classify whether an adversarial attack exists, based on the final denoised output image.

1.4.3 Research Questions

RQ1: How can a diffusion-inspired model be designed and developed to extract predefined noise embedded in images, functioning as a watermark, in order to detect instances of images altered by adversarial noise introduced by attackers?

- **RQ1.1:** What is the relationship between the number of noise layers introduced during the diffusion forward and reverse processes and the quality of the resulting denoised image?
- **RQ1.2:** How do diffusion-based architectures operate, and in what ways can their underlying principles be employed to isolate predefined noise components from adversarially manipulated images?
- **RQ1.3:** How can the diffusion process be optimized or approximated to improve the efficiency of adversarial sample detection without compromising performance?

RQ2: What are the most prevalent adversarial attack strategies in deep learning, and to what extent can they be effectively identified or mitigated through noise-based detection methods?

RQ3: What distinguishes adversarially perturbed images from their clean counterparts in terms of structural or statistical properties?

RQ3.1: What are the underlying mechanisms through which adversarial perturbations deceive DNNs into incorrect classifications?

1.4.4 Research Objectives

RO1: To design and implement a novel model architecture inspired by diffusion processes that can extract predefined noise patterns from images analogous to digital watermarking for the purpose of detecting image poisoning and preventing the use of adversarial samples in downstream deep learning tasks.

RO1.1: To analyze the influence of varying the number of noise layers in the diffusion process on the ability to accurately classify adversarially perturbed images.

RO1.2: To evaluate the suitability of different neural network models for noise extraction and to explore the applicability of diffusion processes in identifying adversarial attacks.

RO1.3: To investigate simplified, layer-wise applications of diffusion models that facilitate efficient and interpretable adversarial sample detection.

RO2: To advance the field of adversarial robustness by proposing an effective and generalizable detection framework capable of defending against diverse attack vectors.

RO3: To examine the nature of adversarial attacks, including how such perturbations are constructed and the specific alterations they introduce to fool DNNs.

1.5 Research Methodology

The research methodology adopted for this study is guided by the ”Research Onion” framework proposed by Saunders et al. (2009). This framework supports the systematic selection of research strategies and methods in alignment with the research objectives.

This study adopts a positivist research philosophy, as it is grounded in observable, measurable phenomena. In this case, the detection of adversarial attacks on images through quantitative evaluation. The research follows a deductive approach, where the development of the proposed system is inspired by established theories, particularly diffusion models, as outlined in the research hypothesis.

A series of experiments were conducted to iteratively develop and evaluate the system. The architecture designed for this research serves as a detection mechanism for adversarially poisoned images, which necessitates its applicability across multiple datasets. To ensure generalizability and reproducibility, publicly available benchmark datasets were used throughout the experiments.

The study is designed with a cross-sectional time horizon, meaning that data were collected and analyzed at specific points in time rather than over an extended period. This approach is suitable for performance evaluation and comparative analysis of the proposed architecture under various attack scenarios and configurations.

Overall, the methodology ensures a rigorous and objective evaluation of the research hypothesis through structured experimentation, enabling clear insights into the effectiveness and generalizability of the proposed solution.

1.6 Significance of the Research

1.6.1 Contribution to the Technology

This work proposes a novel architecture inspired by diffusion models. To the best of the author's knowledge, there has not been any existing architecture similar to the proposed reactive solution that is capable of detecting adversarial noise attacks on images. The proposed method introduces a predefined set of noise layers to an image and then iteratively predicts each noise layer in reverse order. The denoised image is obtained by subtracting the predicted noise layers from the noisy input. This denoised image is then used to detect the presence of adversarial manipulations.

1.6.2 Contribution to the Problem Domain

This research introduces a novel approach to detecting adversarial manipulations performed on images by attackers. Deep neural networks (DNNs) are highly sensitive to adversarial noise, where even minor perturbations can lead to incorrect predictions (Dai et al., 2022; Apostolidis and Papakostas, 2021; Zhang et al., 2021). The proposed solution is capable of identifying such adversarial manipulations before the data is passed downstream to DNNs.

This method is applicable both during training, to filter out poisoned data, and during inference, to reject compromised samples before they influence predictions. Attackers can introduce backdoor attacks into the training data stored on user devices, which are increasingly used to continuously train on-device models such as those in smartphones (Li et al., 2023; Gong et al., 2023). Moreover, attackers can conduct man-in-the-middle attacks during data transmission from a source to a server for downstream DNN tasks, potentially leading to malicious predictions (Moon et al., 2022). The proposed solution has the potential to mitigate such risks by detecting adversarially manipulated inputs in advance.

1.7 Scope of the Project

1.7.1 In-Scope

- The scope is limited to detecting poisoned images (reactive defense) before they are processed by downstream DNNs or image processing tasks.
- The proposed solution is intended for use during either the training or inference stages of downstream DNNs.
- The attack model is limited to scenarios where adversarial attacks are performed prior to the proposed noise watermark prediction network, and it is assumed that the attacker is unaware of the existence of the proposed defense mechanism.

1.7.2 Out-of-Scope

- Environmental factors such as noise introduced during image capture or corruption that occurs during image transmission are not considered in this study.
- Model efficiency in terms of computational light-weightiness or deployment on resource-constrained environments is beyond the scope of this research.
- Attacks targeting the proposed noise prediction network itself, or attacks carried out on the original image after noise removal by the network, are considered out of scope.

1.8 Chapter Conclusion

This chapter established the foundational context for the research by presenting the motivation, research gap, and hypothesis. It laid out the objectives and research questions that guide the direction of the study. The chapter also introduced the research methodology and philosophical approach, providing a roadmap for how the study will be conducted. The subsequent chapters will delve deeper into the theoretical background, implementation, experimentation, and evaluation of the proposed model.

CHAPTER 2: LITERATURE REVIEW

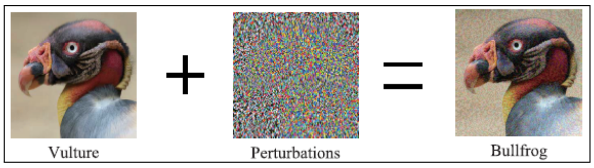
2.1 Chapter Overview

This chapter presents a review of existing literature relevant to the research. It highlights the practicality and growing concern of adversarial attacks, as well as the importance of developing effective defense mechanisms. The literature is used to categorize various types of adversarial attacks and corresponding defense strategies. In addition, the chapter discusses the vulnerabilities of generative models such as VAEs, GANs, and diffusion models to adversarial perturbations, along with the defense techniques proposed in the literature to mitigate such threats.

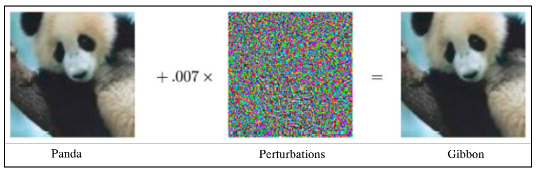
2.2 Adversarial Machine Learning

With the rapid growth of DNN networks, image-processing tasks have become increasingly popular. However, these networks can be fooled by adding small perturbations that are imperceptible to the human eye (Dai et al., 2022). These adversarial examples significantly degrade system performance (Moon et al., 2022), and DNN models in real-world applications may lose users' trust (Apostolidis and Papakostas, 2021).

Attackers use small perturbations, as shown in Figure 2.1, to alter the predictions of DNN methods. The example below demonstrates the classification models' vulnerability to small amounts of noise.



(a) Zhang et al. (2021) example of perturbation added to a model classified correctly as Vulture but after perturbations added, model classifies it as a Bullfrog.



(b) Correctly classified panda with 57.7% before adding 0.007 perturbation while classified as Gibbon with 99.3% confidence after adding perturbations (Apostolidis and Papakostas, 2021)

Figure 2.1: Perturbation Examples: Before adding perturbations, the image was correctly classified as Vulture/Panda. However, after adding perturbations, each model misclassified the image.

These attacks raise serious security concerns for DNN models. Attackers can exploit these vulnerabilities to gain unfair advantages. For instance, Apostolidis and Papakostas (2021) highlights how attackers could perturb medical test reports to fraudulently claim compensation, leading to a waste of critical resources such as patient care, personnel time, medications, and diagnostics. Similarly, Wang et al. (2024) points out that object detection and EEG signal processing can be disrupted by perturbing drone imagery, potentially misidentifying a passenger plane as an attacker plane. Such vulnerabilities also pose significant risks to autonomous vehicles, which rely heavily on DNNs for perception (Chahe et al., 2024). Furthermore, adversarial attacks can be leveraged through MitM exploits to compromise recommendation systems, such as those used in social media auto-tagging applications (Moon et al., 2022).

2.2.1 Adversarial Attack Catergorization

Adversarial attacks can be formally described using the formulae 2.1. Consider the model being targeted by the adversary, denoted as $f(x)$. An adversary is the attacker who performs the attack. The function $f(x) \longrightarrow y$ takes an input x, where $x \in \mathcal{D}$ (the input space), and produces a corresponding output y, where $y \in \mathcal{R}$ (the output space). A perturbed input x^*, where $x^* \notin \mathcal{D}$, is created by adding a small perturbation δ to the original input x, in order to produce an incorrect target $\bar{y}$ from the model. For classification tasks, $\bar{y} \in \mathcal{R}$ and $\bar{y} \neq y$, while for non-classification tasks such as image segmentation or image generation, $y \notin \mathcal{R}$.

$$f(x) \longrightarrow y, \quad x^* = x + \delta \text{ such that } \quad f(x^*) \longrightarrow \bar{y}, \quad \text{where } y \neq \bar{y} \tag{2.1}$$

There are several ways to classify adversarial attacks based on different aspects. The rest of this section highlights the various categories into which adversarial attacks can be classified. A summary of these findings is provided in Table 2.1.

Attack	**Citation**	**Attacker's Knowledge**	**Attack Frequency**	**Attacker Scope**	**Attack's Intention**
L-BFGS	Szegedy et al. (2014)	White box	One shot	Image Specific	Targeted
FGSM	Goodfellow et al. (2015)	White box	One Shot	Image Specific	Targeted
BIM & ILCM	Kurakin et al. (2017)	White box	Iterative	Image Specific	Un-targeted
JSMA	Papernot et al. (2016)	White box	Iterative	Image Specific	Targeted
One-Pixel	Su et al. (2019)	Black box	Iterative	Image Specific	Un-targeted
C & W	Carlini and Wagner (2017)	White box	Iterative	Image Specific	Targeted
DeepFool	Moosavi-Dezfooli et al. (2016)	White box	Iterative	Image Specific	Un-Targeted
Universal Perturbations	Moosavi-Dezfooli et al. (2017)	White box	Iterative	Universal	Un-Targeted
UPSET	Sarkar et al. (2017)	Black box	Iterative	Universal	Targeted
ANGRI	Sarkar et al. (2017)	Black box	Iterative	Image Specific	Targeted
Houdini	Cisse et al. (2017)	Black box	Iterative	Image Specific	Targeted

ATNs	Baluja and Fischer (2018)	White box	Iterative	Image Specific	Targeted

Table 2.1: Summary of Adversarial Attacks

2.2.1.1 Phase of the Attack

The categorization can start from the phase of the victim model. A victim model here refers to the model that is being attacked by an attacker. An attacker can execute attacks either during the training phase or during the inference phase. Training phase attacks are carried out either by perturbing the training samples or by adding perturbed samples into the training data to compromise the training process. Inference attacks, which are known as evasion attacks (Qayyum et al., 2020), involve the attacker providing adversarial samples to mislead the model and achieve falsified results.

2.2.1.2 Attack Specificity

The attacks can be caused by the influence of the attacker or specificity to achieve security violations. The influence of the attacker refers to gaining control over the training data or performing an exploratory attack, which aims to exploit the misclassification of the model without affecting the training process. The adversary specificity can be divided into two sections: targeted and untargeted. Targeted attacks aim to gain an advantage from the victim model to achieve an attacker-specified result, while untargeted attacks target all of the model's outputs to result in falsified outputs. Attackers perform security violations to disrupt the service, integrity, and availability of the model.

2.2.1.3 Attack's Knowledge

The attacks can be categorized based on the information that the adversary knows about the targeted system. This can be divided into three categories: White-Box, Black-Box, and Grey-Box. According to Pitropakis et al. (2019), the categorization is formally denoted as follows. Let the knowledge of the attacker be denoted by $\mathcal{K}$. $\mathcal{K}_1$ indicates that the attacker is aware of the ground truth, and $\mathcal{K}_2$ means the attacker has knowledge of the victim algorithm/model. The categorization can be expressed as: Black-Box Attacks: $\neg\mathcal{K}_1 \bigwedge \neg\mathcal{K}_2$; Grey-Box Attacks: $\mathcal{K}_1 \bigvee \mathcal{K}_2$; White-Box Attacks: $\mathcal{K}_1 \bigwedge \mathcal{K}_2$. The attacker's knowledge $\mathcal{K}$ may refer to the training data, the feature set, the machine learning algorithm including the loss function or the objective function, and the training parameters (Biggio and Roli, 2018). A white-box attack is performed when an attacker has full information about the target model and internal operations regarding defense systems (Lal et al., 2021). The FGSM (Goodfellow et al., 2015) and DeepFool (Moosavi-Dezfooli et al., 2016) attacks fall into this category. A black-box attack means that the attacker is unaware of the system and performs random attacks on it. Momentum Iterative Boosting, Diverse Input, and Transition-Invariant attacks are some examples. When the attacker is not completely aware of the system but has some information, they perform grey-box attacks.

2.2.1.4 Attack Computation

Some attacks are conducted only once, which is known as single-step attacks, while others take iterative steps to produce perturbation x^*. Under this attack frequency classification, iterative attacks are more robust than single-step ones. These attacks try to maximize the model loss. Single-step attacks generate perturbations by calculating the gradient of the model loss only once, adding perturbation in the direction of gradient ascent. FGSM is one such example of a single-step adversarial attack. Multi-step or iterative attacks calculate and add perturbations in each iteration. Basic Iterative Method (BIM) (Kurakin et al., 2017), Carlini and Wagner (C&W) (Carlini and Wagner, 2017), DeepFool, and PGD are examples of iterative attacks (Zhang et al., 2021).

2.2.1.5 Attack Scope

The attacks have a limited scope, where the attacker might be limited to performing unique attacks for each deep learning or machine learning task, whereas some attacks could be generalized to effectively attack all kinds of tasks, independent of the task or the training dataset.

2.2.2 Adversarial Defense Strategies

There are different categorizations proposed by various authors. The following sections have been categorized based on the findings and authors' preferences.

2.2.2.1 Security and Privacy

The findings in Gondim-Ribeiro et al. (2018) have been divided into three categories of security and privacy-safe approaches: hardware-assisted approaches, cryptographic approaches, and differential privacy-based approaches. The hardware-assisted approaches include Trusted-Execution Environments (TEEs) and Dynamic Root of Trust Measurement (DRTM), which offer security and privacy via dedicated hardware modules or operating systems. TEEs lead to white-box scenarios and can invalidate the attacker's hypothesis (Queyrut et al., 2023). Encryption techniques include Homomorphic Encryption (Gondim-Ribeiro et al., 2018) and chaotic-based image encryption (Rezaei et al., 2023), among others. Differential privacy is a technique that protects data by adding noise at a level where the original information becomes unrecoverable. This has been widely used in many domains, including GANs, which are further discussed under countermeasures for adversarial attacks based on generative networks.

2.2.2.2 Level of Defense

Adversarial defense systems have been developed against the above-mentioned adversarial attacks. These can be categorized into two main domains: reactive defense methods and proactive defense methods. Reactive defense methods focus on detecting the attack and taking countermeasures after an attack, while proactive methods aim to improve the resilience of the deep learning network prior to the attack. PixelDefend (Song et al., 2018), ComDefend (Jia et al., 2019), and Defense-GAN (Samangouei et al., 2018) are reactive methods designed to remove perturbations added to the image.

2.2.2.3 Technological Approach

Mahima et al. (2024)'s work has been categorized following Qiu et al. (2019), which consists of three main categories: modifying victim model-based defenses, modifying data-based defenses, and using auxiliary tools against adversarial attacks.

The data-level modifications are applied to the input data or the training data of the deep learning model. Data-level methods include adversarial training (Bai et al., 2021), data compression, transferability blocking, gradient hiding, and random transformation methods (Mahima et al., 2024). In adversarial training, the model is trained with adversarial samples to learn to distinguish between adversarial and original samples.

Model-level defenses include modifying the model's architecture, classifier, and capacity (Apostolidis and Papakostas, 2021). This includes techniques such as feature squeezing (Xu et al., 2018) and defensive distillation. Adversarial training is far more costly than feature squeezing (Xu et al., 2018), but these methods are limited to specific models and lack transferability.

Auxiliary tools are more modular, where these tools are separate components of the deep learning model. These tools could be another model that performs reconstruction and purification, as explained in the countermeasures for adversarial attacks based on generative networks. However, this increases the weight of the deep learning model pipeline due to the addition of a defensive module beforehand.

2.2.2.4 *Defense Scope*

Most of these defense methods are attack-specific. Model-specific defenses are primarily based on adversarial training, gradient regularization, and input transformation-based methods. Adversarial training methods are tailored to specific attacks, which re-train the network using adversarial samples; hence, they are known as attack-specific defenses. Several studies have been conducted in the domain of model-agnostic strategies, but these often lack quality for long-term use (Zhang et al., 2021). Model-agnostic strategies include Super-Resolution defense (Mustafa et al., 2020), JPEG compression (Dziugaite et al., 2016), High-level representation Guided Denoiser (HGD) (Liao et al., 2018), PixelDefend (Song et al., 2018), Defense-GAN (Samangouei et al., 2018), and random resizing and padding (Xie et al., 2018).

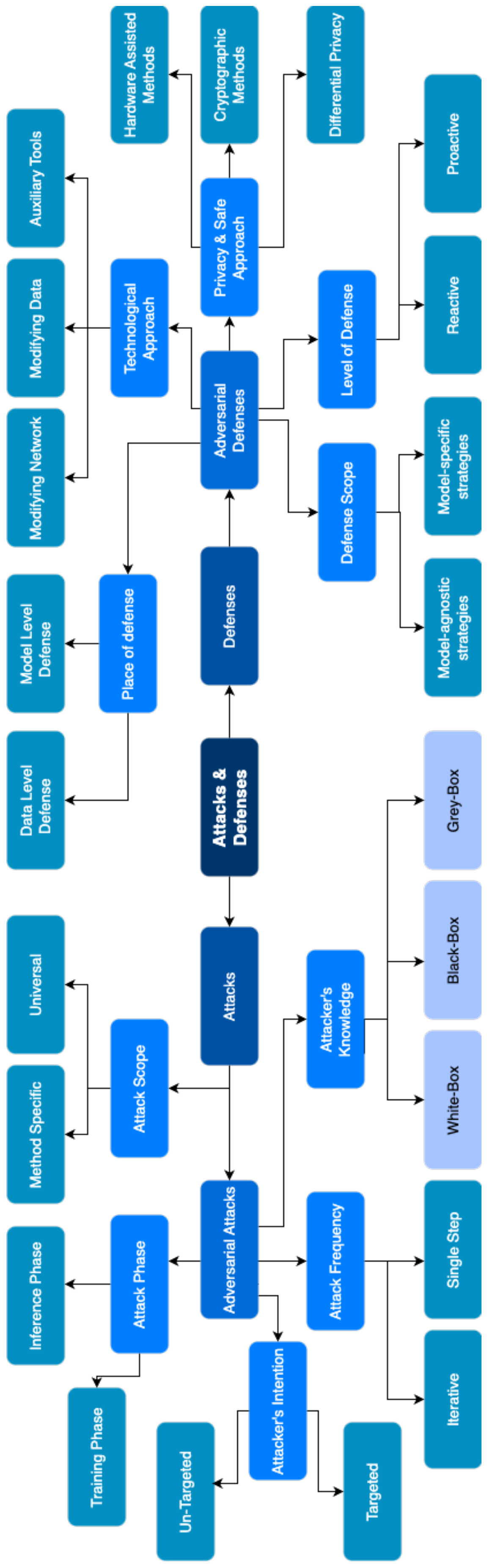


Figure 2.2: Research Findings (Self-Composed)

2.3 Generative Networks in Adversarial Machine Learning

2.3.1 Generative Network Summary

2.3.1.1 Variational Auto Encoders

The VAEs was first introduced by Kingma and Welling (2022). The goal of a VAEs is to reconstruct or generate an output similar to training samples. Its architecture consists of an encoder, which transforms the input data into a probability distribution within the latent space. This latent space is then transformed back into a specific target data space that the model aims to achieve using a decoder. During the training process, both the encoder and decoder aim to minimize the reconstruction loss to achieve a targeted meaningful data distribution or representation.

VAEs can be used as an autoencoder as well as a generative model (Gondim-Ribeiro et al., 2018). The applications of VAEss span various domains such as Natural Language Processing (NLP) and image processing tasks, including text embedding, image or signal compression (Berahmand et al., 2024), generating images, and image classification (Imran and Terzopoulos, 2021).

2.3.1.2 Generative Adversarial Networks

GAN networks were initially introduced by Goodfellow et al. (2014) and consist of two network architectures: the generator and the discriminator.

The generator model aims to generate synthetic samples starting from random noise. The discriminator is provided with both real samples and the synthetic samples from the generator to distinguish between them. In this two-player network, the generator tries to fool the discriminator by generating realistic synthetic samples. GANs have expanded into various architectures, such as Conditional GAN (Mirza and Osindero, 2014), Deep Convolutional GAN, and even more complex architectures like Cycle-GAN (Zhu et al., 2020).

The applications of GANs extend to real-world scenarios such as image-to-image translation for improving image quality in mobile phones (Zhu et al., 2020; Walgama and Mahima, 2024), image super-resolution (Zhang et al., 2022), and data augmentation (Bissoto et al., 2021).

2.3.1.3 Diffusion Models

In recent years, diffusion models have become more popular due to their enormous capability to generate quality samples surpassing GANs and VAEs. The state of research in diffusion models expands across different domains such as image processing, NLP, temporal data modeling, multi-modal modeling, robust machine learning, and interdisciplinary applications in fields such as computational chemistry and medical image reconstruction (Yang et al., 2023).

Diffusion models consist of a forward and a backward process, where the forward process continues to iteratively add random noise to the initial image (denoted as T; typically T is around 1000 to 2000+ iterations) until the image is completely filled with noise. The reverse process then tries to remove the noise added in the forward process and attempts to generate an image. Based on these diffusion processes, state-of-the-art diffusion models can be categorized into discrete-time diffusion models and continuous-time diffusion models.

2.3.2 Denoising Diffusion Probabilistic Model Process

The DDPM formally the forward diffusion process generates a sequence of distribution denoted as $x_1, x_2, ...x_T$ using transition kernel $q(x_t|x_{t-1})$ where this Markov chained distribution conditioned on the initial image x_0.

$$q(\mathbf{x}_1, \ldots, \mathbf{x}_T \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1}). \tag{2.2}$$

The transition kernel with the prior distribution of $q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ is defined as,

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right), \tag{2.3}$$

where $t \in \{0, 1, \ldots, T\}$ and the β is a hyperparameter such that $\beta_t \in (0, 1)$. Simplifying the equation by substituting $\alpha_t := 1 - \beta_t$ and $\bar{\alpha_t} := \prod_{s=0}^{t} \alpha_x$.

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\bar{\alpha_t}}\,\mathbf{x}_{t-1}, (1-\bar{\alpha_t})\mathbf{I}\right), \tag{2.4}$$

Can derive sample of x_t by obtaining the x_0 with the Gaussian vector $\epsilon\ \mathcal{N}(0, I)$ and applying the transformation,

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \tag{2.5}$$

The reverse diffusion process of DDPM iteratively predicts and removes the noise added during the forward process mentioned above by running a learnable chained Markov model in the reverse time direction, i.e. T to 1. The forward process prior distribution was $q(x_T) \approx \mathcal{N}(x_T; 0, I)$ which would be similar in the reverse process as $p(x_T) = \mathcal{N}(x_T; 0, I)$. The reverse process kernel, $p_\theta(x_{t-1}|x_t)$ is defined as follows.

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(x_t, t)\right), \tag{2.6}$$

where θ is parameters of the models to the mean, $\mu_\theta(\mathbf{x}_t, t)$ and the variance, $\Sigma_\theta(x_t, t)$. The reverse transition kernel aims to achieve x_0 by sampling through the noise distribution till $t = 1$. This forward Markov chain process in Equation 2.2, $q(x_0, x_1, \ldots, x_T) := q(x_T) \prod_{t=1}^{T} q(x_0|x_T)$ been trying to closely approximate using the reverse Markov process of $p_\theta(x_0, x_1, \ldots, x_T) := p(x_T) \prod_{t=1}^{T} p_\theta(x_0|x_T)$. Minimization can be achieved using Kullback-Leibler (KL) divergence as follows,

$$\begin{aligned}
&\mathrm{KL}(q(x_0, x_1, \cdots, x_T) \parallel p_\theta(x_0, x_1, \cdots, x_T)) && (2.7)\\
&\overset{(i)}{=} -\mathbb{E}_{q(x_0,x_1,\cdots,x_T)}\left[\log p_\theta(x_0, x_1, \cdots, x_T)\right] + \text{const} && (2.8)\\
&\overset{(ii)}{=} \mathbb{E}_{q(x_0,x_1,\cdots,x_T)}\left[-\log p(x_T) - \sum_{t=1}^{T} \log \frac{p_\theta(x_{t-1} \mid x_t)}{q(x_t \mid x_{t-1})}\right] + \text{const} && (2.9)\\
&\overset{(iii)}{\geq} \mathbb{E}\left[-\log p_\theta(x_0)\right] + \text{const.} && (2.10)
\end{aligned}$$

Here, (i) is direct applying KL divergences while the (ii) is products of two distributions of $q(x_0, x_1, \ldots, x_T)$ and $p_\theta(x_0, x_1, \ldots, x_T)$. The (iii) is a derivation from Jensen's inequality. The first term of the Equation (8), $\mathbb{E}_{q(x_0,x_1,\cdots,x_T)}\left[-\log p(x_T) - \sum_{t=1}^{T} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right]$ is the variational lower bound (VLB). Yang et al. (2023) have used *const*, constant independent of θ model parametrization which will not affect the optimizations. The aim here is to maximize the VLB which eventually minimizes the negative VLB during the DDPM training. Yang et al. (2023) have stated that this is unchallenging to optimize as the sum of independent terms which could be estimated by Monte Carlo sampling and optimized using stochastic optimization.

2.3.3 Adversarial Attacks Based on Generative Networks

Adversarial attacks are most common in classifiers, but attacks on autoencoders are a much less explored area, possibly due to the complexity of the encoder-decoder architecture (Gondim-Ribeiro et al., 2018). The first attacks on VAEs were introduced by Tabacof et al. (2016). This section primarily focuses on attacks generated using generative models such as VAEs and GANs.

VAEs and GAN are generative models that raise concerns about generating adversarially perturbed samples. This was highlighted by Zhang et al. (2023), who introduced three major threats posed by GANs:

- **Data Generation and Manipulation:** GANs are commonly used for the generation of images that can be used to create synthetic samples aligned with the data distribution.
- **Detection Evasion:** GAN-generated samples can be crafted in such a way that classifier networks misclassify the samples, mislead the victim model, or are ignored by the detection system.
- **Privacy Breach:** Data manipulation and replication can be performed using GANs. For example, user-sensitive data transferred in Federated Learning (FL)-like architectures may be replicated using GANs. In FL, a single model is trained on multiple clients and the trained weights are transferred to the server, where a GAN model could interpret the weights transferred between the server and clients (Maliakel et al., 2024).

Sun et al. (2024) has proven the risks of GAN-based image fusion by demonstrating attacks on them. Sun et al. (2024) has proposed two methods. First, proposed a subtle attack on an input image which could lead the GAN-based image fusion model to yield the same results for all inputs; secondly, proposed adversarial patches which could cause the image fusion GAN-based model to produce meaningless outputs. Their patch attacks are universal and could even be transferable to other datasets. Real-world applications such as re-identification (ReID) of humans or vehicles require more robust methods. This has been demonstrated by Zhao et al. (2022), where a GAN-based patch attack was applied to images for ReID. The attacks were performed using the proposed Adversarial Patch GAN (AP-GAN), which is trained on a small amount of data, addressing the issue of the lack of retrieval of well-defined labels from datasets. Moreover, minor perturbations do not cause significant changes in the ranking of results. Zhao et al. (2022) classified their work as a semi-white-box attack. Similar work, UAA-GAN, has been carried out by Zhao et al. (2019) on the same domain of image retrieval, except for vehicle ReID, but extending the work with face search. Both architectures, unsupervised GANs, are similar, but Zhao et al. (2022) focused on mask-provided attacks where the attack is performed on the masked region, while Zhao et al. (2019) generates perturbations without a mask on the object region of the photo. The attacks are invisible to the human eye, even the UAA-GAN attack performed on the object of the image rather than the background, which is not the focus.

A couple of attacks have been proposed against intrusion detection systems (IDS) based on GANs (Hu and Tan, 2017; Lin et al., 2022; Usama et al., 2019; Zhao et al., 2021).

MalGAN generates malware to bypass black-box machine learning-based IDS (Hu and Tan, 2017).

2.3.4 Defense Methods Developed using Generative Networks

2.3.4.1 Adversarial Purification

Defense-GAN is a robust defense mechanism proposed against adversarial attacks which does not require knowledge of the adversary or how the attack was performed to purify the image. Defense-GAN is trained on unperturbed images and is more reliable since it does not modify the main perception network. Further studies (Qin et al., 2024; Kang et al., 2021) are a few more examples of GAN-based adversarial attacked image purification. However, studies Nie et al. (2022) have identified that using GAN for the purification process can lead to mode collapse, while energy-based models can lead to low-quality purification and a lack of randomness. Supporting this, Shi et al. (2021) used Pixel-Defend, which is a purification-based network training approach (Song et al., 2018), instead of a GAN-based purification approach.

The concept proposed by Santhanam and Grnarova (2018) works similarly to the theory based on the Lorenz et al. (2024) work, where Santhanam and Grnarova (2018) stated that adversarial samples lie outside of the data manifold. In simple terms, attacked images have a different distribution compared to unattacked images, and while this is not identifiable by the human eye, it can be identified using a discriminator of a GAN. Santhanam and Grnarova (2018) used a discriminator to score the samples without providing any adversarial samples during training. Observations show that adversarial samples score lower, and once an attack is detected, they used the generator of the GAN for purification and observed a lower score. Extensive experiments have been conducted with 5 different attacks on 3 different datasets. Zhang et al. (2023) stated that these kinds of techniques are ”counter GAN-based attacks using another GAN - like cures like”. A Cycle-GAN can be used to demonstrate this, as it consists of two generators. CycleAdvGAN is such an example, where both adversarial samples and clear samples for sample images are generated using the translation process of the CycleGAN Jiang et al. (2020).

Inspired by Samangouei et al. (2018)'s DefenseGAN and Kim et al. (2017)'s DiscoGAN, Laykaviriyakul and Phaisangittisagul (2023) proposed Collaborative Defense-GAN, which is very similar to CycleAdvGAN. This architecture consists of an attacker generator that generates a noise image instead of a perturbed image. The generated noise is combined with the original image to create a perturbed image, which is then fed into another generator for the purification process. This collaborative network includes two discriminators during training, similar to CycleAdvGAN.

A conditional paired image GAN model has been proposed by Yu et al. (2020). The conditional Pix2Pix-based GAN is able to purify adversarial samples. This network is trained with a PatchGAN as a discriminator for better localized results (Yu et al., 2020).

2.3.4.2 Adversarial Training

Hashemi and Mozaffari (2019) proposed an adversarial training method that could make DNNs more robust against adversarial evasion attacks by training with adversarial samples generated by the proposed Noise-GAN network. The study by Hashemi and Mozaffari (2019) used a multi-class discriminator to generate adversarial noise, which is combined with the input of the DNN network for adversarial sample generation. A counter-argument was found in Zhang et al. (2023)'s work, which mentioned that only the discriminator has access to real data, but not the generator; hence, there is a natural cover for privacy in GANs. The evaluations of Hashemi and Mozaffari (2019)'s work were done on the MNIST (Deng, 2012) and CIFAR-10 (Krizhevsky et al., n.d.) datasets.

2.3.4.3 Differential Privacy

Differential privacy has been used in almost all domains, such as FL (Xin et al., 2020) and GANs, where the privacy of the data is hidden by adding noise to the data. However, this has shown a lack of improvement in GAN-based attacks (Zhang et al., 2023). The importance of differential privacy application on GAN was highlighted in Xu et al. (2019)'s work, mentioning the weak side of GANs: information leakage. In contrast, several differential privacy-based GANs have been proposed, but mostly in domains outside of image processing (Jiang et al., 2022; Yoon et al., 2019).

2.3.5 Diffusion Models against Adversarial Attacks

There have been a couple of studies on the use of diffusion models against adversarial attacks. In the following section, categorize diffusion model applications in adversarial defense into two main categories: adversarial purification, which involves using diffusion models to purify or recover the attacked image (Nie et al., 2022), and diffusion model-based classification, which is used to determine whether an image has been attacked. These categories fall under the reactive defense methods in the taxonomy shown in Figure 2.2. The diffusion model for adversarial defense is a more robust method compared to approaches like adversarial training (Blau et al., 2022; Yu et al., 2024).

There has been research conducted using diffusion models for adversarial image purification, which outperform GAN models (Nie et al., 2022; Ho et al., 2020). The study by Nie et al. (2022) proposed a continuous-time diffusion network based on Stochastic Differential Equations (SDE) as an adversarial purification method. Notably, this purification model only requires a few steps x_{t^*} rather than fully reaching x_T, where $t^* < T$, as it only requires noise that is not high enough to destroy the label semantics of the images while being sufficient to remove adversarial perturbations. However, Xiao et al. (2022) stated that DiffPure is not robust due to the careful consideration needed regarding the amount of noise to add to the image.

The study by Lorenz et al. (2024) highlights and demonstrates an insightful concept where a perturbed image going through the forward and reverse diffusion processes would have a distinct distribution compared to a non-perturbed image undergoing the same diffusion processes. Further explained, the output distributions of attacked and non-attacked images are clearly separable. In the aforementioned studies, CIFAR10 (Krizhevsky et al., n.d.) and ImageNet (Deng et al., 2009) are recognized as widely used datasets.

The study by Nie et al. (2022) is the closest research to the proposed solution. Unlike diffusion networks, which require substantial computational resources, the proposed solution is comparatively lightweight. The study itself acknowledges its limitations, particularly its unsuitability for real-time applications due to the diffusion timestep, which leads to lengthy purification processes. In light of these limitations, the author presents ”Diffusion-Inspired Image Watermarking Against Adversarial Attacks,” which is inspired by the forward and reverse processes of diffusion rather than entirely applying a diffusion model as it is.

2.4 Chapter Conclusion

This chapter reviewed existing literature on adversarial attacks and their corresponding defense mechanisms, emphasizing the limitations of current models that remain vulnerable to such threats. The discussion highlighted the gap in leveraging generative models, particularly diffusion models for adversarial defense. Based on the identified shortcomings in existing approaches, the novelty of the proposed solution was outlined, wherein the diffusion process is utilized as a mechanism for detecting adversarially perturbed inputs.

CHAPTER 3: METHODOLOGY

3.1 Chapter Overview

This chapter provides an overview of the proposed system's design and architecture. It outlines the overall structure and flow of the process, detailing how each component fits within the broader framework. The chapter also describes the models utilized in the implementation, emphasizing their roles and interactions within the system.

3.2 System Flow

Based on the research hypothesis, the following architecture in Figure 3.1 was derived. The proposed solution can be trained using a dataset in which the system adds a random set of noise to an image in a sequence through the forward process ($x_1 \rightarrow x_T$ where $T > 0$). The reverse process tries to predict each added noise layer in the reverse process ($x_T \rightarrow x_1$).

During the inference stage, the proposed solution tries to predict the noise added at the initial stage. If an attacker poisons the data at any point in the prediction pipeline, the proposed solution aims to classify whether it is adversarially corrupted or not using three main steps: 1) iteratively add random noise to the image, 2) denoise the added noise iteratively, and 3) detect and reject the adversarially altered samples using the denoised image.

Here, the author assumed that the adversary has no knowledge about the proposed defense solution and is only able to make the attack before it. This is reasonable, as in a well-secured software system, an attacker has a higher probability of altering the raw sensor inputs rather than the ones taken into the main prediction pipeline. Hence, the proposed solution can be used to identify poisoned data before passing it into the DNN networks, either during training or during inference, to avoid incorrect predictions.

3.3 System Architecture

Based on Lorenz et al. (2024) work, as mentioned in the section 1.4.1, the following architecture has been proposed.

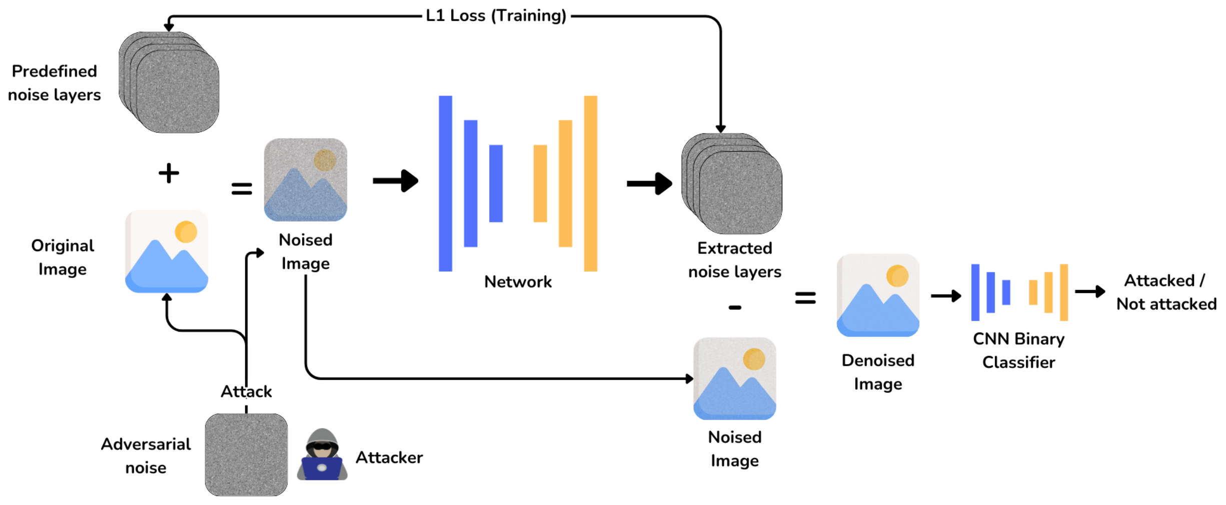


Figure 3.1: Proposed System Architecture (Self-composed)

3.4 Models

Several experiments with different model architectures and designs were conducted by the author. Specifically, the diffusion model and the classification network were experimented with using different model architectures. The summary of the experimental model architectures is as follows, and detailed explanations are provided in the subsequent section:

1. **UNet + ResNet50 -** UNet model as a noise extractor with ResNet50 as the classifier
2. **UNet + ResNet50 (Merged Noise Classifier) -** UNet model as a noise extractor with ResNet50 as the classifier. Rather training the classifier with a denoised image, the classifier trains with the extracted noise layers.
3. **DDPM + ResNet50 -** DDPM as a noise extractor with ResNet50 as the classifier
4. **DDIM + ResNet50 -** Denoising Diffusion Implicit Models (DDIM) as a noise extractor with a simple CNN network
5. **DDIM + CNN -** DDIM as a noise extractor with a simple CNN network

3.4.1 UNet + ResNet50

The author has used a UNet architecture without a noise scheduler to extract the noise layers. This UNet is inspired by the study Saharia et al. (2023). As a replacement for the scheduler, the author created a custom time embedding by adding another channel to the image, which consists of the time step information. Thus, the image has four channels, with the shape `Tensor(4, width, height)`, where the first channel is the time embedding, and the remaining three channels are the color channels of the image. The extracted image from the UNet is input into the ResNet50 classifier network to determine whether an image has been attacked or not, based on the concept in Lorenz et al. (2024).

The author conducted two experiments under this architecture using two different datasets. The first experiment was conducted using the CIFAR-10 dataset, which contains 32 × 32 images, and the second experiment was conducted using the Places365 dataset. During the experiments with the CIFAR-10 dataset, it was observed that the images lacked sufficient resolution to determine whether the noise had been extracted correctly by the human eye. Thus, the first experiment with the CIFAR-10 dataset was limited to training the UNet model. In contrast, the second experiment trained both the UNet noise extractor and the ResNet binary classifier network using the Places365 dataset.

3.4.2 UNet + ResNet50 (Merged Noise Classifier)

The experiment was conducted using the same networks used in the previous setup: UNet as a denoiser to extract the image from the previously added iterative noise, and ResNet50 as the decision maker to identify whether an attack exists or not. However, as shown in the architecture 3.2, the classification network was modified to take the merged denoised noise layers into a single noise layer rather than taking the denoised image.

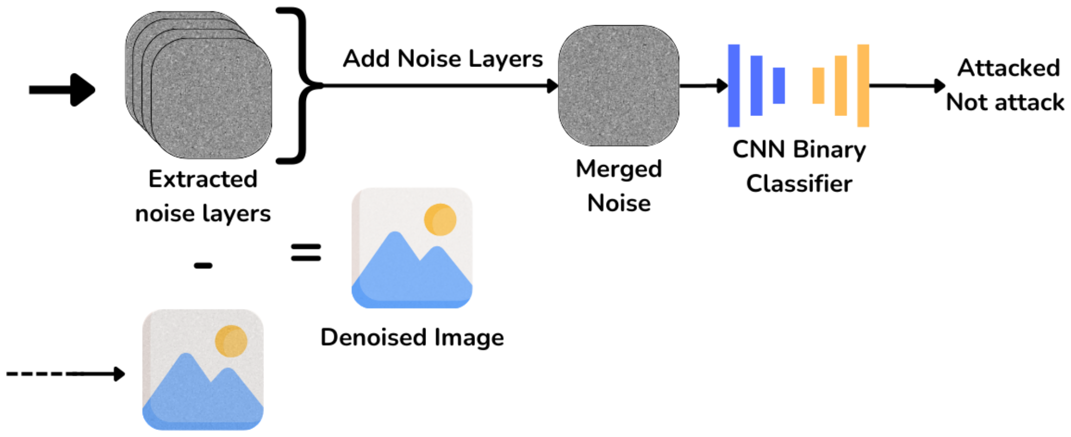


Figure 3.2: Modified Architecture (Self-composed)

The hypothesis behind this architecture is that if a noise attack occurs, the noise extracted by the UNet would exhibit a different distribution between attacked and non-attacked layers.

3.4.3 DDPM + ResNet50

The experiment was carried out with the existing pre-trained diffusion model and noise scheduler introduced in the study Saharia et al. (2023). This model is used for super-resolution, enhancing images from a size of 64×64 to 512×512.

The CelebA-HQ dataset was preprocessed by resizing the 1024 images available in the dataset. The images were first resized to 64×64 and then upscaled to 512×512 using bicubic interpolation. This bicubic-interpolated image was then passed as a conditioned image to the diffusion model to generate a 512×512 output. The generated 512×512 image was then input into the classifier. Before passing the input image to the diffusion model, FGSM attacks were performed to train the classifier network.

Since the model is a DDPM model, it took an average of 2 minutes and 50 seconds to generate a single sample. Generating training samples for the classifier network would take more than 11 days on a RTX-3080 10GB GPU, making the entire training process highly resource-intensive and time-consuming.

3.4.4 DDIM + ResNet50

This experiment is similar to the previous one, DDPM + ResNet50, where the DDPM based inference sampling process used in Saharia et al. (2023) was updated to a DDIM sampling process by analyzing the code-level changes in Guo et al. (2023). The purpose of this modification was to accelerate the denoising process.

The code was modified to utilize the pre-trained weights of the DDPM model, since training a diffusion model requires high computational resources. The ResNet50 architecture was kept the same as before, and the attacks were also performed using the same configuration as above.

This approach reduced the sample generation time to less than a minute, as the number of time steps required to remove noise was reduced from 2000 (the default for DDPM) to 1000 in this DDIM setup.

3.4.5 DDIM + CNN

In this experiment, the ResNet50 based classification network was changed to a simple CNN classification network and combined with the previous DDIM model in order to further reduce the computational complexity of the proposed solution.

3.5 Chapter Conclusion

This chapter presents the proposed system flow, the hypothesized architectural structure of it, including the different model configurations used.

CHAPTER 4: EXPERIMENTAL SETUP

4.1 Chapter Overview

The following chapter contains the datasets used in the experiments, as well as the types of adversarial attacks employed for both training and evaluation purposes, with the experiment setup, including both the hardware and the software, are discussed. In addition, the evaluation metrics used to assess the performance of the models are outlined.

4.2 Datasets

The author has conducted the experiments with the following datasets.

Dataset	Resolution	Images	Description	Citation	Experiment
CIFAR10	32×32	60k	It contains 10 classes of images, including animals, vehicles, and various items. The training dataset consists of 50,000 images, while the test dataset contains 10,000 images.	Krizhevsky et al. (n.d.)	UNet + ResNet50
Places365	256×256	1.8 million	It contains images of places categorized into 365 scenes. The validation set includes 50 images per category, while the test set contains 900 images per category.	Zhou et al. (2017)	
CelebAHQ	1024×1024	30k	This dataset contains a high-resolution version of the CelebA dataset (Liu et al., 2015). It consists of human face images and is commonly used for image generation.	Karras et al. (2018)	DDPM/ DDIM + ResNet50, DDIM + CNN

Table 4.1: Dataset Details

The reason for choosing the CIFAR-10 dataset was that it is a commonly used dataset for generative and classification tasks. The initial experiments were conducted with the CIFAR-10 dataset, but due to its low resolution, the author moved to a higher-resolution dataset like Places365, which could be implemented with a small classifier network serving as the victim model.

Since the author used a pre-trained DDPM model from Saharia et al. (2023), there was no choice but to use the same dataset as Saharia et al. (2023). The author attempted to feed the DDPM model with Places365 or another dataset, but the pre-trained DDPM model lacked the ability to generate images for unseen distributions, such as scenes or locations from Places365. Generated samples from DDPM are included in Figure 4.2.

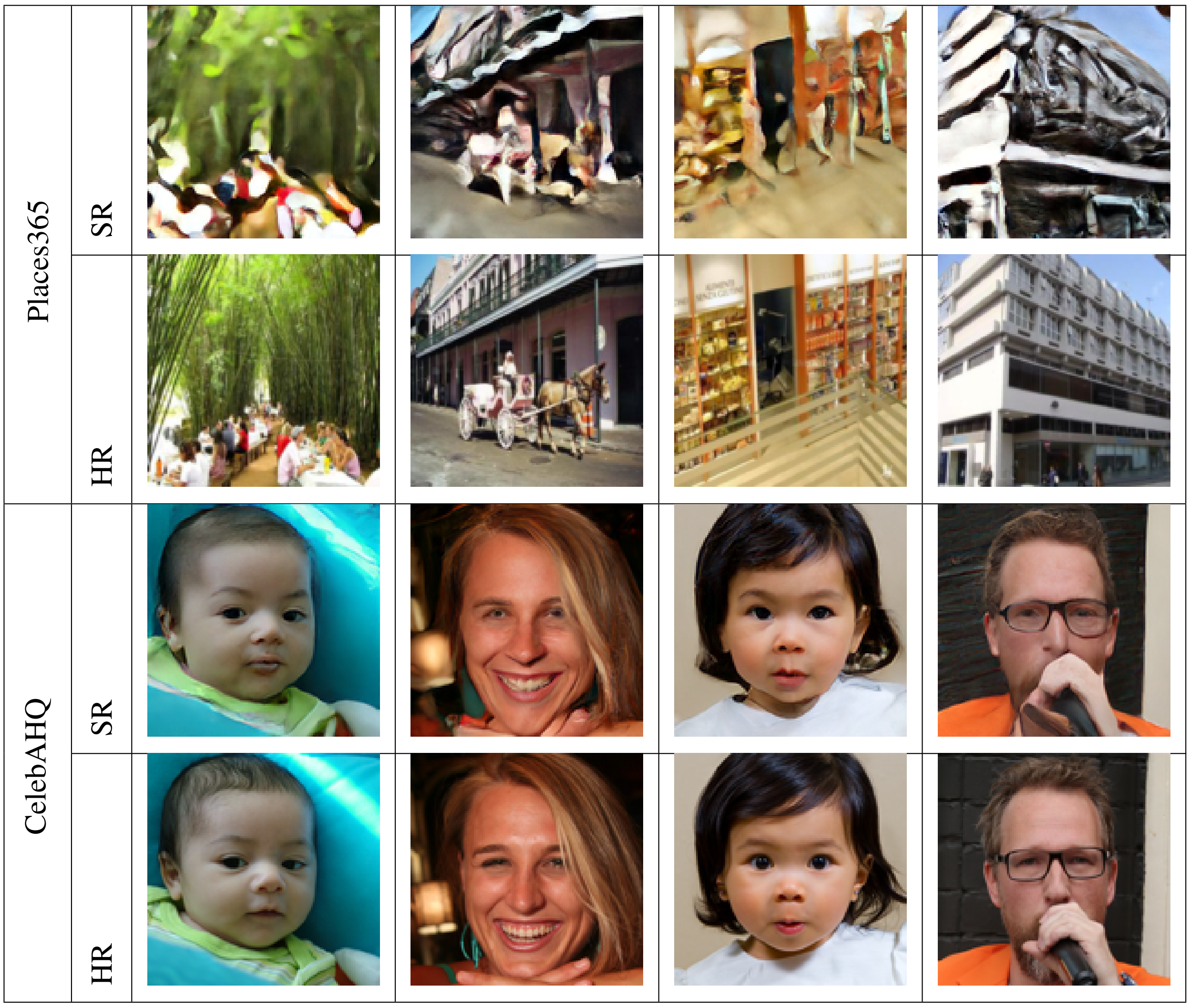


Table 4.2: DDPM generated samples from CIFAR10 and Places365 dataset. SR refers to the Super Resolution image generated from the DDPM and HR refers to High Resolution image which is the ground truth.

4.3 Attacks Configuration

The models were primarily trained using the FGSM with an ϵ value of 0.3. This value was selected as it induces significant perturbations, thereby increasing the vulnerability of most DNN. Although FGSM is a single-step attack, it is considered a foundational method and serves as the basis for more advanced iterative attacks such as the PGD attack.

The adversarial attacks employed to evaluate the robustness of the proposed solution are listed below.

#	**Attack**	*standard deviation*			
1	Gaussian Noise	0.1			
		ϵ	α	*steps*	
2	PGD	0.05	$\frac{2}{255}$	10	
3	PGD	0.15			
4	PGD	0.30			
		ϵ			
5	FGSM	0.05			
6	FGSM	0.15			
7	FGSM	0.30			
		c	κ	lr	*steps*
8	CW	1	0	0.01	50

Table 4.3: Attack evaluation setup

4.4 Evaluation Metrics

The evaluation of the proposed system can be done using reference image quality assessment methods as the system adds noise to the original image and makes it perturbed. Thus, as in Figure 3.1, the model will extract the added noise to obtain a denoised image and cross-validate with the original image. The layers would be cross-validated against the model predicted noise layers during training. This could be achieved using MAE. Other metrics have been used to prove the hypothesis. The final classifier is evaluated using accuracy.

4.4.1 Accuracy

Accuracy is a way measure a classification model how accurately a model can predict results. This can be use for classification tasks using the following formulae.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ predictions} \tag{4.1}$$

4.4.2 MAE

The MAE is one of the ways to measure the absolute difference between intensities of the pixels between the two images, i.e. added noise and the predicted noise from the model. For each noise layer, the model calculates MAE for a single image. The final MAE would be averaged. The MAE is calculated as below,

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n} \tag{4.2}$$

4.4.3 SSIM

The SSIM is a reference image quality assessment method that compromises on structural information of the considered pair of images which includes luminance masking, contrast masking etc. The structural comparison is done using the inter or spatially closed pixels. In contrast, contrast masking involves distortions that are also less apparent within the texture of the image. Luminance masking refers to a situation in which the distorted portions of an image are less noticeable around the edges (Sara et al., 2019). SSIM is a value between 0 and 1. The higher the value represents the quality of the image. This could be used to measure the quality are the extracted images after deducting the predicted noise. SSIM value can be calculated as follow,

$$\text{SSIM} = [l(x, y)]^{\alpha}.[c(x, y)]^{\beta}.[s(x, y)]^{\gamma}. \tag{4.3}$$

where l: luminance, c: contrast, s: structure while α, β, γ are parameters greater than 0.

4.4.4 PSNR

PSNR is a paired image quality measurement whereas to uses maximum possible value (power) of a signal and the distorting noise (Sara et al., 2019). The quality of the image is greater as the PSNR values are higher (Nadipally, 2019) but that does not guarantee the best visual results (Ignatov et al., 2018). Similar to the SSIM, this could use to compare the extracted image and the original image. PSNR value can be calculated as follows,

$$\text{PSNR} = 10 \log_{10} \left(\frac{\text{peakval}^2}{\text{MSE}} \right). \tag{4.4}$$

where $peakval$ refers to the Peak Value which is the maximal in the image data.

4.5 Training Setup

This section would give the reader information to replicate results mentioning the software requirements for the implementations as well as the information about the author's machine hardware which is used to train the model.

4.5.1 Software Requirements

The following software with listed version has been used to implement the proposed solution.

Software	**Version**
Python	3.8
Conda	24.1.2
CUDA	12.2

Table 4.4: Software Requirements

The implementation of the proposed solution was done using Python due to the availability of a wide range of libraries. Also, the author's expertise dives into Python.

4.5.2 Hardware Setup

The model was trained on an Ubuntu desktop machine using RTX-3080 10 GB GPU. The operating system was 24.04 LTS. Additionally to that, the processor was 12 core, i5 processor with 16 GB memory.

4.6 Chapter Conclusion

The experiments outlined in this chapter are structurally aligned with the proposed architecture; however, variations in the classifier architecture and its input representations were introduced to explore different configurations aimed at achieving improved performance. The rationale behind the selection of datasets has been clearly articulated, ensuring relevance to the research objectives. Additionally, detailed descriptions of attack configurations, software frameworks, and hardware specifications have been provided to support reproducibility and enable validation of the results by future researchers.

CHAPTER 5: EXPERIMENTS AND TRAINING

5.1 Chapter Overview

Building upon the architectural foundation established in the previous chapter, this section details the training procedures of the proposed solution. It elaborates on the various model training processes. The chapter provides insights into the specific training strategies, parameters, and methodologies adopted to evaluate the effectiveness of each model variant.

5.2 UNet: CIFAR10 Dataset Training

The first experiments were conducted using the CIFAR-10 dataset. The image size was much smaller, making it difficult for a human to identify the images. The following are log samples taken from Weights and Biases.

The training loss, i.e., noise prediction loss, is shown in Figure 5.1 over 400,000+ iterations.

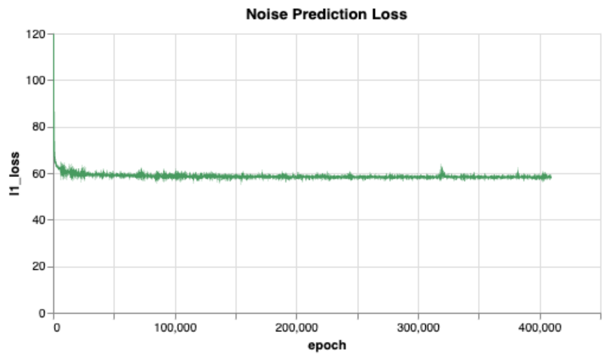
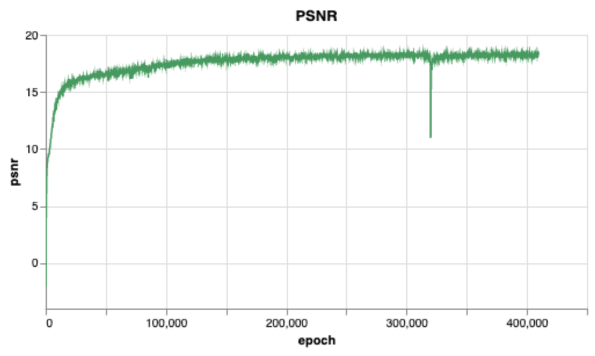


Figure 5.1: CIFAR10 Training-Training loss (L2 loss)

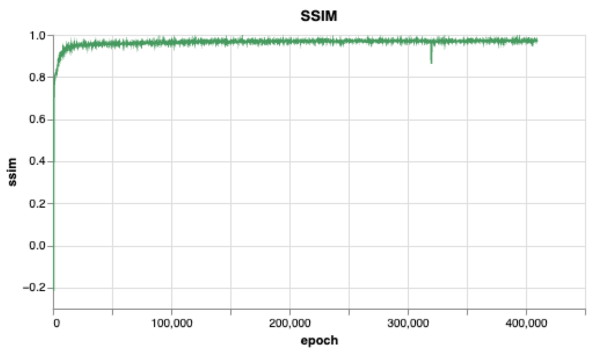
The following graphs were plotted using the PSNR and SSIM values, considering the noise-added image at the t^{th} iteration during the forward process and the recovered image from the model at the reverse t^{th} iteration. For a single image, the sum of these values was taken and averaged over 10 noise layers, as explained in the evaluation chapter.

The following figures 5.3, 5.4, and 5.5 represent the backward process; the noise extraction process from the original image. The author added 10 random noise image layers, which have been removed using the UNet model, as visible in the following time sequence from $t = 10$, which is the image with all noise layers added, to $t = 0$, which is the fully noise-extracted image. From $t = 9$ to $t = 0$, the image is recursively input into the model.

Since the dataset samples are in a lower resolution of 32×32, the images are difficult to compare with the original image since the details are very few. It is very difficult for the



(a) PSNR Value variation



(b) SSIM Value variation

Figure 5.2: PSNR and SSIM Value variation during training

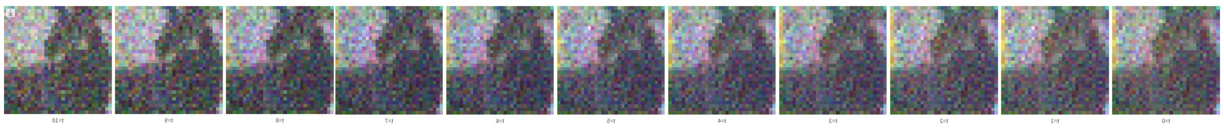


Figure 5.3: CIFAR10 Training sample during 1^{st} *iteration*

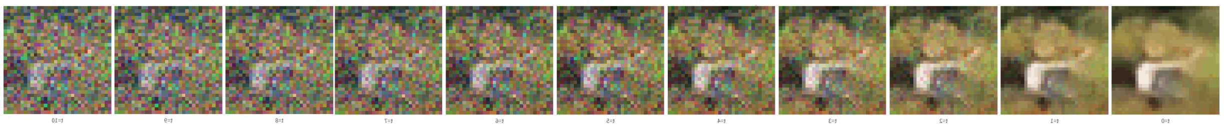


Figure 5.4: CIFAR10 Training sample during 200000^{th} *iteration*

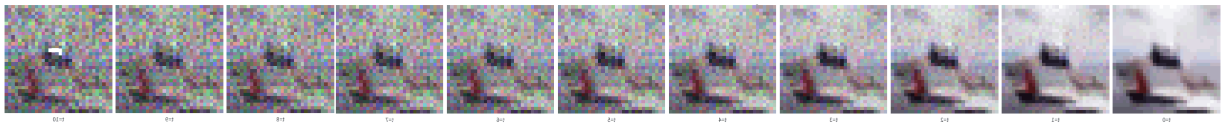


Figure 5.5: CIFAR10 Training sample during 400000^{th} *iteration*

human eye to identify the image object (Figure 5.5). Testing under CIFAR10 has been skipped. Thus, the experiment was extended with the Places365 dataset to observe the extraction of noise in higher-quality images.

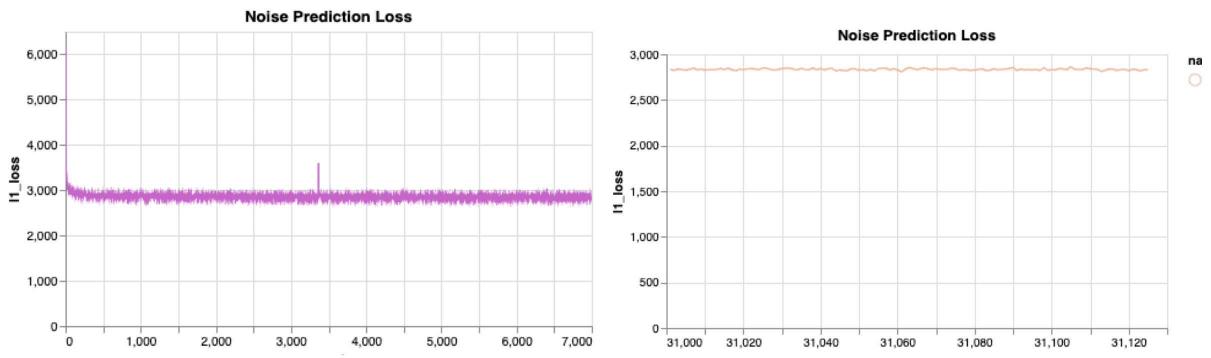
5.3 UNet + ResNet50: Places365 Dataset Training

The training process was conducted by varying the number of noise layers. The first experiment was conducted by adding 10 noise layers and extracting them using the UNet. Similarly, the number of noise layers was changed to 5 for further experimentation to identify the ideal number of noise layers required to determine whether there is an attack on the input image.

5.3.1 Training with 10 Noise Layers

5.3.1.1 UNet Training

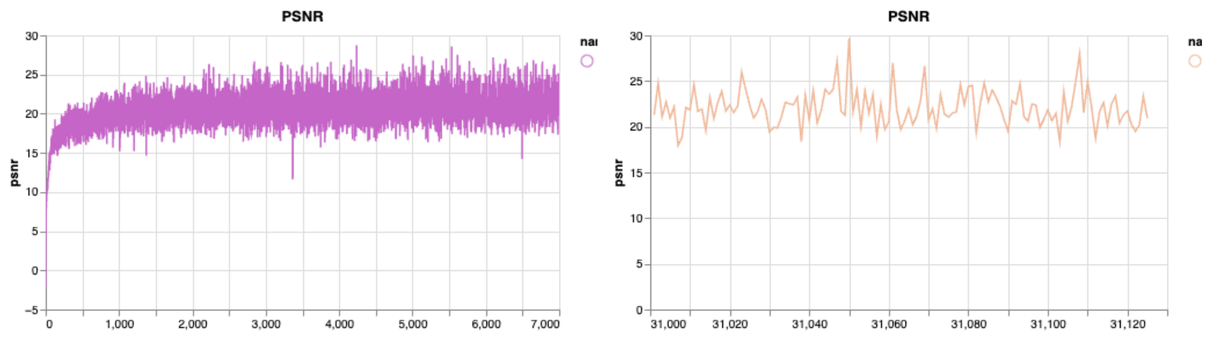
The experiments were continued from the above experiments using the Places365 dataset. The training was done in two separate sections where the first training was stopped and then started again from the last checkpoint of the first section. The L1 loss is as follows during the first and second sections. There was no purpose for breaking the training into two sections, but due to a technical issue, the training had to be begun again from the last saved checkpoint from the first run. There were also some missing iterations in the middle of the graph where the model was actually trained but lacks visualization due to the technical issue.



(a) Training first run *(b) Training second run*

Figure 5.6: Places365 Training-Training loss of UNet on 10 noise layer (L2 loss)

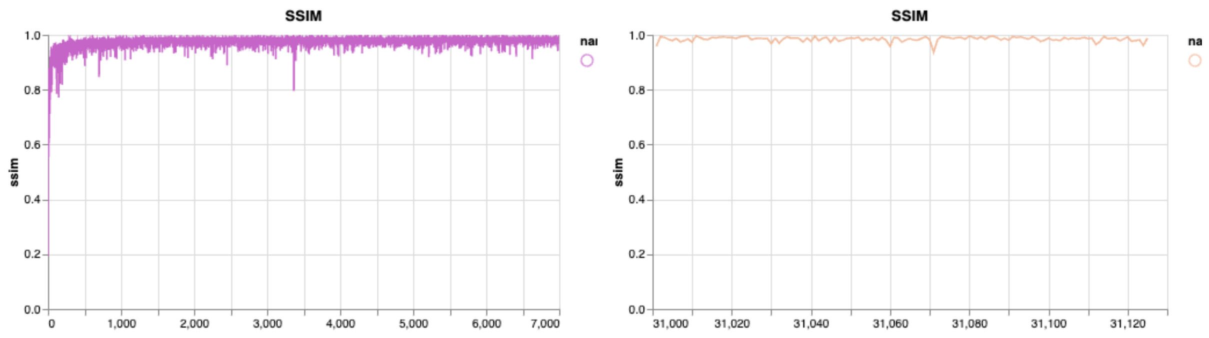
The PSNR and SSIM values are as shown in the figure 5.7 and figure 5.8. The PSNR value has been varied a lot between 15 and 30 while the SSIM value is very close to the 1.



(a) PSNR Value during first run *(b) PSNR Value during second run*

Figure 5.7: PSNR between denoised image and the original image - UNet training on 5 noise layer

After the above training, the following images in the figure 5.9, figure 5.10 and figure 5.11 were took from the Weights and Biases logs. This is same as the experiments done with the CIFAR10 dataset.



(a) SSIM Value during first session *(b) SSIM Value during second session*

Figure 5.8: SSIM between denoised image and the original image - UNet training on 10 noise layer

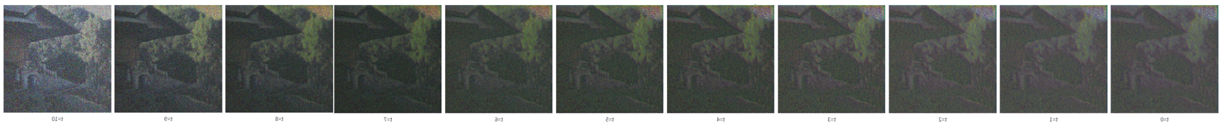


Figure 5.9: Places365 Training sample during first iteration

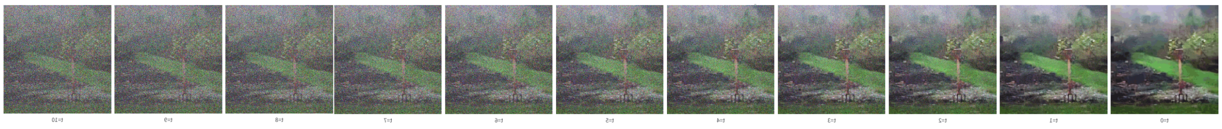


Figure 5.10: Places365 Training sample during a middle iteration

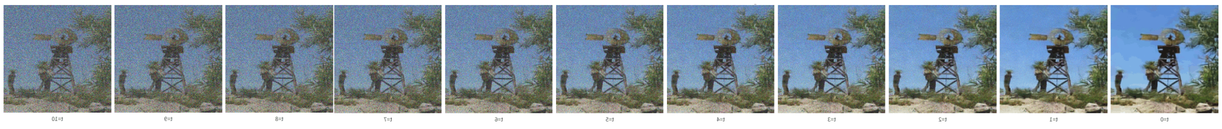


Figure 5.11: Places365 Training sample during last iteration

5.3.1.2 Preliminary experiment for hypothesis validation

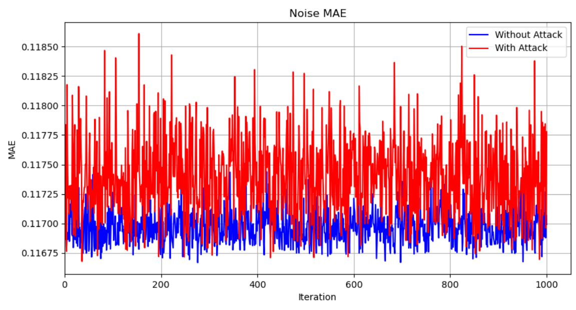
The following experiments were conducted to validate the hypothesis before proceeding into further experiments. The hypothesis was based on the Lorenz et al. (2024) study, where an attacked image and a non-attacked image would have different distributions after going through the diffusion process (refer to the research hypothesis section 1.4.1).

The experiments were conducted using 1000 samples of the Places365 dataset with and without attacks to ensure that the extracted image and denoised image would have a different distribution after going through the UNet denoising model compared to a non-attacked image going through the UNet denoising model. For this verification, MAE, PSNR, and SSIM were used to prove there is a difference in distributions. For each metric, the original image and the extracted denoised image were used. To verify that the attacks were not biased toward the trained attack method, two types of attacks, PGD and FGSM, were performed on the input images to observe the effects.

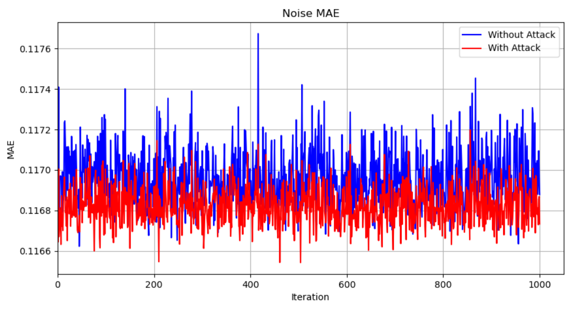
Each graph contains two lines which indicate attacked or without attack which means,

1. **Without attack** - Without adding any attacks, the original image passed through the trained UNet model to observe the evaluation metric value.

2. **With attack** - Adding an attack to the input image and passed through the UNet model to observe evaluation metrics.

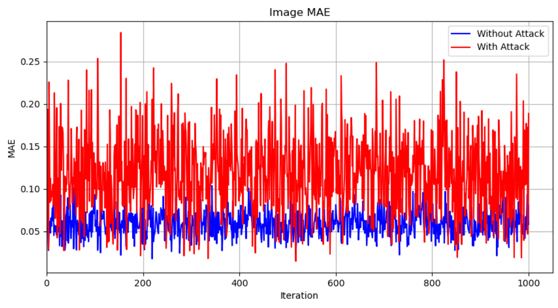


(a) FGSM Attack

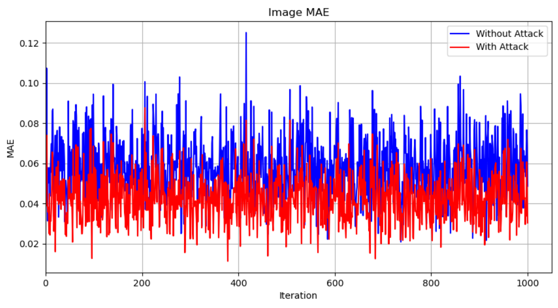


(b) PGD Attack

Figure 5.12: MAE value plots of attacked and without attacked image for 1000 image samples comparing extracted noise and added noise layers

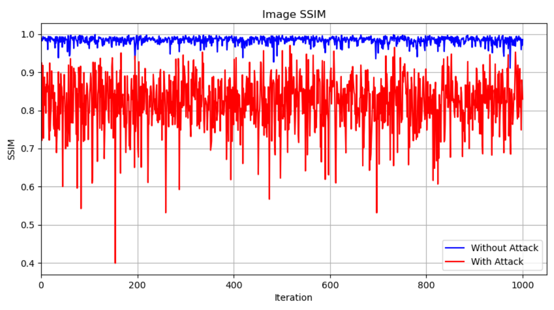


(a) FGSM Attack

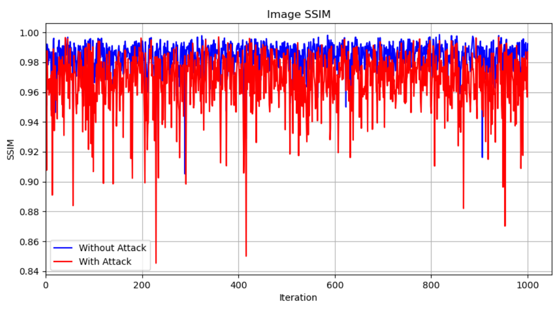


(b) PGD Attack

Figure 5.13: Evaluation done using MAE metric on two different attacks - FGSM and PGD

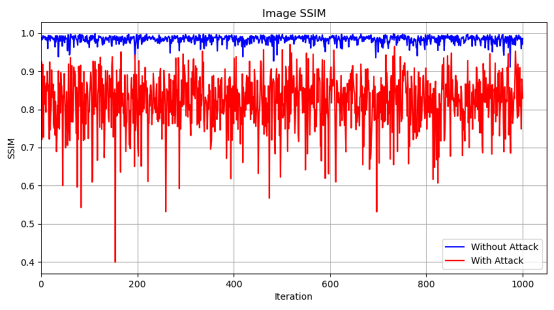


(a) FGSM Attack

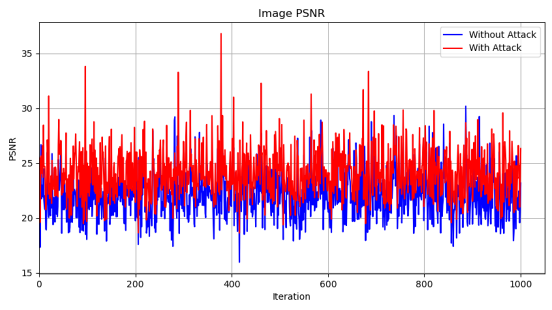


(b) PGD Attack

Figure 5.14: Evaluation done using SSIM metric on two different attacks - FGSM and PGD



(a) FGSM Attack



(b) PGD Attack

Figure 5.15: Evaluation done using PSNR metric on two different attacks - FGSM and PGD

5.3.1.3 ResNet50 Training

Attacks were performed randomly on the inference input of the UNet model to create binary classifier training samples. Noise extracted images were used to train the classifier network. The process ensured that an equal number of attacked and non-attacked samples were shuffled randomly and used to train the classifier model. The classification network was trained for 790 iterations with a batch size of 64, totaling 50,560 samples, as shown in Figure 5.16. This graph has been smoothed to visualize the decrease of the Binary Cross Entropy loss over the iterations to identify how well the model learns.

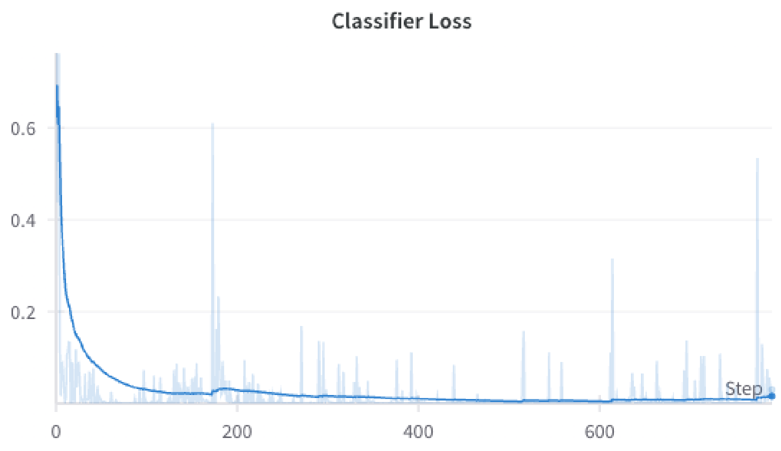
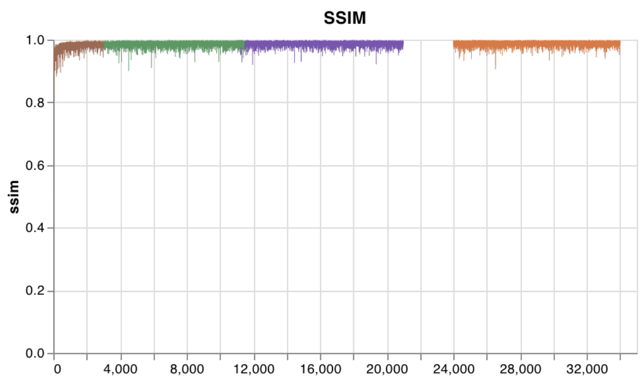


Figure 5.16: UNet - ResNet50 Binary Classifier Training Loss (BCE Loss) - Denoised image to classify attack or not

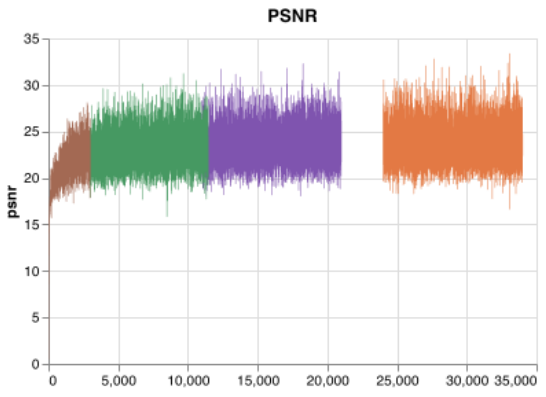
5.3.2 Training with 5 Noise Layers

5.3.2.1 UNet Training

The above experimental setup was conducted with 10 noise layers and was kept the same in this training process, except the number of noise layers added to the image was changed to 5 noise layers. The training process is similar to the above, as shown in Figure 5.17 on the Places365 dataset. Due to a technical issue, the iterations between 21,000 and 24,000 were missing from the total of 34,000 iterations.



(a) SSIM Value variation during training



(b) SSIM Value variation during training

Figure 5.17: SSIM & PSNR values during training - UNet training on 5 noise layer

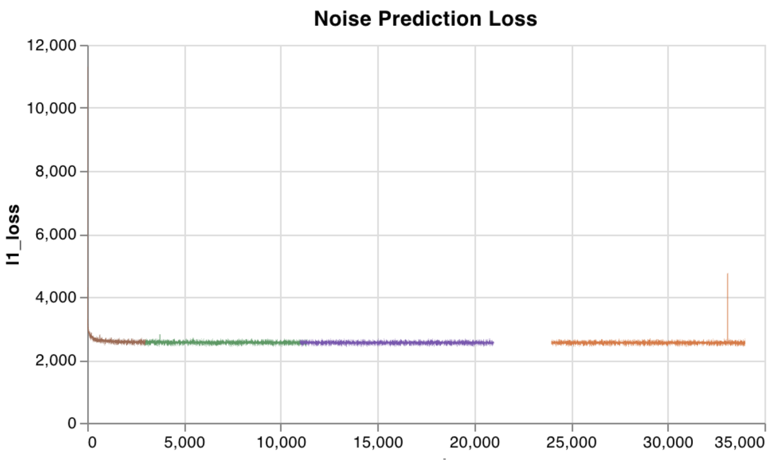


Figure 5.18: Places365 Training-Training loss of UNet on 5 noise layer (L2 loss)

5.4 UNet + ResNet50 (Merged Noise Classifier)

The UNet model trained in the previous experiment (with 10 noise layers) was used to conduct this experiment using the Places365 dataset, as the UNet architecture remained unchanged. Only the inputs to the classifier network were modified, as shown in the architecture in Figure 3.2. Since the input to the classifier network (ResNet50) was changed, it was necessary to re-train the ResNet50 classifier. The 10 extracted noise layers from the UNet were summed together into a single noise layer and fed into the classifier network to classify whether an attack exists or not. The graph in Figure 5.19 represents the learning curve of the ResNet50 classifier under this setup.

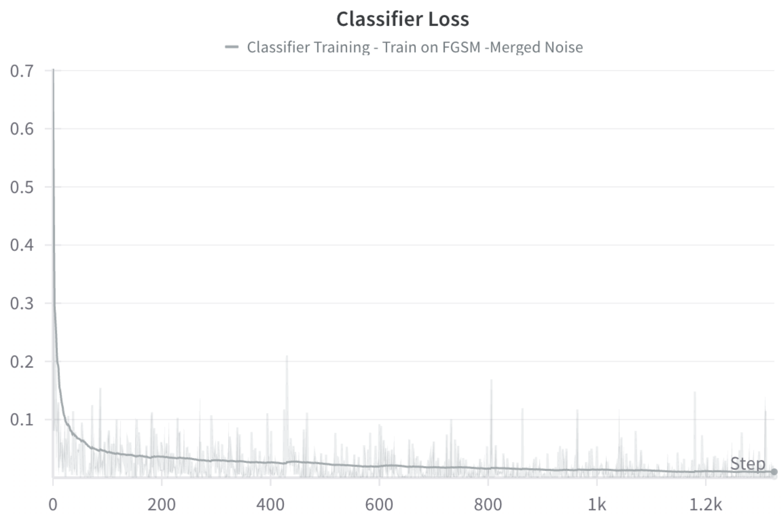


Figure 5.19: ResNet50 Binary Classifier Training Loss (BCE Loss) - Input merged noise layers to classify attack or not

5.5 DDPM + ResNet50

There was no training data for DDPM, as the author used the pretrained DDPM from the Saharia et al. (2023) study. The author experimented to determine whether attacking an image on the diffusion model would affect the image generation process of the diffusion model (Figure 5.20).

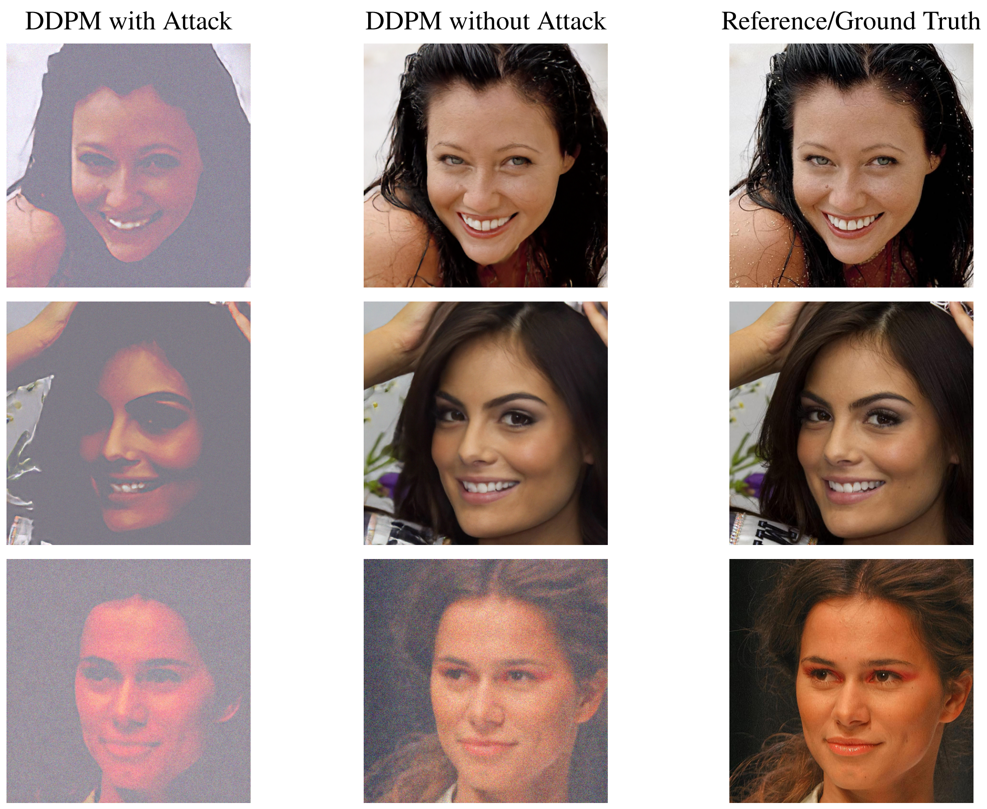


Figure 5.20: The DDPM generated outputs using both attacked and non-attacked images, referencing them against the ground truth image.

There was no clear difference between the attacked image received from the diffusion model and the non-attacked image received from the diffusion model to the human eye. However, the distribution could be different, which may not be visible visually. Thus, these training data were fed into the classifier to observe whether the classifier would be able to learn to differentiate the attacked and non-attacked images received from the DDPM. The training loss plotted in Figure 5.21 clearly shows that the ResNet50 classifier was not able to distinguish the attacks after 128 iterations, as the graph in Figure 5.21 shows a non-decreasing function even after smoothing.

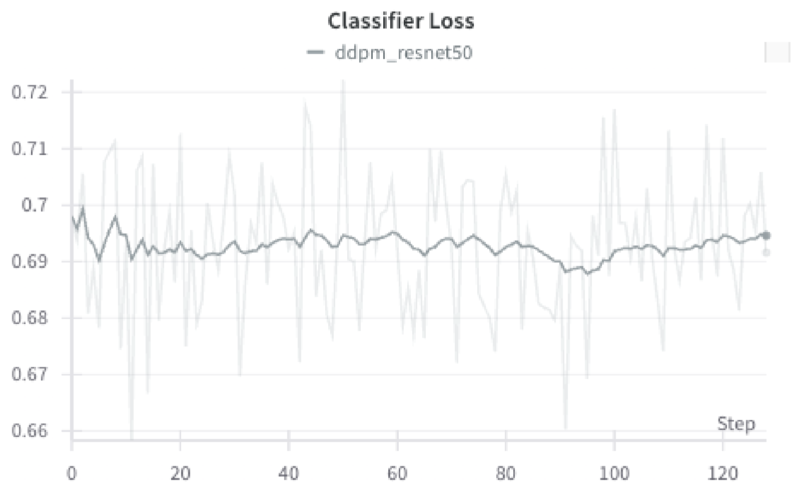


Figure 5.21: DDPM - ResNet50 Binary Classifier Training Loss (BCE Loss)

5.6 DDIM + ResNet50

Similar to the above experiment, as explained in the previous chapter, DDIM was used to generate training samples for the classifier network more efficiently. It is observed that there is no significant improvement after upgrading the DDPM model to a DDIM model, since the classifier training loss also remains a non-decreasing function, as the smoothed graph shown in Figure 5.22.

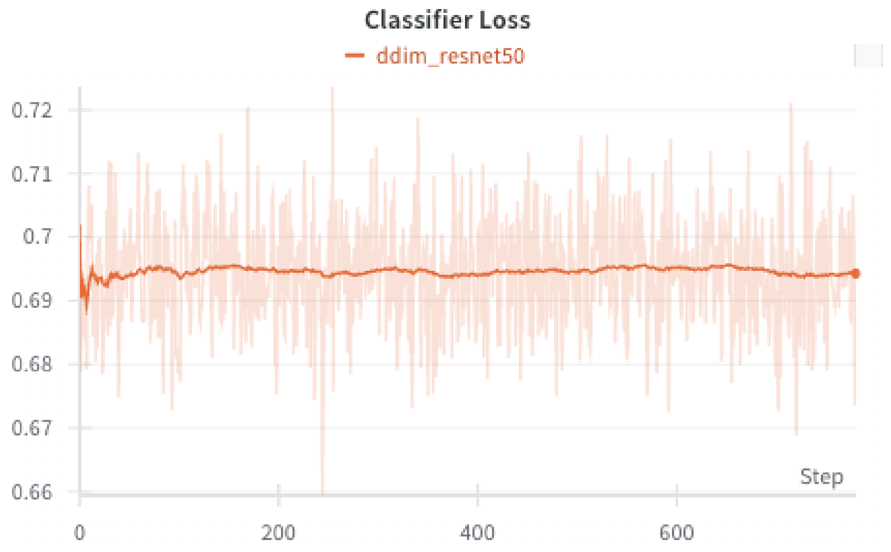


Figure 5.22: DDIM - ResNet50 Binary Classifier Training Loss (BCE Loss)

5.7 DDIM + CNN

The author only replaced the ResNet50 binary classifier from the above experiment with a small CNN binary classifier using Binary Cross Entropy loss. Even though there seems to be a decreasing loss function of BCE (Figure 5.23), the BCE does not change after the first few iterations (Figure 5.24).

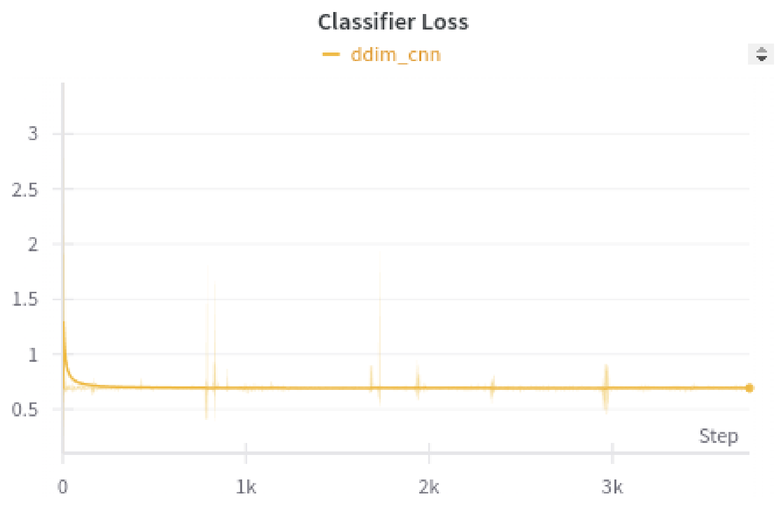
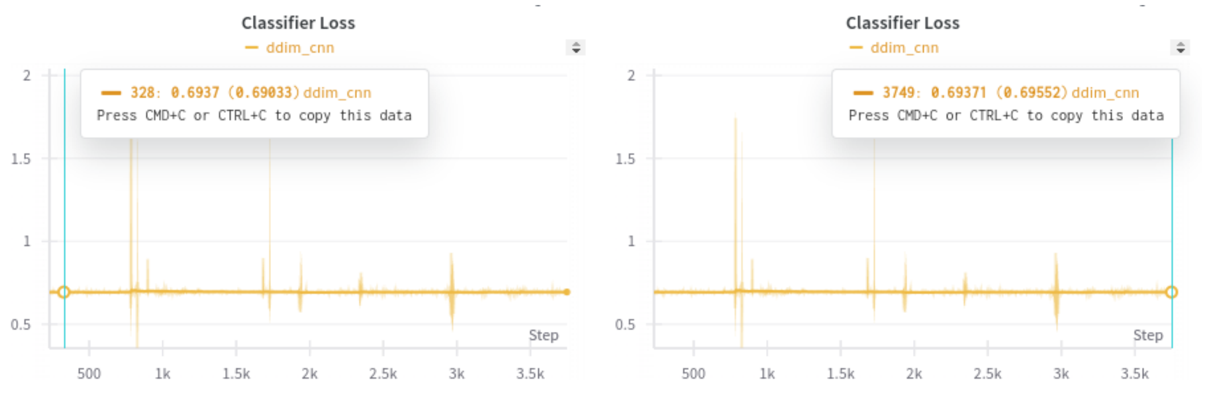


Figure 5.23: DDIM-CNN Binary Classifier Training Loss (BCE Loss)



(a) CNN Training Loss - 328 Iteration

(b) CNN Training Loss - 3749 Iteration

Figure 5.24: DDIM-CNN Binary Classifier Training Loss (BCE Loss) during 328th and 3749th iterations

5.8 Chapter Conclusion

All models introduced in the previous section were trained to evaluate their suitability for the proposed solution. The training performance of the DDPM + ResNet50, DDIM + ResNet50, and DDIM + CNN models demonstrated relatively linear learning curves, suggesting limited learning capacity in the context of adversarial detection. In contrast, the UNet + ResNet50 configuration showed promising learning behavior, indicating its potential as a better fit for the proposed approach.

However, rather than relying solely on preliminary observations, the author proceeded with a more in-depth analysis and testing in subsequent sections. A key commonality among the DDPM + ResNet50, DDIM + ResNet50, and DDIM + CNN experiments was the use of a pretrained diffusion model, with only the ResNet50 classifier being trained to detect adversarial attacks. The learning curves from these classifiers revealed minimal loss reduction, suggesting that the models failed to effectively learn the distinction between adversarial and clean samples.

CHAPTER 6: EVALUATIONS AND RESULTS

6.1 Chapter Overview

This chapter presents the results obtained from the experiments conducted using the trained models discussed in the previous section. The primary focus is placed on the UNet + ResNet50 configuration, which was evaluated under various testing setups. These include experiments involving different numbers of noise layers to assess their impact on model performance and to identify the optimal number of layers for adversarial detection. Furthermore, a series of adversarial attacks were applied to evaluate the robustness and generalizability of the proposed solution.

6.2 UNet: CIFAR10 Testing

Testing for this experiment was skipped since the resolution of the images is too small (32×32), as explained in the training phase of this experiment (Reference 5.2).

6.3 UNet + ResNet50: Places365 Testing

Experiment under this setup has done with several configurations but all the tests were based on the Places365 Dataset.

1. UNet Evaluation for 10 Noise Layers on Places365 Dataset
2. ResNet50 Evaluation for 10 Noise Layers on Places365 Dataset
   - 2.1. ResNet50 Evaluation with Different Number of Noise Layers
   - 2.2. ResNet50 Evaluation with Different Types of Attacks
   - 2.3. Performance evaluation of the pipeline consists of both UNet and ResNet models
3. ResNet50 Evaluation for 5 Noise Layers on Places365 Dataset
   - 3.1. ResNet50 Evaluation with Different Number of Noise Layers
   - 3.2. ResNet50 Evaluation with Different Types of Attacks
   - 3.3. Performance evaluation of the pipeline consists of both UNet and ResNet models

Note that the number of noise layers here refers to the number of random noise layers that are added to the image during the initial step of training the UNet. These noise layers are extracted by the UNet to obtain the denoised image (Refer to the Architecture Diagram 3.1).

6.3.1 Attack Detection Accuracy

The classifier robustness was experimented with different types of attacks. The attack was added to the input image after adding the noise layers, but performing the attack before adding the noise layers would not make any difference to the final input image for the UNet. The extracted image with different attacks at different levels of attack was evaluated under this experiment. The following attacks were conducted with the following parameters. The tested dataset contains 10,140 images, processed as a batch of 64 images in each iteration, for a total of 156 iterations to feed into the classifier for testing.

#	**Attack**	*standard deviation*				**5 Layer**	**10 Layer**
1	Gaussian Noise	0.1				99.90%	99.67%
		ϵ	α	$steps$			
2	PGD	0.05	$\frac{2}{255}$	10		99.90%	99.64%
3	PGD	0.15				99.90%	99.75%
4	PGD	0.30				99.90%	99.75%
		ϵ					
5	FGSM	0.05				99.90%	99.63%
6	FGSM	0.15				99.90%	99.77%
7	FGSM	0.30				99.90%	99.77%
		c	κ	lr	$steps$		
8	CW	1	0	0.01	50	99.90%	99.53%

Table 6.1: Testing phase: Robustness to different attacks on Places365 Dataset

The attacks were applied equally across each type of attack for the 5 noise layer model, as shown in Figure 6.1, resulting in the overlapping of the plot lines into a single line.

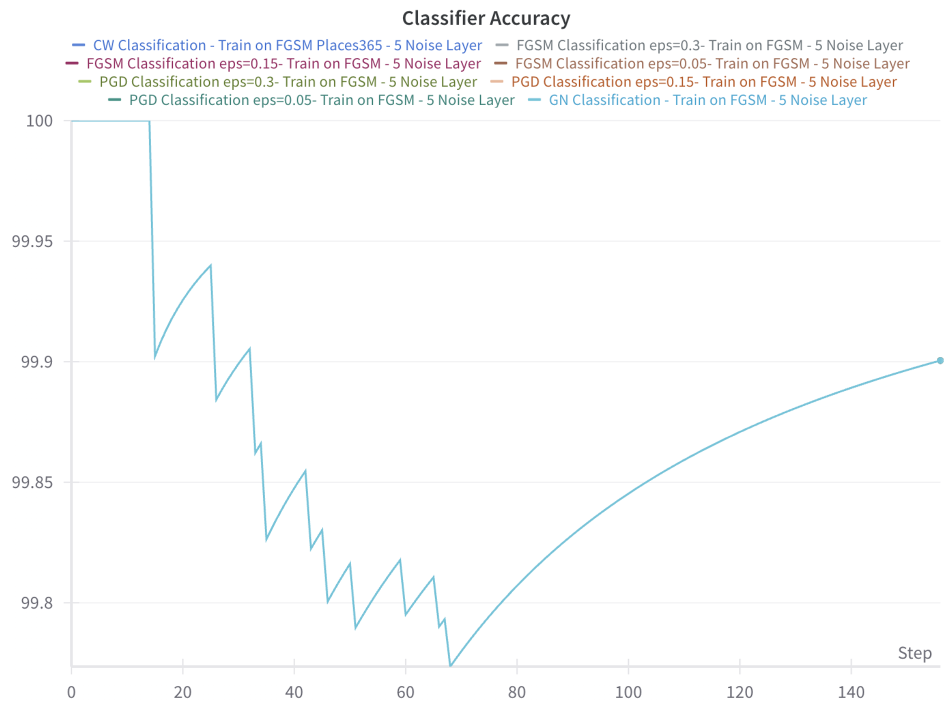


Figure 6.1: Testing Phase(5 Noise Layers): Robustness to different attacks on Places365 Dataset

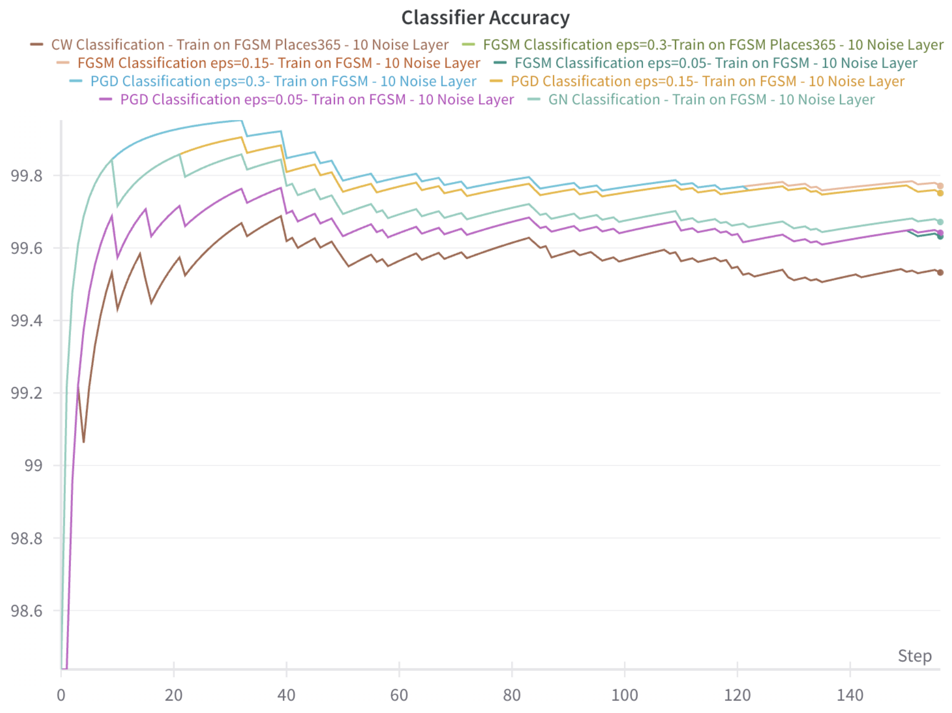


Figure 6.2: Testing Phase(10 Noise Layers): Robustness to different attacks on Places365 Dataset

In order to evaluate the model’s robustness across different datasets, the author conducted experiments using the CelebA dataset with a gender classification model. The accuracy curves obtained from these evaluations are illustrated in Figure 6.3.

#	Attack	Accuracy of 5 Layer model on CelebA Dataset
1	Gaussian Noise	95.58%
2	PGD	95.07%
3	PGD	95.37%
4	PGD	96.40%
5	FGSM	94.83%
6	FGSM	94.92%
7	FGSM	94.99%

Table 6.2: Testing phase: Robustness to different attacks on CelebA Dataset

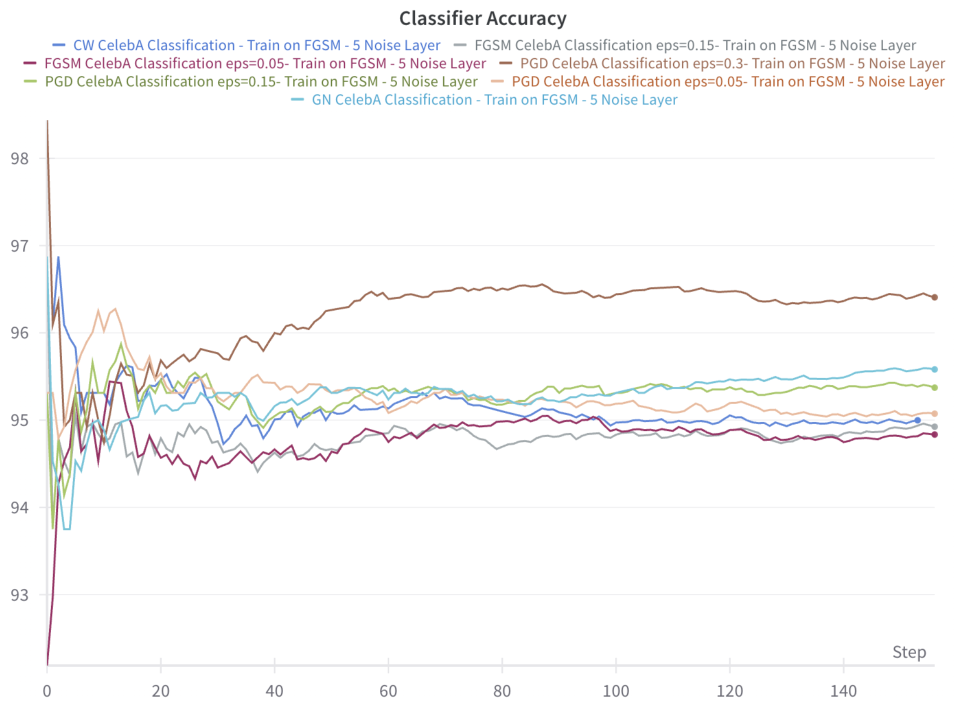


Figure 6.3: Testing Phase(5 Noise Layers): Robustness to different attacks on CelebA Dataset

6.3.2 Number of Noise Layers Effectiveness

Even though the classifier was trained with a fixed number of noise layers, where the UNet was trained to extract a fixed number of noise layers (10 or 5) to obtain a denoised image, and the ResNet50 classifier was trained to classify whether the image was attacked or not based on the denoised output, further tests were conducted by varying the number of noise layers added during the inference process. This was done to evaluate the trade-off between the number of noise layers and the classification accuracy of the entire network.

The previously trained ResNet50 and UNet models with 5 noise layers were tested with a reduced number of noise layers, 5, 4, 3, 2, and 1 to assess the effectiveness of varying the noise layer count.

Similarly, the ResNet50 and UNet models trained with 10 noise layers were tested with different numbers of noise layers during this test — 2, 4, 6, 8, and 10.

Model (No. of noise layers the model has trained)	**FGSM ($\epsilon = 0.3$)**	
	No. of Noise Layers	**Accuracy**
5	1	99.92%
	2	99.92%
	3	99.92%
	4	99.91%
	5	99.90%
10	2	94.08%
	4	98.51%
	6	99.20%
	8	99.55%
	10	99.77%

Table 6.3: Testing Phase: effectiveness of noise layers

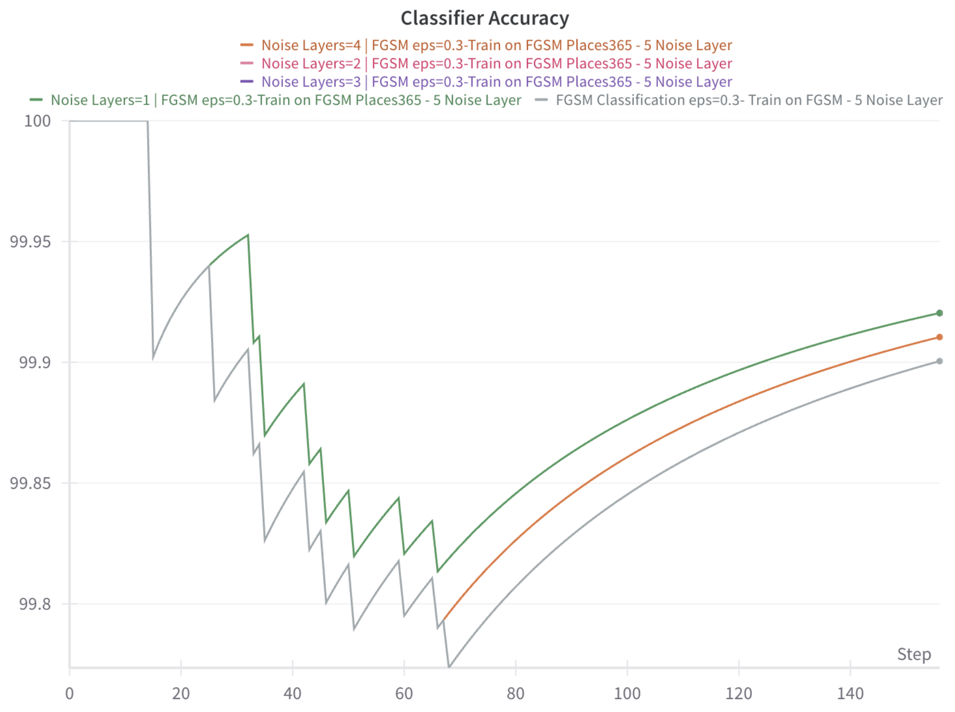


Figure 6.4: Testing Phase(5 Noise Layers): effectiveness of noise layers

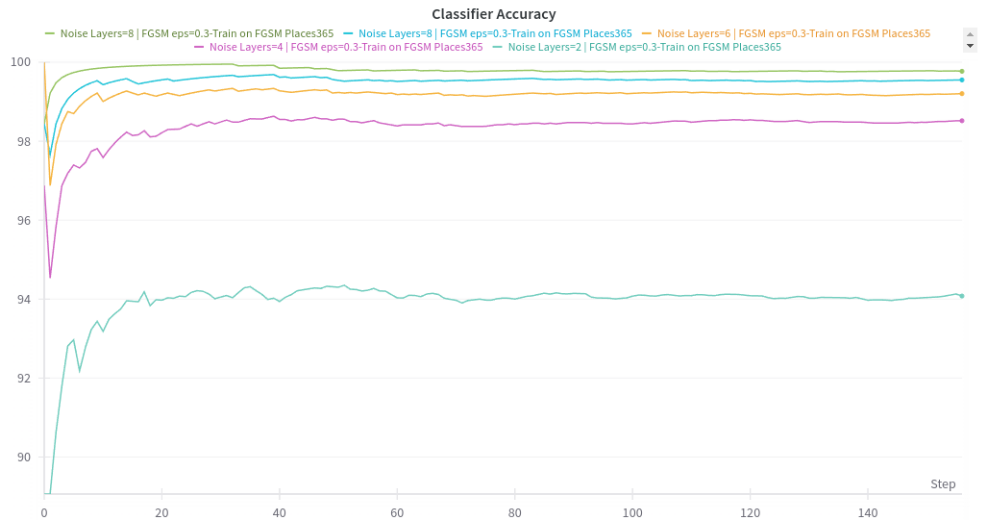


Figure 6.5: Testing Phase(10 Noise Layers): effectiveness of noise layers

6.3.3 Performance Evaluation

Both model pipelines, consisting of both the UNet image purifier and the ResNet50 classifier, perform under Places365 10,140 image samples. The time taken for the entire image classification process is divided by the time taken for classifying all 10,140 images.

Performance is considered for the above Table 6.2. Refer to Table 6.2 for attack configurations.

#	Attack	Time in minutes for all testing samples		Average Time in milliseconds for one sample	
		5 Layer	**10 Layer**	**5 Layer**	**10 Layer**
1	Gaussian Noise	55.75252	109.65887	329	649
2	PGD	71.41789	124.89266	423	739
3	PGD	71.11372	125.62177	421	743
4	PGD	71.11100	125.53311	421	743
5	FGSM	56.11143	109.68333	332	649
6	FGSM	56.11189	110.33841	332	653
7	FGSM	56.11189	109.9742	332	651
8	CW	69.86054	126.88656	413	750

Table 6.4: Testing Phase: Performance of the models based on different attacks

Performance based on the number of noise layers referring to the Table 6.3 as follows,

Model (No. of noise layers the model has trained)	FGSM ($\epsilon = 0.3$)		
	No. of Noise Layers	**Time in minutes for all test samples**	**Average time in milliseconds for one sample**
5	1	13.11059	77
	2	23.31707	138
	3	34.2046	202
	4	44.95609	266
	5	55.86192	331
10	2	25.01688	148
	4	46.2248	274
	6	68.25018	404
	8	90.11074	533
	10	109.9742	651

Table 6.5: Testing Phase: Performance under different number of noise layers

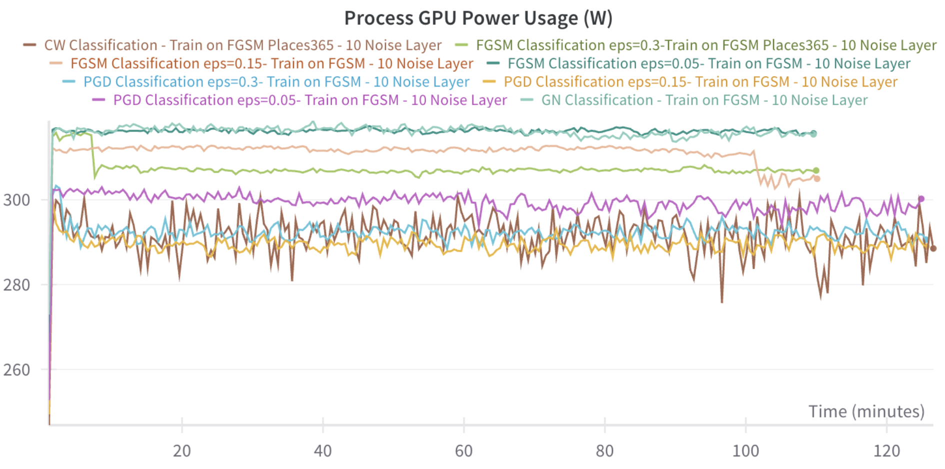


Figure 6.6: Testing Phase(10 Noise Layer): Power and time usage during attacks

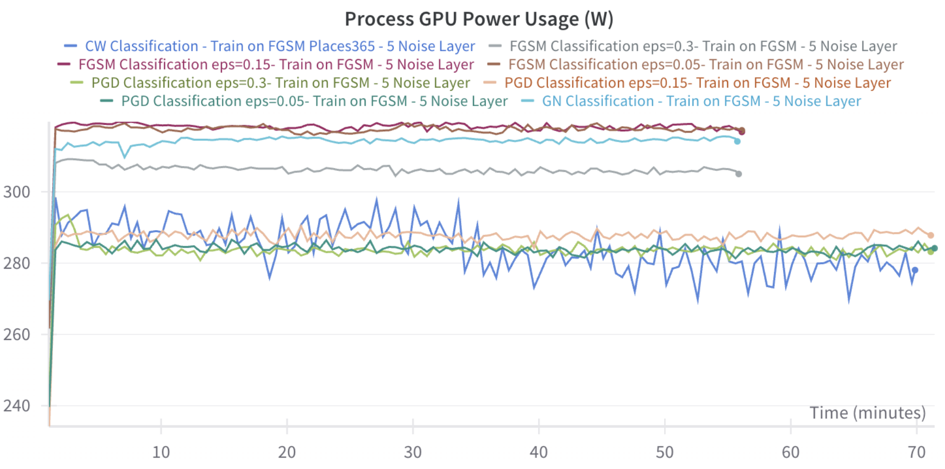


Figure 6.7: Testing Phase(5 Noise Layer): Power and time usage during attacks

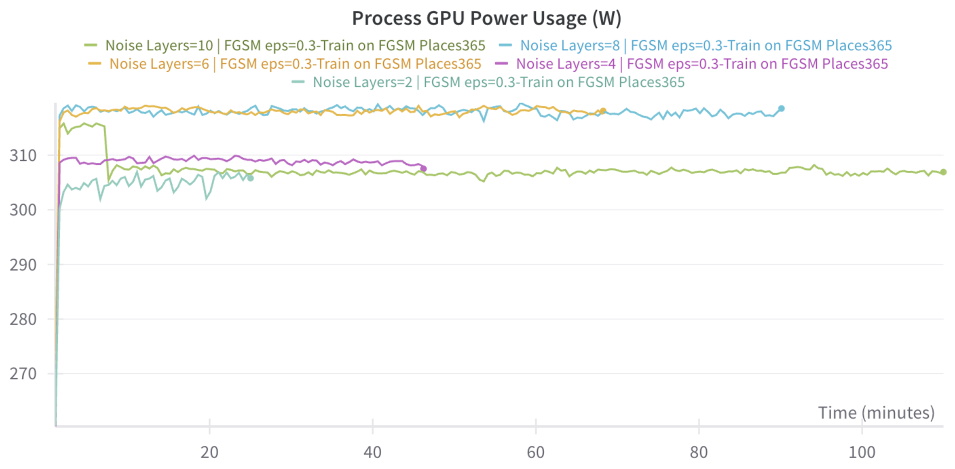


Figure 6.8: Testing Phase(10 Noise Layer): Power and Time usage during noise layer evaluations

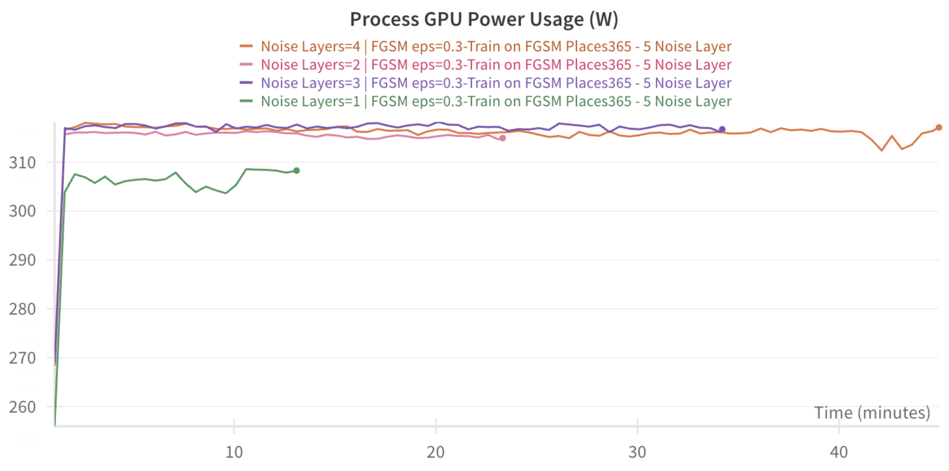


Figure 6.9: Testing Phase(10 Noise Layer): Power and Time usage during noise layer evaluations

6.4 DDPM + ResNet50, DDIM + ResNet50 and DDIM + CNN Testing

The tests for these experiments were skipped due to the models being untrained. As observed in the previous model training section, the models were unable to learn to differentiate between attacked and non-attacked images, as the learning curve remained almost linear throughout the training process.

6.5 Critical Evaluation of Results

The UNet and ResNet50 models were able to achieve excellent results, surpassing the rest of the other diffusion models mentioned in the thesis.

It is observed that the model trained with 5 noise layers obtained a higher uniform accuracy of 99.9%, with only a 0.37% maximum accuracy gap compared to the model trained on 10 noise layers (Table 6.2). A key highlight is that the 5-layer model maintained a constant accuracy of 99.9% across different attacks as well as across different numbers of noise layers (Table 6.3).

It is worth noting that decreasing the number of noise layers increases the visibility of the image, thereby reducing the privacy provided by the noise addition process. Thus, the number of noise layers is directly proportional to the level of privacy added. Privacy here refers to the fact that noise can be added at the source device before transmitting the image through the network to the destination, making it difficult for an intruder to identify the original image since it is obscured by noise.

The performance of these models (Table 6.5) demonstrates that they can be deployed in real-world environments, as they are capable of detecting adversarial samples in less than half a second.

6.6 Chapter Conclusion

The UNet + ResNet50 model trained with 5 noise layers demonstrated greater overall effectiveness in handling adversarial attacks compared to the model trained with 10 noise layers. Specifically, the 5-layer model maintained consistent accuracy across various attack types, whereas the 10-layer model exhibited fluctuating accuracy depending on the nature of the attack. Furthermore, the 5-layer model displayed robustness in scenarios involving different numbers of noise layers during training and inference, suggesting its ability to generalize better under varying conditions. In conclusion, the model trained with 5 noise layers proved to be both more efficient and more accurate, making it a stronger candidate for the proposed adversarial detection framework.

CHAPTER 7: CONCLUSION

7.1 Chapter Overview

This chapter presents a summary and conclusion of the research, reflecting on the key findings and achievements derived from the previous chapters. It highlights the challenges encountered during the research process and outlines the strategies implemented to overcome them, ultimately leading to promising results from the proposed novel architecture. The chapter also discusses any deviations from the original plan and suggests potential directions for future enhancements. Finally, the chapter concludes by summarizing the overall contributions and significance of the research.

7.2 Achievements of Research Aims, Objectives Research Questions

7.2.1 Aim of the Project

The aim of this research is to design and develop a novel architecture that embeds a set of predefined noise layers into an image as a watermark and enables the extraction of this noise to classify whether an adversarial attack exists, based on the final denoised output image.

The aim of this research was successfully achieved using a UNet architecture along with a ResNet50 classifier. The hypothesis was successfully proven within Section 5.3.1.2, where there is a separation of benign samples and adversarial samples.

7.2.2 Research Questions and Objectives

This research was guided by a set of clearly defined research questions and objectives aimed at addressing the growing challenge of adversarial attacks on deep neural networks. The central research question explored how predefined noise added as a watermark could be extracted using diffusion-based models to detect whether an image has been manipulated through adversarial attacks.

To this end, a series of sub-questions were investigated, including the impact of the number of noise layers on the denoising process, the applicability of diffusion models in extracting noise, and the efficiency of the diffusion process in real-time scenarios. These were aligned with objectives that involved designing and implementing a novel architecture inspired by diffusion processes, evaluating its performance on various datasets and attack types, and contributing to the understanding of adversarial image behavior.

Throughout the experimentation and implementation phases, the research demonstrated that the proposed architecture, particularly the UNet and ResNet50 combination—was capable of effectively detecting adversarial interference in images. The findings validated the hypothesis that noise extracted through a diffusion-inspired process carries distinguishable patterns for adversarially perturbed images, and these patterns could be leveraged for robust classification.

Overall, the research successfully fulfilled its primary objectives and answered the corresponding research questions, laying the foundation for future exploration of diffusion based watermarking techniques as a defense against adversarial attacks.

7.3 Problems and Challenges Faced

The author conducted several experiments in order to improve the novelty and performance of the proposed solution. Throughout these experiments, multiple challenges were encountered, most of which were successfully mitigated.

Initial experiments using a pre-existing UNet architecture combined with a ResNet50 classifier yielded promising results. These models were sourced from well-established implementations—UNet from a diffusion model architecture and ResNet50 from the PyTorch model zoo. However, subsequent experiments involving custom-designed models from scratch did not perform as well, highlighting the challenges in recreating complex architectures with equivalent performance.

Further experimentation involving DDPM and DDIM models required not only a deep theoretical understanding but also practical implementation knowledge. In particular, while the diffusion process is described theoretically as an iterative denoising procedure, its practical implementation relies on the use of schedulers. The author utilized a pre-trained DDPM model and modified it to follow a DDIM approach, which required a careful translation of mathematical theory into functional code.

Due to the computational complexity and size of diffusion models, running and debugging them locally posed a significant challenge. With the guidance of the co-supervisor, the author was able to understand the underlying codebase. The inference process was locally executed using an RTX 3080 (10GB) GPU by strategically placing breakpoints in the PyCharm debugger to trace data flow across functions. Although local resource limitations were a constraint, they were partially addressed using Kaggle's virtual machines equipped with P100 GPUs and local training sessions on the RTX 3080.

7.4 Deviations

The project was completed according to the initial plan without any major deviations. The UNet + ResNet50 model demonstrated strong performance and outperformed the alternative experimental setups discussed in previous chapters. While other models were explored, they did not yield satisfactory results, affirming the validity and strength of the selected architecture.

7.5 Future Enhancements

This research primarily focused on introducing a novel approach to adversarial attack detection by embedding noise layers into images as a unique form of watermarking. These noise layers were later extracted, and the denoised image was used to determine the presence of adversarial perturbations. The central objective was to enhance the robustness of the model, while computational efficiency, particularly GPU utilization was not a primary concern.

The UNet + ResNet50 architecture demonstrated strong performance with minimal GPU usage, making it an efficient and effective baseline. However, the diffusion-based models, such as DDPM and DDIM, imposed significantly higher computational demands. Although resource optimization for these models was beyond the scope of this research, it remains a viable direction for future work. Specifically, optimizing the diffusion process to reduce GPU consumption could enable more practical deployment of such models.

To address some of these challenges, the author attempted to convert DDPM models into DDIM models to improve inference efficiency. However, the combined diffusion-classifier models did not succeed in learning to differentiate between benign and adversarial samples. This limitation was primarily due to the lack of a proper noise scheduling mechanism in the experiments; instead of leveraging schedulers to add noise progressively, noise was added directly in an iterative manner. Integrating a proper scheduler into the training process remains an important avenue for future exploration, which could significantly enhance model learning and adversarial detection accuracy. This would have better results in purification as well which was a limitation in this research.

7.6 Concluding Remarks

This research introduced a novel approach to adversarial attack detection by leveraging noise-based watermarking through diffusion-inspired architectures. The experiments demonstrated the potential of using extracted noise patterns and denoised images to effectively classify adversarial samples, particularly with the UNet + ResNet50 architecture. Despite challenges in training diffusion models due to computational constraints and implementation complexity, the findings lay a strong foundation for future enhancements. The proposed framework offers a promising direction for building robust, noise-aware defense mechanisms against adversarial attacks in deep learning models.

References

Apostolidis, K. D. and Papakostas, G. A. (2021), ‘A survey on adversarial deep learning robustness in medical image analysis’, *Electronics* **10**(17), 2132.

Bai, T., Luo, J., Zhao, J., Wen, B. and Wang, Q. (2021), ‘Recent advances in adversarial training for adversarial robustness’.
URL: *https://arxiv.org/abs/2102.01356*

Baluja, S. and Fischer, I. (2018), ‘Learning to attack: Adversarial transformation networks’, *Proceedings of the AAAI Conference on Artificial Intelligence* **32**(1).
URL: *https://ojs.aaai.org/index.php/AAAI/article/view/11672*

Berahmand, K., Daneshfar, F., Salehi, E. S., Li, Y. and Xu, Y. (2024), ‘Autoencoders and their applications in machine learning: a survey’, *Artificial Intelligence Review* **57**(2), 28.

Biggio, B. and Roli, F. (2018), ‘Wild patterns: Ten years after the rise of adversarial machine learning’, *Pattern Recognition* **84**, 317–331.
URL: *https://www.sciencedirect.com/science/article/pii/S0031320318302565*

Bissoto, A., Valle, E. and Avila, S. (2021), Gan-based data augmentation and anonymization for skin-lesion analysis: A critical review, *in* ‘Proceedings of the IEEE/CVF conference on computer vision and pattern recognition’, pp. 1847–1856.

Blau, T., Ganz, R., Kawar, B., Bronstein, A. and Elad, M. (2022), ‘Threat model-agnostic adversarial defense using diffusion models’.
URL: *https://arxiv.org/abs/2207.08089*

Carlini, N. and Wagner, D. (2017), ‘Towards evaluating the robustness of neural networks’.

Chahe, A., Wang, C., Jeyapratap, A., Xu, K. and Zhou, L. (2024), ‘Dynamic adversarial attacks on autonomous driving systems’.

Cisse, M., Adi, Y., Neverova, N. and Keshet, J. (2017), ‘Houdini: Fooling deep structured prediction models’.
URL: *https://arxiv.org/abs/1707.05373*

Dai, T., Cai, J., Zhang, Y., Xia, S.-T. and Zhang, L. (2019), Second-order attention network for single image super-resolution, *in* 'Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)'.

Dai, T., Feng, Y., Chen, B., Lu, J. and Xia, S.-T. (2022), 'Deep image prior based defense against adversarial examples', *Pattern Recognition* **122**, 108249.
URL: *https://www.sciencedirect.com/science/article/pii/S0031320321004295*

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. and Fei-Fei, L. (2009), ImageNet: A Large-Scale Hierarchical Image Database, *in* 'CVPR09'.

Deng, L. (2012), 'The mnist database of handwritten digit images for machine learning research [best of the web]', *IEEE Signal Processing Magazine* **29**(6), 141–142.

Dziugaite, G. K., Ghahramani, Z. and Roy, D. M. (2016), 'A study of the effect of jpg compression on adversarial images'.

Fei, J., Xia, Z., Tondi, B. and Barni, M. (2022), 'Supervised gan watermarking for intellectual property protection'.

Gondim-Ribeiro, G., Tabacof, P. and Valle, E. (2018), 'Adversarial attacks on variational autoencoders'.
URL: *https://arxiv.org/abs/1806.04646*

Gong, X., Chen, Y., Wang, Q. and Kong, W. (2023), 'Backdoor attacks and defenses in federated learning: State-of-the-art, taxonomy, and future directions', *IEEE Wireless Communications* **30**(2), 114–121.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. (2014), 'Generative adversarial networks'.
URL: *https://arxiv.org/abs/1406.2661*

Goodfellow, I. J., Shlens, J. and Szegedy, C. (2015), 'Explaining and harnessing adversarial examples'.

Guo, L., Wang, C., Yang, W., Huang, S., Wang, Y., Pfister, H. and Wen, B. (2023), Shadowdiffusion: When degradation prior meets diffusion model for shadow removal, *in* 'Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition', pp. 14049–14058.

Hashemi, A. S. and Mozaffari, S. (2019), 'Secure deep neural networks using adversarial image generation and training with noise-gan', *Computers Security* **86**, 372–387.
URL: *https://www.sciencedirect.com/science/article/pii/S016740481930121X*

He, K., Zhang, X., Ren, S. and Sun, J. (2016), Identity mappings in deep residual networks, *in* B. Leibe, J. Matas, N. Sebe and M. Welling, eds, 'Computer Vision – ECCV 2016', Springer International Publishing, Cham, pp. 630–645.

Ho, J., Jain, A. and Abbeel, P. (2020), 'Denoising diffusion probabilistic models'.

Hu, W. and Tan, Y. (2017), 'Generating adversarial malware examples for black-box attacks based on gan'.
URL: *https://arxiv.org/abs/1702.05983*

Ignatov, A., Timofte, R., Van Vu, T., Minh Luu, T., X Pham, T., Van Nguyen, C., Kim, Y., Choi, J.-S., Kim, M., Huang, J. et al. (2018), Pirm challenge on perceptual image enhancement on smartphones: Report, *in* 'Proceedings of the European Conference on Computer Vision (ECCV) Workshops', pp. 0–0.

Imran, A.-A.-Z. and Terzopoulos, D. (2021), *Multi-Adversarial Variational Autoencoder Nets for Simultaneous Image Generation and Classification*, Springer Singapore, Singapore, pp. 249–271.
URL: *https://doi.org/10.1007/978-981-15-6759-9*$_1$1

Jia, X., Wei, X., Cao, X. and Foroosh, H. (2019), Comdefend: An efficient image compression model to defend adversarial examples, *in* 'Proceedings of the IEEE/CVF conference on computer vision and pattern recognition', pp. 6084–6092.

Jiang, L., Qiao, K., Qin, R., Wang, L., Yu, W., Chen, J., Bu, H. and Yan, B. (2020), 'Cycle-consistent adversarial gan: The integration of adversarial attack and defense', *Security and Communication Networks* **2020**(1), 3608173.

Jiang, X., Niu, C., Ying, C., Wu, F. and Luo, Y. (2022), 'Pricing gan-based data generators under rényi differential privacy', *Information Sciences* **602**, 57–74.
URL: *https://www.sciencedirect.com/science/article/pii/S0020025522003723*

Kang, M., Tran, T. Q., Cho, S. and Kim, D. (2021), Cap-gan: Towards adversarial robustness with cycle-consistent attentional purification, *in* '2021 International Joint Conference on Neural Networks (IJCNN)', pp. 1–8.

Karras, T., Aila, T., Laine, S. and Lehtinen, J. (2018), 'Progressive growing of gans for improved quality, stability, and variation'.
URL: *https://arxiv.org/abs/1710.10196*

Kim, T., Cha, M., Kim, H., Lee, J. K. and Kim, J. (2017), 'Learning to discover cross-domain relations with generative adversarial networks'.
URL: *https://arxiv.org/abs/1703.05192*

Kingma, D. P. and Welling, M. (2022), 'Auto-encoding variational bayes'.
URL: *https://arxiv.org/abs/1312.6114*

Krizhevsky, A., Nair, V. and Hinton, G. (n.d.), 'Cifar-10 (canadian institute for advanced research)'.
URL: *http://www.cs.toronto.edu/ kriz/cifar.html*

Kurakin, A., Goodfellow, I. and Bengio, S. (2017), 'Adversarial examples in the physical world'.

Lal, S., Rehman, S. U., Shah, J. H., Meraj, T., Rauf, H. T., Damaševičius, R., Mohammed, M. A. and Abdulkareem, K. H. (2021), 'Adversarial attack and defence through adversarial training and feature fusion for diabetic retinopathy recognition', *Sensors* **21**(11), 3922.

Laykaviriyakul, P. and Phaisangittisagul, E. (2023), 'Collaborative defense-gan for protecting adversarial attacks on classification system', *Expert Systems with Applications* **214**, 118957.
URL: *https://www.sciencedirect.com/science/article/pii/S0957417422019753*

Li, H., Ye, Q., Hu, H., Li, J., Wang, L., Fang, C. and Shi, J. (2023), 3dfed: Adaptive and extensible framework for covert backdoor attack in federated learning, *in* '2023 IEEE Symposium on Security and Privacy (SP)', pp. 1893–1907.

Liao, F., Liang, M., Dong, Y., Pang, T., Hu, X. and Zhu, J. (2018), 'Defense against adversarial attacks using high-level representation guided denoiser'.

Lin, Z., Shi, Y. and Xue, Z. (2022), Idsgan: Generative adversarial networks for attack generation against intrusion detection, *in* J. Gama, T. Li, Y. Yu, E. Chen, Y. Zheng and F. Teng, eds, ‘Advances in Knowledge Discovery and Data Mining’, Springer International Publishing, Cham, pp. 79–91.

Liu, C., Chen, L.-C., Schroff, F., Adam, H., Hua, W., Yuille, A. L. and Fei-Fei, L. (2019), Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation, *in* ‘2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)’, pp. 82–92.

Liu, Z., Luo, P., Wang, X. and Tang, X. (2015), Deep learning face attributes in the wild, *in* ‘Proceedings of International Conference on Computer Vision (ICCV)’.

Lorenz, P., Durall, R. and Keuper, J. (2024), Adversarial examples are misaligned in diffusion model manifolds, *in* ‘2024 International Joint Conference on Neural Networks (IJCNN)’, pp. 1–8.

Ma, J., Liang, P., Yu, W., Chen, C., Guo, X., Wu, J. and Jiang, J. (2020), ‘Infrared and visible image fusion via detail preserving adversarial learning’, *Information Fusion* **54**, 85–98.
URL: *https://www.sciencedirect.com/science/article/pii/S1566253519300314*

Mahima, K. T., Perera, A. G., Anavatti, S. and Garratt, M. (2024), ‘Toward robust 3d perception for autonomous vehicles: A review of adversarial attacks and countermeasures’, *IEEE Transactions on Intelligent Transportation Systems* **25**(12), 19176–19202.

Maliakel, P. J., Ilager, S. and Brandic, I. (2024), ‘Fligan: Enhancing federated learning with incomplete data using gan’.
URL: *https://arxiv.org/abs/2403.16930*

Mirza, M. and Osindero, S. (2014), ‘Conditional generative adversarial nets’.
URL: *https://arxiv.org/abs/1411.1784*

Moon, S., An, G. and Song, H. O. (2022), ‘Preemptive image robustification for protecting users against man-in-the-middle adversarial attacks’, *Proceedings of the AAAI Conference on Artificial Intelligence* **36**(7), 7823–7830.
URL: *https://ojs.aaai.org/index.php/AAAI/article/view/20751*

Moosavi-Dezfooli, S.-M., Fawzi, A., Fawzi, O. and Frossard, P. (2017), ‘Universal adversarial perturbations’.
URL: *https://arxiv.org/abs/1610.08401*

Moosavi-Dezfooli, S.-M., Fawzi, A. and Frossard, P. (2016), ‘Deepfool: a simple and accurate method to fool deep neural networks’.

Mustafa, A., Khan, S. H., Hayat, M., Shen, J. and Shao, L. (2020), ‘Image super-resolution as a defense against adversarial attacks’, *IEEE Transactions on Image Processing* **29**, 1711–1724.
URL: *http://dx.doi.org/10.1109/TIP.2019.2940533*

Nadipally, M. (2019), Chapter 2 - optimization of methods for image-texture segmentation using ant colony optimization, *in* D. J. Hemanth, D. Gupta and V. Emilia Balas, eds, ‘Intelligent Data Analysis for Biomedical Applications’, Intelligent Data-Centric Systems, Academic Press, pp. 21–47.
URL: *https://www.sciencedirect.com/science/article/pii/B9780128155530000021*

Nie, W., Guo, B., Huang, Y., Xiao, C., Vahdat, A. and Anandkumar, A. (2022), ‘Diffusion models for adversarial purification’.
URL: *https://arxiv.org/abs/2205.07460*

Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z. B. and Swami, A. (2016), The limitations of deep learning in adversarial settings, *in* ‘2016 IEEE European Symposium on Security and Privacy (EuroSP)’, pp. 372–387.

Pitropakis, N., Panaousis, E., Giannetsos, T., Anastasiadis, E. and Loukas, G. (2019), ‘A taxonomy and survey of attacks against machine learning’, *Computer Science Review* **34**, 100199.
URL: *https://www.sciencedirect.com/science/article/pii/S1574013718303289*

Qayyum, A., Usama, M., Qadir, J. and Al-Fuqaha, A. (2020), ‘Securing connected autonomous vehicles: Challenges posed by adversarial machine learning and the way forward’, *IEEE Communications Surveys Tutorials* **22**(2), 998–1026.

Qin, H., Fu, Y., Zhang, H., El-Yacoubi, M. A., Gao, X., Song, Q. and Wang, J. (2024), ‘Msmemorygan: A multi-scale memory gan for palm-vein adversarial purification’.
URL: *https://arxiv.org/abs/2408.10694*

Qiu, S., Liu, Q., Zhou, S. and Wu, C. (2019), ‘Review of artificial intelligence adversarial attack and defense technologies’, *Applied Sciences* **9**(5).
URL: *https://www.mdpi.com/2076-3417/9/5/909*

Queyrut, S., Schiavoni, V. and Felber, P. (2023), Mitigating adversarial attacks in federated learning with trusted execution environments, *in* '2023 IEEE 43rd International Conference on Distributed Computing Systems (ICDCS)', pp. 626–637.

Quiring, E., Arp, D. and Rieck, K. (2017), 'Fraternal twins: Unifying attacks on machine learning and digital watermarking'.

Rezaei, B., Ghanbari, H. and Enayatifar, R. (2023), 'An image encryption approach using tuned henon chaotic map and evolutionary algorithm', *Nonlinear Dynamics* **111**(10), 9629–9647.
URL: *https://doi.org/10.1007/s11071-023-08331-y*

Saharia, C., Ho, J., Chan, W., Salimans, T., Fleet, D. J. and Norouzi, M. (2023), 'Image super-resolution via iterative refinement', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **45**(4), 4713–4726.

Samangouei, P., Kabkab, M. and Chellappa, R. (2018), 'Defense-gan: Protecting classifiers against adversarial attacks using generative models'.

Santhanam, G. K. and Grnarova, P. (2018), 'Defending against adversarial attacks by leveraging an entire gan'.
URL: *https://arxiv.org/abs/1805.10652*

Sara, U., Akter, M. and Uddin, M. S. (2019), 'Image quality assessment through fsim, ssim, mse and psnr—a comparative study', *Journal of Computer and Communications* **7**(3), 8–18.

Sarkar, S., Bansal, A., Mahbub, U. and Chellappa, R. (2017), 'Upset and angri : Breaking high performance image classifiers'.
URL: *https://arxiv.org/abs/1707.01159*

Saunders, M., Lewis, P. and Thornhill, A. (2009), *Research Methods for Business Students*, Always learning, Prentice Hall.
URL: *https://books.google.lk/books?id=u-txtfaCFiEC*

Shi, C., Holtz, C. and Mishne, G. (2021), 'Online adversarial purification based on self-supervision'.
URL: *https://arxiv.org/abs/2101.09387*

Song, Y., Kim, T., Nowozin, S., Ermon, S. and Kushman, N. (2018), 'Pixeldefend: Leveraging generative models to understand and defend against adversarial examples'.

Su, J., Vargas, D. V. and Sakurai, K. (2019), 'One pixel attack for fooling deep neural networks', *IEEE Transactions on Evolutionary Computation* **23**(5), 828–841.
URL: *http://dx.doi.org/10.1109/TEVC.2019.2890858*

Sun, H., Wu, S. and Ma, L. (2024), 'Adversarial attacks on gan-based image fusion', *Information Fusion* **108**, 102389.
URL: *https://www.sciencedirect.com/science/article/pii/S1566253524001672*

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. and Fergus, R. (2014), 'Intriguing properties of neural networks'.
URL: *https://arxiv.org/abs/1312.6199*

Tabacof, P., Tavares, J. and Valle, E. (2016), 'Adversarial images for variational autoencoders', *CoRR* **abs/1612.00155**.
URL: *http://arxiv.org/abs/1612.00155*

Tancik, M., Mildenhall, B. and Ng, R. (2020), Stegastamp: Invisible hyperlinks in physical photographs, *in* '2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)', pp. 2114–2123.

Usama, M., Asim, M., Latif, S., Qadir, J. and Ala-Al-Fuqaha (2019), Generative adversarial networks for launching and thwarting adversarial attacks on network intrusion detection systems, *in* '2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)', pp. 78–83.

Walgama, R. and Mahima, K. Y. (2024), Fl-cyclegan: Enhancing mobile photography with federated learning-enabled cyclegan, *in* '2024 Moratuwa Engineering Research Conference (MERCon)', pp. 688–693.

Wang, D., Jin, W., Wu, Y. and Khan, A. (2021), 'Improving global adversarial robustness generalization with adversarially trained gan'.
URL: *https://arxiv.org/abs/2103.04513*

Wang, X., Mei, S., Lian, J. and Lu, Y. (2024), 'Fooling aerial detectors by background attack via dual-adversarial-induced error identification', *IEEE Transactions on Geoscience and Remote Sensing* .

Xiao, C., Chen, Z., Jin, K., Wang, J., Nie, W., Liu, M., Anandkumar, A., Li, B. and Song, D. (2022), 'Densepure: Understanding diffusion models towards adversarial robustness'.
URL: *https://arxiv.org/abs/2211.00322*

Xie, C., Wang, J., Zhang, Z., Ren, Z. and Yuille, A. (2018), 'Mitigating adversarial effects through randomization'.

Xin, B., Yang, W., Geng, Y., Chen, S., Wang, S. and Huang, L. (2020), Private fl-gan: Differential privacy synthetic data generation based on federated learning, *in* 'ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)', pp. 2927–2931.

Xu, C., Ren, J., Zhang, D., Zhang, Y., Qin, Z. and Ren, K. (2019), 'Ganobfuscator: Mitigating information leakage under gan via differential privacy', *IEEE Transactions on Information Forensics and Security* **14**(9), 2358–2371.

Xu, W., Evans, D. and Qi, Y. (2018), Feature squeezing: Detecting adversarial examples in deep neural networks, *in* 'Proceedings 2018 Network and Distributed System Security Symposium', NDSS 2018, Internet Society.
URL: *http://dx.doi.org/10.14722/ndss.2018.23198*

Yang, L., Zhang, Z., Song, Y., Hong, S., Xu, R., Zhao, Y., Zhang, W., Cui, B. and Yang, M.-H. (2023), 'Diffusion models: A comprehensive survey of methods and applications', *ACM Comput. Surv.* **56**(4).
URL: *https://doi.org/10.1145/3626235*

Yoon, J., Jordon, J. and van der Schaar, M. (2019), PATE-GAN: Generating synthetic data with differential privacy guarantees, *in* 'International Conference on Learning Representations'.
URL: *https://openreview.net/forum?id=S1zk9iRqF7*

Yu, F., Wang, L., Fang, X. and Zhang, Y. (2020), 'The defense of adversarial example with conditional generative adversarial networks', *Security and Communication Networks* **2020**(1), 3932584.
URL: *https://onlinelibrary.wiley.com/doi/abs/10.1155/2020/3932584*

Yu, W., Xu, Y. and Ghamisi, P. (2024), 'Universal adversarial defense in remote sensing based on pre-trained denoising diffusion models', *International Journal of Applied Earth Observation*

and Geoinformation **133**, 104131.
URL: *https://www.sciencedirect.com/science/article/pii/S1569843224004850*

Zeng, Y., Dai, T., Chen, B., Xia, S.-T. and Lu, J. (2021), 'Correlation-based structural dropout for convolutional neural networks', *Pattern Recognition* **120**, 108117.
URL: *https://www.sciencedirect.com/science/article/pii/S0031320321003046*

Zhang, B., Gu, S., Zhang, B., Bao, J., Chen, D., Wen, F., Wang, Y. and Guo, B. (2022), Styleswin: Transformer-based gan for high-resolution image generation, *in* 'Proceedings of the IEEE/CVF conference on computer vision and pattern recognition', pp. 11304–11314.

Zhang, C., Yu, S., Tian, Z. and Yu, J. J. Q. (2023), 'Generative adversarial networks: A survey on attack and defense perspective', *ACM Comput. Surv.* **56**(4).
URL: *https://doi.org/10.1145/3615336*

Zhang, S., Gao, H. and Rao, Q. (2021), 'Defense against adversarial attacks by reconstructing images', *IEEE Transactions on Image Processing* **30**, 6117–6129.

Zhao, G., Zhang, M., Liu, J., Li, Y. and Wen, J.-R. (2022), 'Ap-gan: Adversarial patch attack on content-based image retrieval systems', *GeoInformatica* **26**(2), 347–377.
URL: *https://doi.org/10.1007/s10707-020-00418-7*

Zhao, G., Zhang, M., Liu, J. and Wen, J.-R. (2019), 'Unsupervised adversarial attacks on deep feature-based retrieval with gan'.
URL: *https://arxiv.org/abs/1907.05793*

Zhao, S., Li, J., Wang, J., Zhang, Z., Zhu, L. and Zhang, Y. (2021), 'attackgan: Adversarial attack against black-box ids using generative adversarial networks', *Procedia Computer Science* **187**, 128–133. 2020 International Conference on Identification, Information and Knowledge in the Internet of Things, IIKI2020.
URL: *https://www.sciencedirect.com/science/article/pii/S1877050921009303*

Zhou, B., Lapedriza, A., Khosla, A., Oliva, A. and Torralba, A. (2017), 'Places: A 10 million image database for scene recognition', *IEEE Transactions on Pattern Analysis and Machine Intelligence* .

Zhu, J.-Y., Park, T., Isola, P. and Efros, A. A. (2020), ‘Unpaired image-to-image translation using cycle-consistent adversarial networks’.
URL: *https://arxiv.org/abs/1703.10593*