

# Defence for the Adversarial Attack on Object Detection

Project report submitted in partial fulfillment  
of the requirements for the degree of

*Bachelor of Technology*  
*in*  
*Computer Science Engineering*

by

Suryansh Bhardwaj - 20UCC108  
Madhav Maheshwari - 20UCC061  
Sankalp Jain - 20UCS172  
Arunabh Agarwal - 20UEC032

Under Guidance of  
Dr. Indra Deep Mastan



Department of Computer Science Engineering  
The LNM Institute of Information Technology, Jaipur

November 2023



The LNM Institute of Information Technology  
Jaipur, India

**CERTIFICATE**

This is to certify that the project entitled “Defence for the Adversarial Attack on Object Detection” , submitted by Suryansh Bhardwaj(20UCC108), Madhav Maheshwari(20UCC061), Sankalp Jain(20UCS172) and Arunabh Agarwal(20UEC032) in partial fulfillment of the requirement of degree in Bachelor of Technology (B. Tech), is a bonafide record of work carried out by them at the Department of Computer Science Engineering, The LNM Institute of Information Technology, Jaipur, (Rajasthan) India, during the academic session 2023-2024 under my supervision and guidance and the same has not been submitted elsewhere for award of any other degree. In my/our opinion, this report is of standard required for the award of the degree of Bachelor of Technology (B. Tech).

---

Date

---

Adviser: Dr. Indra Deep Mastan

# Acknowledgments

We would like to express our sincere gratitude to the following individuals and organizations for their invaluable support and assistance in the completion of this project:

- Our supervisor, Dr. Indra Deep Mastan, for his immense guidance, expertise, and continuous encouragement throughout the project.
- Our batchmates, seniors and juniors who provided valuable insights, feedback, and moral support.
- LNM Institute of Information Technology for academic support and resources.

We are truly thankful for the help and encouragement we received, which significantly contributed to the successful completion of this project.

# Abstract

Object detection systems act as eyes of machines, helping them to understand their surroundings. Object detection models have wide range of use cases, the most common use case is the use in autonomous vehicles. However, recent research has revealed that these object detection models can be easily mislead by adversarial attacks. Adversarial attacks involve making small changes to the object that is to be detected by adding a small patch on it or adding a small amount of noise to it. This project aims to address this issue and work on different attack and defences to get the best results for object detection model even after suffering adversarial attack.

Our primary focus while working on this project was to improve the existing defence strategies. We selected two attacks and two defence methods for this project. The attacks being - FGSM(Fast Gradient Sign Method) and DeepFool, and the defences being DIP Defend and ZS-N2N.

We will achieve our aim by using the technique of adversarial training and thus exposing our model to adversarial attacks. We think that if the model understands the process of adversarial attack then it will be easy for us to train it against them.

The project consists of three main components:

1. **Understanding Adversarial Attacks:** Forecast of the nature of the threat is needed so that we can develop effective defense mechanisms. Let's start by analyzing various adversarial attacks, such as DeepFool and FGSM. The result of this knowledge will be key for creating defenses against these particular attacks.
2. **Strengthening Defense Strategies:** Using DIP Defend and Noise-to-Noise, which are popular existing defence patterns and we now make them more robust and flexible to object detection models.
3. **Adversarial Training:** Adversarial Training is the very core basis of our defence strategy. We incorporate adversarial examples into our training process, with an aim to give object identification models the ability to allow hostile and indeed strong perturbations.

The success of this project would be finely calculated by the performance of the object detection models. The aim is to have high accuracy even when exposed to these adversarial attacks, keeping least possible impact on their execution, non-adversarial cases.

# Contents

<b>List of Figures</b>	<b>viii</b>
<b>List of Tables</b>	<b>ix</b>
<b>1 Introduction</b>	<b>1</b>
1.1 The Area of Work . . . . .	1
1.2 Problem Addressed . . . . .	2
1.3 Existing System . . . . .	2
1.3.1 Dataset: PASCAL Visual Object Classes (VOC) . . . . .	2
1.3.2 Object Detection model: Residual Network with 34 Layers or ResNet-34	3
<b>2 Literature Review</b>	<b>4</b>
2.1 FGSM . . . . .	4
2.2 DeepFool . . . . .	5
2.3 Deep Image Prior Defense . . . . .	6
2.4 Zero-Shot Noise2Noise . . . . .	9
<b>3 Proposed Work</b>	<b>11</b>
3.1 FGSM . . . . .	11
3.2 DeepFool . . . . .	12
3.2.1 DeepFool Algorithm . . . . .	12
3.2.2 Perturbation Computation . . . . .	12
3.2.2.1 Explanation of Euclidean Norm . . . . .	13
3.2.3 Image Update . . . . .	13
3.2.4 Optimization Loop . . . . .	13
3.2.5 DeepFool for Binary Classifiers . . . . .	14
3.2.5.1 Explanation . . . . .	14
3.3 DIPDefend . . . . .	14
3.4 Zero-Shot Noise2Noise . . . . .	16
<b>4 Simulation and Results</b>	<b>19</b>
4.1 FGSM . . . . .	19
4.2 DeepFool . . . . .	21
4.3 DIP Defend . . . . .	23
4.4 Zero-Shot Noise2Noise . . . . .	26
4.5 Tabular Comparision . . . . .	29
<b>5 Conclusions and Future Work</b>	<b>30</b>

**Bibliography**

**31**

# List of Figures

2.1	FGSM . . . . .	4
2.2	DeepFool . . . . .	5
2.3	Plotting the smoothed PSNR curves (SES-PSNR) allows us to identify the ideal stopping iteration (vertical line). It is evident that SES is capable of precisely identifying the anticipated turning point. . . . .	8
2.4	Image Denoising . . . . .	9
3.1	In the above example the input is a $4 \times 4$ image, and the output is two $2 \times 2$ images . . . . .	16
4.1	Accuracy vs Epsilon . . . . .	19



# List of Tables

4.1	FGSM Model Results . . . . .	20
4.2	DeepFool Model Results . . . . .	22
4.3	FGSM attacked image defended through DIP Defend . . . . .	24
4.4	DeepFool attacked image defended through DIP Defend . . . . .	25
4.5	FGSM attacked image defended through Zero-Shot Noise2Noise . . . . .	27
4.6	DeepFool attacked image defended through Zero-Shot Noise2Noise . . . . .	28
4.7	Comparision between the predictions of different attack and defence models as compared to the Orignal image Note: The attacked images were passed through the defence models to get the output. Also this is just a test dataset inspired from PASCAL VOC . . . . .	29

# Chapter 1

## Introduction

### 1.1 The Area of Work

We are working in the field of object detection and computer vision. When we talk about adversarial attack they are of mainly two types noised based and patch based, in this project we have focused on noise based adversarial attack. In this also there are two sub-types targeted and untargeted and we have focused on untargeted noise based adversarial attack.

While working on this project we characterised the adversarial attacks into 2 main parts weak adversarial attack in our case FGSM[1] and strong adversarial attack in our case DeepFool[2]. FGSM is being characterised as a weak attack because when additive noise is added to the image the changes are significant and can be visible by naked eyes. Whereas the working of DeepFool is completely different it adds very less amount of noise to the image until it is being detected as an object of the nearest class to the current object. The changes made in the attacked image are so minute that they can't be noticed by naked eyes.

In DIP Defend[3][4], we demonstrate that the structure of a generator network is sufficient to capture a great deal of low-level image statistics before any learning. To do so, we show that a randomly initialized neural network can be used as a prior that gives excellent results in standard problems such as denoising, super-resolution and inpainting.

ZS-N2N, or Zero Shot Noise2Noise[5], is a novel image denoising approach that defies standard training rules by utilising a single noisy image. ZS-N2N, inspired by Noise2Noise and Neighbor2Neighbor, takes a novel technique to denoising by decomposing noisy images into downsampled pairs. ZS-N2N provides high-quality picture denoising using a lightweight network and explicit regularisation, demonstrating its efficiency and effectiveness in circumstances with limited data availability and processing resources.

## 1.2 Problem Addressed

The Problem we addressed is to show that we can have object detection models that can sustain the various adversarial attacks[6][7] and how we could keep our Data protected. Adversarial attacks aim is to make the model misclassify the input data and perturb the Input data to fool AI/ML models but we still can build upon Robust and Effective Object Detection Models using Adversarial Defences Too. We used PASCAL VOC[8] training data-set to enhance the model's resilience against potential attacks. Aim is to create models that are not just accurate but also resilient against adversarial manipulation and to ensure integrity and trustworthiness of AI systems in various domains. Deepfool , Adversarial Attack[9][10] aims to create the most minimal perturbations to an image to deceive the model. These attacks can also affect the way in which interpretations are made. These attacks are Required to constantly check and increase the robustness of the system. Whereas FGSM is Relatively simple to implement and computationally more efficient.

## 1.3 Existing System

We have used two existing systems namely PASCAL Visual Object Classes (VOC) dataset and ResNet-34 or Residual Network with 34 Layers.

### 1.3.1 Dataset: PASCAL Visual Object Classes (VOC)

The Pascal VOC (Visual Object Classes) benchmark dataset is well-known and frequently used in computer vision applications. The dataset was produced by the University of Oxford's Visual Geometry Group and is a valuable tool for testing and improving object recognition systems. The reason for its prominence is that it is all-inclusive, covering a wide range of object types and intricate scenarios.

Photos are gathered from many sources, including people, cars, and aeroplanes, depict commonplace scenarios with objects belonging to 20 different classes. These photos make up the Pascal VOC dataset. Bounding boxes and segmentation masks are used to annotate each image, defining the exact borders of the objects of interest. Because of its thorough annotation, Pascal VOC is a great help to researchers and practitioners in the computer vision community for both segmentation and object localization tasks.

Being a common baseline for assessing how well object detection and segmentation algorithms perform is one of the dataset's main advantages. The yearly Pascal VOC competitions have developed into a cutting-edge venue for academics to compete and present their work, encouraging an innovative atmosphere and pushing the boundaries of computer vision. With several annual releases, the dataset's lifespan guarantees its relevance throughout time, facilitating ongoing advancements in the creation of reliable and adaptable computer vision models.

Because Pascal VOC strikes a balance between complexity and diversity, researchers frequently utilise it as a starting point for training and testing models. Because of its balance, the dataset is a trustworthy metric for evaluating how well algorithms can generalise across different circumstances and object classes. This helps to prevent biases.

To sum up, the Pascal VOC dataset is a major contribution to the field of computer vision, having helped to develop object identification, segmentation, and recognition. Its extensive annotations, wide range of information, and benchmarking function make it a valuable asset that propels advancement in the rapidly changing field of visual recognition technologies.

### **1.3.2 Object Detection model: Residual Network with 34 Layers or ResNet-34**

Residual Network with 34 Layers, or ResNet-34, is a significant advancement in the field of deep convolutional neural networks (CNNs). ResNet was created by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun in order to overcome the difficulties associated with training extremely deep networks. These difficulties were caused by vanishing gradients and performance loss as network depth rose with conventional architectures.

ResNet-34 stands out for its creative application of residual learning, a deep learning paradigm-shifting idea. ResNet adds residual blocks in place of trying to learn the underlying mapping directly. These blocks have shortcut connections that let the network bypass one or more layers, improving information flow and lowering the possibility of disappearing gradients. This method allows for the training of very deep networks, which improves the accuracy of the models.

Convolutional layers, batch normalisation, and non-linear activation functions like rectified linear units (ReLU) are among the 34 layers that make up ResNet-34. since of its architecture, the network is very good at tasks like object detection, image classification, and image recognition since it can extract complex properties from input data. The 34-layer depth is widely used in a variety of computer vision applications because it strikes a balance between expressive power and computational efficiency.

ResNet-34's success has extended beyond its initial goal, impacting later models and setting the standard for deep learning research. Its influence goes beyond computer vision; it spurs developments in other fields such as natural language processing. Scholars and practitioners are still using the concepts introduced by ResNet-34 to push the limits of what is possible in deep learning. All things considered, ResNet-34 is a monument to the strength of creativity in neural network architecture, opening up new avenues and clearing the path for more advanced and successful deep learning models.

## Chapter 2

# Literature Review

### 2.1 FGSM

FGSM(Fast Gradient Sign Method) is an adversarial attack algorithm used to generate adversarial examples for deep learning models. The Main Idea is to add a small perturbation to the input data to misclassify it. In FGSM we add large amount of noise to the input image to perform the adversarial attack. Talking about the main idea of FGSM, it is basically a method to intelligently generate adversarial examples. FGSM uses the gradient information of the loss function with respect to the input to perturb the input data in the direction that maximizes the loss, resulting in misclassification. Here By giving the perturbation a value that most nearly resembles the weight vector  $w$ , it is possible to maximize the perturbation that results in an inaccurate prediction. Weight vector refers to a vector of numerical values that are used to represent features of an image.

In an adversarial attack scenario, the value of epsilon  $\epsilon$  indicates the magnitude of the disturbance added to the input image. It is usually a small value. The perturbations should be small enough so that the human eye cannot distinguish the adversarial image from the original image. The role of epsilon is the most important in the process of this attack as varying this quantity , we can obtain various false images of the same orinigal image.

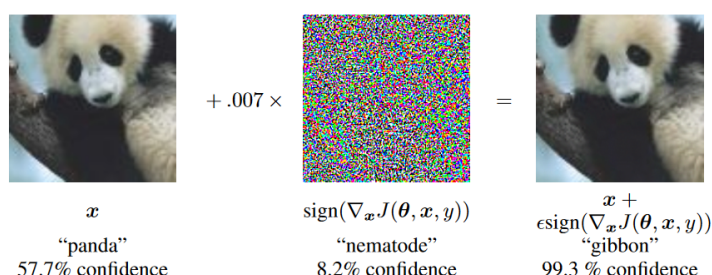


FIGURE 2.1: FGSM

With regard to the input data, FGSM makes use of the gradient information of the loss function. What we observe is that Larger confidence rate , Larger is the perturbation to fool My Network , to Fool My Object Detection Model. The number of dimensions in the problem can lead the perturbation's change in activation to increase linearly, allowing multiple minor input changes to combine into a single significant output change. The final Aim is to basically minimize  $n$  so that we don't lose the main meaning of the image.

The Paper reveals a basic conflict between the simplicity of training of linear models and the requirement for nonlinearity to withstand adversarial attacks. The relevance of FGSM in comprehending weaknesses and enhancing model resilience against adversarial perturbations is highlighted in the paper.

## 2.2 DeepFool

DeepFool is an adversarial attack algorithm introduced in the paper "DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks" by Seyed-Mohsen Moosavi-Dezfooli, Al-hussein Fawzi, and Pascal Frossard, which was presented at the IEEE Conference on Computer Vision and Pattern Recognition in 2016. The paper's primary focus is on creating perturbations that can efficiently fool deep neural networks (DNNs).

The authors begin by emphasising DNNs' vulnerability to adversarial instances, which are carefully prepared input samples designed to lead the network astray and cause it to make inaccurate predictions. The DeepFool algorithm was created with the goal of developing a mechanism that is both simple and effective for creating adversarial perturbations.

DeepFool is intended to compute the smallest amount of perturbation required to misclassify an input sample. The technique iteratively estimates the direction in which the input needs be altered in order to approach the neural network's decision boundary. It computes the perturbation by estimating the decision boundary's linear approximation and updating the input in the direction that minimises the distance to this boundary.



FIGURE 2.2: DeepFool

DeepFool's efficiency in creating adversarial perturbations is a crucial advantage. The authors show that DeepFool requires far less iterations than existing approaches, making it a computationally appealing option for creating hostile instances. Furthermore, the algorithm's simplicity contributes to its usefulness and ease of implementation.

The study gives a theoretical analysis of the algorithm, demonstrating its convergence features and elucidating the relationship between the perturbation and the network's decision boundaries. The authors also propose a metric termed "adversarial distance" to quantify DeepFool's success in creating adversarial samples.

To assess DeepFool's performance, the authors run tests on numerous benchmark datasets and compare the results to existing adversarial attack strategies. They show that DeepFool has a high success rate in creating adversarial cases with few perturbations. The algorithm's effectiveness is demonstrated across multiple neural network designs, demonstrating its adaptability.

The authors look into the transferability of adversarial instances generated by DeepFool further. The ability of adversarial examples designed for one model to trick other models is referred to as transferability. DeepFool has high transferability, allowing the created adversarial perturbations to be successful across multiple neural network designs.

In summary, the study introduces DeepFool, a strong and efficient technique for creating adversarial instances that may be used to mislead deep neural networks. The method's simplicity and computational efficiency, as well as its good performance across multiple datasets and architectures, make it an important contribution to the field of adversarial attacks. The theoretical analysis offered in the study adds to our understanding of the algorithm's behaviour, and the experiments demonstrate its practical usefulness in real-world circumstances where neural network security and robustness are critical.

## 2.3 Deep Image Prior Defense

Before studying the DIPDefense let us see the dip model first, we demonstrate that generator networks structure is capable enough to deal with the low level images statistics before any learning. In order to do this, we demonstrate that a randomly initialised neural network may be utilised as a prior that produces outstanding outcomes in common issues including inpainting, denoising, and super-resolution.

The main idea in this is that basic low-level image statistics are intrinsically captured by the architectural characteristics of a neural network, even in the absence of any particular training on a dataset. The paper takes advantage of this to show how a randomly initialised neural network can function as an efficient handdesigned prior for tasks like super-resolution, inpainting, and image denoising.

**Context and Motivation:** Deep neural networks are trained on massive datasets to discover complex patterns and characteristics as part of the traditional method for restoring images. Nevertheless, this procedure demands a large amount of labelled data and is computationally costly. By arguing that key picture priors can be captured by a randomly initialised neural network's inherent structure without the requirement for deliberate training, Ulyanov challenges this paradigm.

**Denoising:** Image denoising is one of the main applications that are examined in the study. Paper shows how to efficiently remove noise from photos using a neural network with a randomly initialized architecture. The architecture presents a viable alternative to conventional denoising approaches that mainly rely on extensive training on noisy-clean image pairs, as it can discern between signal and noise thanks to its grasp of low-level image statistics. The work also explores the area of image inpainting, which is the process of repairing image damaged part. The paper demonstrates how a neural network's intrinsic structure may produce logical and believable replacements for an image's missing portions. This is a very useful contribution because inpainting is a difficult computer vision problem with applications in image modification and restoration. In this, we see the pixel that are model knows and will enhance the pixel and will not touch the unknown part.

**Super-Resolution:** The method of creating high-resolution images from low-resolution inputs is known as super-resolution, and it is one of the important applications that is examined in this study. Ulyanov shows that high-quality super-resolved images can be generated by the randomly initialised neural network without requiring pre-training on high-resolution examples. This refutes the widely held notion that large datasets are necessary for super-resolution models to acquire realistic image priors. **Experimental Validation:** To validate the suggested approach, a full set of experiments is provided in the study. By comparing the performance of the randomly initialised neural network with classically trained models, Ulyanov carries out experiments on a variety of datasets. The outcomes demonstrate the deep image prior's effectiveness in obtaining results in denoising, inpainting, and super-resolution tasks that are on par with or better than competitors. Now let us see the DIPDefense model,

This method is adapted to various inputs by creating an halting strategy. The defense method developed till DIP were still relying on external prior heavily learned from the training dataset while neglecting the internal prior within the input. In DipDefend we exploit the internal priors from a single adversarial input itself.

The suggested defense strategy is an attack- and model-agnostic, simple yet effective mechanism. It takes advantage of the built-in priors in a target image and uses a reparameterized generative network to lessen the effect of adversarial perturbations. These are the method's salient features:

**Reparameterized Generative Network:**



Reparameterization techniques are integrated into a generative network, which is used in the defense strategy. Reparameterization strengthens the model's resilience by lessening the effects of adversarial perturbations. From the standpoint of feature learning:

The defense strategy shows a significant learning preference, especially when used to Deep Image Prior (DIP). When learning features for patterns that are resilient to disturbances, it usually gives priority to robust feature learning rather than non-robust feature which are prone to disturbances .

Modified Stopping Technique:

A strategy for adaptive halting is developed so that the system can be adjusted to different images. By using second-order exponential smoothing, this tactic enables the defense mechanism to adapt and change dynamically in response to various aspects of the image.

Empirical Validation and Effectiveness Demonstration: The suggested approach is demonstrated to be more effective than cutting-edge defense strategies in a range of adversarial perturbation types and intensities. This empirical data emphasizes how strong and efficient the suggested defense mechanism is.

We proposed a two-stage learning strategy: 1) robust feature learning for patterns that are not sensitive to perturbations at an early stage, and 2) non-robust feature learning for patterns that are sensitive to perturbations at a later stage. We also create a flexible stopping plan using second-order exponential smoothing to stop overfitting.

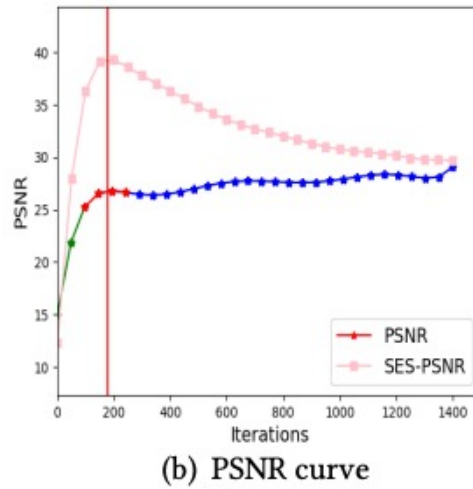


FIGURE 2.3: Plotting the smoothed PSNR curves (SES-PSNR) allows us to identify the ideal stopping iteration (vertical line). It is evident that SES is capable of precisely identifying the anticipated turning point.

## 2.4 Zero-Shot Noise2Noise

The Zero-Shot Noise2Noise method introduced in this study is a novel approach to picture denoising that is distinguished by its efficiency and data independence. This unique technique is inspired by Noise2Fast but differs in that it incorporates a consistency loss, eliminates the requirement for early pausing, and uses a considerably smaller and faster network. The ultimate goal is to attain competitive denoising quality without relying on large training datasets.

When compared to Noise2Fast, the proposed technique offers several enhancements. While Noise2Fast requires an early stopping criterion since it uses a somewhat large network, Zero-Shot Noise2Noise streamlines the process by applying a consistency loss, allowing for smoother training without early stopping. Furthermore, it uses a much smaller network that is more computationally efficient and faster in terms of forward pass computations. The proposed network is twelve times smaller than Noise2Fast, and a forward pass through it is seven times faster. Because of these developments, the Zero-Shot Noise2Noise approach is now an appealing option for applications with computational constraints.

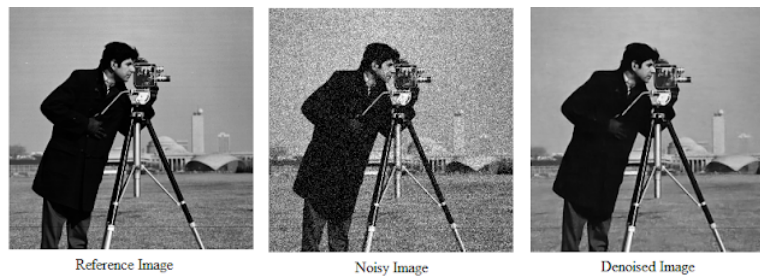


FIGURE 2.4: Image Denoising

When compared to traditional denoising techniques, the method's efficiency and computational savings are especially apparent. Traditional approaches, such as BM3D and Anscombe, require noise level knowledge as an input, but neural network-based solutions, such as DIP and Self2Self, which use UNet or variants as backbone networks, can be computationally costly. In contrast, Zero-Shot Noise2Noise distinguishes itself by developing a shallow and simple network with few parameters, making it ideal for applications with limited computational resources.

In addition, the study includes a brief description of several denoising methods, such as blind spot networks, probabilistic variations, and those based on Stein's unbiased risk estimator. It emphasises the significance of a general framework that may be applied to varied noise distributions.

In terms of performance, Zero-Shot Noise2Noise outperforms Noise2Fast on grayscale photos, yielding higher-quality output. This enhancement is due to the retention of all information while downsampling, which sets it apart from Noise2Fast, which drops pixel values during this process. Thus, the suggested method not only produces competitive denoising scores but also improves image visual quality.

In conclusion, the Zero-Shot Noise2Noise approach provides substantial advances in picture denoising by providing an efficient and data-independent solution. Its competitive denoising quality is enhanced by its quicker training procedure, reduced network size, and information preservation while downsampling. This methodology makes an important contribution to the broader landscape of denoising approaches, demonstrating the potential for attaining high-quality outcomes without the need for complex algorithms.

## Chapter 3

# Proposed Work

### 3.1 FGSM

The perturbation added to the input data  $x$  to create the adversarial example is represented by  $\eta$ :

$$\eta = \epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y))$$

Here,

- $\eta$ : Perturbation added to the input data  $x$ .
- $\epsilon$ : Small constant controlling the magnitude of the perturbation.
- $\nabla_x J(\theta, x, y)$ : Gradient of the loss function  $J$  with respect to the input  $x$  at the point defined by the model's parameters  $\theta$  and the true label  $y$ .
- $\text{sign}(\cdot)$ : Function extracting the sign of its argument.

The Fast Gradient Sign Method (FGSM) formula:

$$\eta = \epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y))$$

The equation  $w^T \tilde{x} = w^T x + w^T \eta$ :

$$w^T \tilde{x} = w^T x + w^T \eta$$

The adversarial image  $X'$  is obtained by adding the noise  $n$  to the original image  $X$ :

$$X' = X + n$$

For generating the perturbation  $\epsilon$ , the sign of the gradient is used:

$$\epsilon \cdot \text{sign}(\nabla_x J(\theta, x, y))$$

The attack involves perturbing the input  $x$  to create the adversarial example  $X'$ , which is then fed into the neural network model for classification.

The dot product between a weight vector  $w$  and an adversarial example  $x$  is also utilized in this attack.

In machine learning, the loss function  $J(\theta, x, y)$  measures the degree to which a model's predictions agree with the actual (true) values in the dataset. It represents the difference between expected and actual values, helping the model learn from its errors during the training process.

The Fast Gradient Sign Method (FGSM) uses the gradient information of the loss function with respect to the input data. The direction to perturb the input data is determined by the formula involving the gradient of the loss function. Adversarial examples are produced by FGSM by making slight adjustments in the direction that maximizes the loss.

## 3.2 DeepFool

### 3.2.1 DeepFool Algorithm

The DeepFool algorithm iteratively computes perturbations in the direction of the decision boundary until the image is misclassified.

#### 3.2.2 Perturbation Computation

The perturbation  $r_i$  at each iteration is computed as follows:

$$r_i = \frac{(\text{pert} + 1e - 4) \cdot \mathbf{w}}{\|\mathbf{w}\|}$$

where:

$\text{pert}$  : Current perturbation value,

$\mathbf{w}$  : Difference between gradients of the current class and other classes,

$\|\mathbf{w}\|$  : Euclidean norm of  $\mathbf{w}$ .

### 3.2.2.1 Explanation of Euclidean Norm

The Euclidean norm, also known as the  $L^2$  norm, is a measure of the magnitude or length of a vector in Euclidean space. For a vector  $\mathbf{v}$  with components  $v_1, v_2, \dots, v_n$ , the Euclidean norm is defined as:

$$\|\mathbf{v}\|_2 = \sqrt{v_1^2 + v_2^2 + \dots + v_n^2} \quad (3.1)$$

Here:

- $\mathbf{v}$  is the vector.
- $\|\mathbf{v}\|_2$  denotes the Euclidean norm of  $\mathbf{v}$ .
- $v_1, v_2, \dots, v_n$  are the components of the vector.
- The square of each component is summed, and the square root of the sum gives the Euclidean norm.

The Euclidean norm is a fundamental concept in linear algebra and is widely used in various mathematical and computational applications.

### 3.2.3 Image Update

The perturbed image is updated using the formula:

$$\text{pert\_image} = \text{image} + (1 + \text{overshoot}) \cdot \mathbf{r}_{\text{tot}}$$

where:

- image : Original input image tensor,
- overshoot : A small constant to ensure slight overshoot,
- $\mathbf{r}_{\text{tot}}$  : Cumulative perturbation vector.

### 3.2.4 Optimization Loop

The algorithm continues iterating until the image is misclassified ( $k_i \neq \text{label}$ ) or the maximum number of iterations (`max_iter`) is reached.

### 3.2.5 DeepFool for Binary Classifiers

The DeepFool algorithm is used for crafting adversarial perturbations for binary classifiers. The goal is to find the smallest perturbation needed to change the classification of an input image.

---

#### Algorithm 1 DeepFool for Binary Classifiers

---

```

1: Input: Image  $x$ , classifier  $f$ .
2: Output: Perturbation  $\hat{r}$ .
3: Initialize  $x_0 \leftarrow x$ ,  $i \leftarrow 0$ .
4: while  $\text{sign}(f(x_i)) = \text{sign}(f(x_0))$  do
5:    $r_i \leftarrow -\frac{f(x_i)}{\|\nabla f(x_i)\|_2^2} \nabla f(x_i)$ 
6:    $x_{i+1} \leftarrow x_i + r_i$ 
7:    $i \leftarrow i + 1$ 
8: end while
9: Return  $\hat{r} = \sum_i r_i$ .

```

---

#### 3.2.5.1 Explanation

- **Input:** The algorithm takes an input image  $x$  and a binary classifier  $f$ .
- **Output:** It outputs the perturbation  $\hat{r}$ .
- **Initialization:** Initialize the current image  $x_0$  as the input image and set  $i$  to 0.
- **Main Loop:** While the sign of the classifier's output for the current image is the same as the sign for the original image, perform the following steps:
  1. Calculate the perturbation  $r_i$  using the formula:  $r_i = -\frac{f(x_i)}{\|\nabla f(x_i)\|_2^2} \nabla f(x_i)$ .
  2. Update the image:  $x_{i+1} = x_i + r_i$ .
  3. Increment  $i$ .
- **Output:** The final perturbation is the sum of all perturbations:  $\hat{r} = \sum_i r_i$ .

The algorithm iteratively adjusts the input image until the classifier's output changes its sign. The perturbation  $\hat{r}$  is the accumulated perturbation over these iterations.

## 3.3 DIPDefend

Now let us look at the algorithm that we have used in our model:

Explaining the algorithm step wise :

**Initialization:**

**Require:**

- 1:  $x_{adv}$ : Adversarial example
- 2:  $x_t$ : Restored image at iteration  $t$
- 3:  $z$ : Random noise
- 4:  $p_t$ : PSNR value at iteration  $t$
- 5:  $s_t$ : Smoothed value at iteration  $t$
- 6:  $G_t$ : Generator at iteration  $t$

**Ensure:**

- 7: The restored image
- 8: Initialize  $p_0 = 0$ ,  $s_0 = 0$ , and  $t = 0$
- 9: **repeat**
- 10:    $x_{t+1} \leftarrow$  Compute  $G_t(z)$
- 11:    $p_{t+1} \leftarrow$  Compute PSNR between  $x_{t+1}$  and  $x_{adv}$
- 12:    $s_{t+1} \leftarrow$  Smooth  $p_{t+1}$  by Equ. (4) and Equ. (5)
- 13:    $G_{t+1} \leftarrow$  Update  $G_t$  with gradient descent
- 14:    $t \leftarrow t + 1$
- 15: **until**  $s_{t+1} < s_t$
- 16: **return**  $x_{t+1}$

- Initialize the variables:  $p_t$  for PSNR value,  $s_t$  for a smoothed value, and  $t$  for the iteration counter.

**Iteration:**

- Enter a loop to iterate until the smoothed value  $s_{t+1}$  becomes less than the previous smoothed value  $s_t$ .
- **Image Generation:**
  - Generate a new image  $x_{t+1}$  using a generator  $G_t$  with random noise  $z$ .
- **PSNR Computation:**
  - Compute the PSNR ( $p_{t+1}$ ) between the generated image  $x_{t+1}$  and the adversarial example  $x_{adv}$ .
- **Smoothing:**
  - Smooth the PSNR value ( $p_{t+1}$ ) using a smoothing process described by Equations (4) and (5).
- **Generator Update:**
  - Update the generator ( $G_{t+1}$ ) using gradient descent. This involves adjusting the parameters of the generator to improve the image generation process.

**Convergence Check:**



- Check if the smoothed value  $s_{t+1}$  is less than the previous smoothed value  $s_t$ . If not, continue the iterations.

### Return the Restored Image:

- Once the algorithm converges (i.e.,  $s_{t+1} < s_t$ ), return the final restored image  $x_{t+1}$ .

## 3.4 Zero-Shot Noise2Noise

The proposed method introduces a novel approach to single-image denoising, departing from traditional methods by utilizing a lightweight network trained on a pair of downsampled images derived from the original noisy observation. The process involves a pair downsampler and a zero-shot-image denoising method.

### Pair Downsampler:

The downsampler takes an input image  $y$  and generates two downsampled images  $D1(y)$  and  $D2(y)$ . This is achieved by dividing the image into non-overlapping patches, averaging diagonal and anti-diagonal pixels, resulting in two low-resolution images.

$$D1(y) = y * k1, \quad \text{where} \quad k1 = \begin{bmatrix} 0 & 0.5 \\ 0.5 & 0 \end{bmatrix}$$

$$D2(y) = y * k2, \quad \text{where} \quad k2 = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

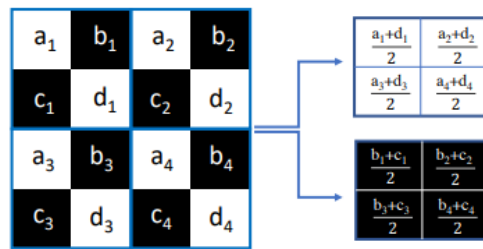


FIGURE 3.1: In the above example the input is a  $4 \times 4$  image, and the output is two  $2 \times 2$  images

### Zero-shot-image Denoising Method:

Given a test image  $y$  to denoise, the method involves fitting a small image-to-image neural network  $f_\theta$  to map  $D1(y)$  to  $D2(y)$  by minimizing the loss:

$$L(\theta) = \|f_\theta(D1(y)) - D2(y)\|_2^2 \quad \} \quad (3.2)$$

However, the method introduces critical elements for good performance:

### **Residual Learning:**

The loss function incorporates residual learning, optimizing the network to fit the noise instead of the image.

$$L_{\text{res}}(\theta) = \frac{1}{2} (\|D1(y) - f_\theta(D1(y)) - D2(y)\|_2^2 + \|D2(y) - f_\theta(D2(y)) - D1(y)\|_2^2) \quad \} \quad (3.3)$$

### **Symmetric Loss:**

A symmetric loss is adopted for residual learning.

### **Consistency-Enforcing Term:**

A consistency-enforcing term ensures that denoising before or after downsampling yields similar results.

$$L_{\text{cons}}(\theta) = \frac{1}{2} (\|D1(y) - f_\theta(D1(y)) - D1(y - f_\theta(y))\|_2^2 + \|D2(y) - f_\theta(D2(y)) - D2(y - f_\theta(y))\|_2^2) \quad (3.4)$$

The overall loss function is a combination of the residual and consistency losses:

$$L(\theta) = L_{\text{res}}(\theta) + L_{\text{cons}}(\theta) \quad (3.5)$$

### **Network:**

The employed network is a lightweight two-layer image-to-image network with about 20k parameters, avoiding normalization or pooling layers. This simplicity enables fast denoising, even on CPUs. The ablation studies demonstrate the superior performance of this lightweight network compared to larger networks.

In summary, the method offers a unique perspective on single-image denoising, emphasizing downsampling and an explicit regularization term to mitigate overfitting to a single image, providing a practical and efficient solution to the problem.

## Chapter 4

# Simulation and Results

### 4.1 FGSM

- Percentage of perturbation: 2.29%

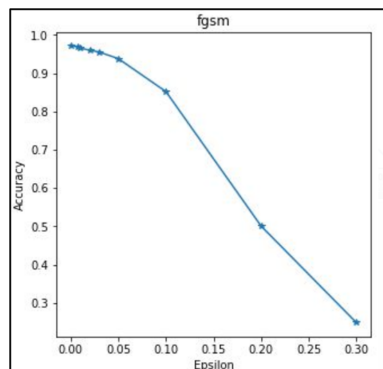


FIGURE 4.1: Accuracy vs Epsilon

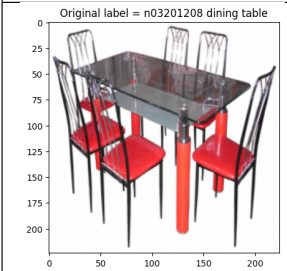
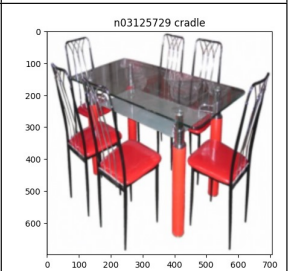




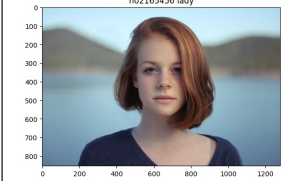
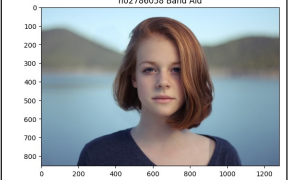




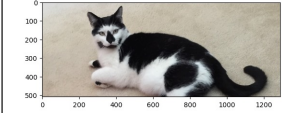
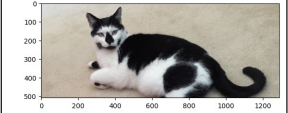
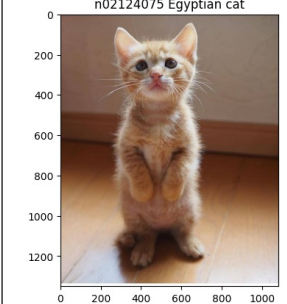
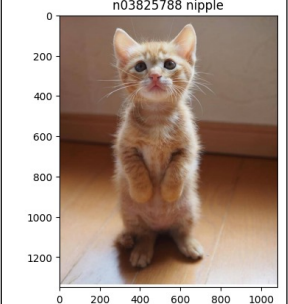
Original Image	Attacked Image
<p>Original label = n03201208 dining table</p> 	<p>n03125729 cradle</p> 
<p>n04285008 sports car, sport car</p> 	<p>n03100240 convertible</p> 
<p>n04285008 sports car, sport car</p> 	<p>n02704792 amphibian</p> 
<p>n02165456 lady</p> 	<p>n02786058 Band Aid</p> 
<p>n03792782 mountain bike</p> 	<p>n03791053 motor scooter</p> 
<p>n03124171 baby</p> 	<p>n03825788 nipple</p> 
<p>n02123597 Siamese cat</p> 	<p>n04372370 switch</p> 
<p>n02124075 Egyptian cat</p> 	<p>n03825788 nipple</p> 

TABLE 4.1: FGSM Model Results

## **4.2 DeepFool**

- Percentage of perturbation: 0.3%

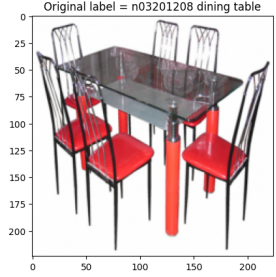
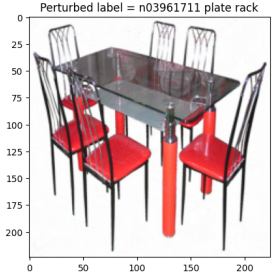




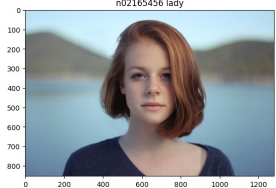





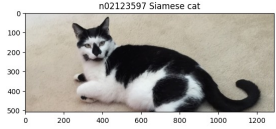
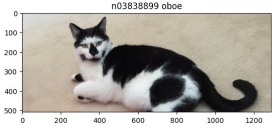


Original Image	Attacked Image
<p>Original label = n03201208 dining table</p> 	<p>Perturbed label = n03961711 plate rack</p> 
<p>n04285008 sports car, sport car</p> 	<p>n03791053 motor scooter</p> 
<p>n04285008 sports car, sport car</p> 	<p>n03977966 police van</p> 
<p>n02165456 lady</p> 	<p>n03595614 jersey</p> 
<p>n03792782 mountain bike</p> 	<p>n03594945 landrover</p> 
<p>n03124171 baby</p> 	<p>n03691459 loudspeaker</p> 
<p>n02123597 Siamese cat</p> 	<p>n03838899 oboe</p> 
<p>n02124075 Egyptian cat</p> 	<p>n01616318 vulture</p> 

TABLE 4.2: DeepFool Model Results

## **4.3 DIP Defend**

- Accuracy with FGSM: 94.9%
- Accuracy with DeepFool: 96.7%



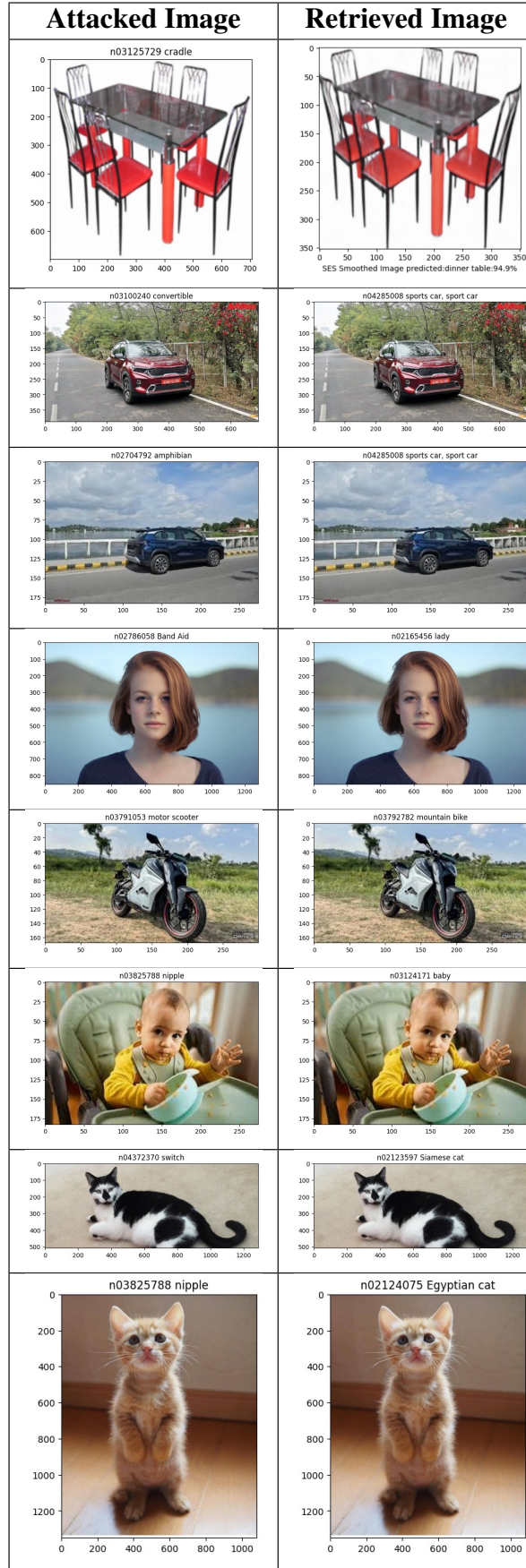


TABLE 4.3: FGSM attacked image defended through DIP Defend

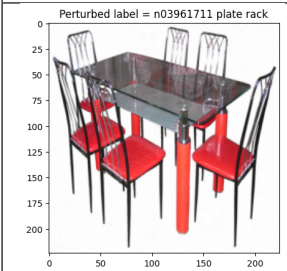
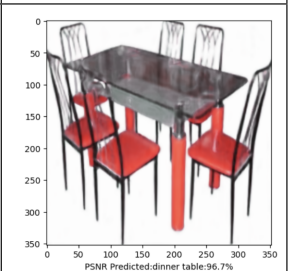
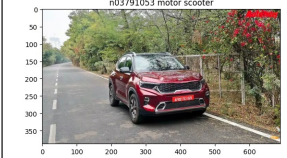



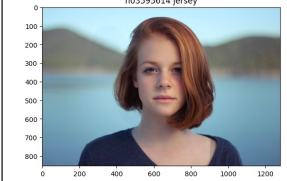
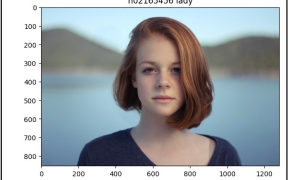




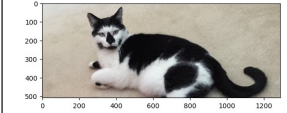
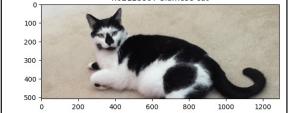
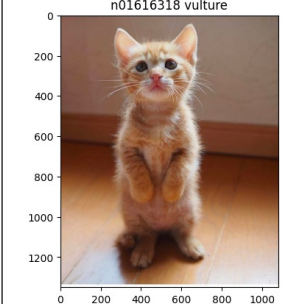
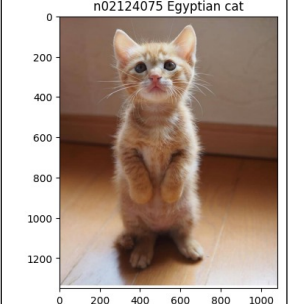
Attacked Image	Retrieved Image
<p>Perturbed label = n03961711 plate rack</p> 	 <p>PSNR Predicted:dinner table:96.7%</p>
<p>n03791053 motor scooter</p> 	<p>n04285008 sports car, sport car</p> 
<p>n03977966 police van</p> 	<p>n04285008 sports car, sport car</p> 
<p>n03595614 jersey</p> 	<p>n02165456 lady</p> 
<p>n03594945 landrover</p> 	<p>n03792782 mountain bike</p> 
<p>n03691459 loudspeaker</p> 	<p>n03124171 baby</p> 
<p>n04372370 switch</p> 	<p>n02123597 Siamese cat</p> 
<p>n01616318 vulture</p> 	<p>n02124075 Egyptian cat</p> 

TABLE 4.4: DeepFool attacked image defended through DIP Defend

## 4.4 Zero-Shot Noise2Noise

- PSNR with FGSM:  $22.02dB$
- Accuracy with DeepFool:  $27.1dB$

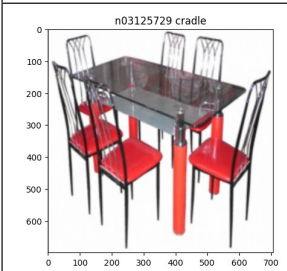
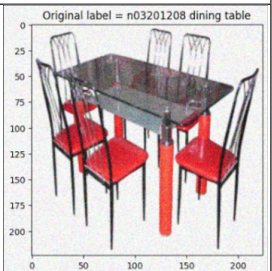





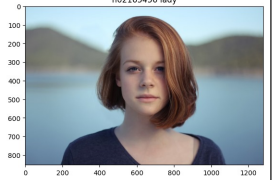




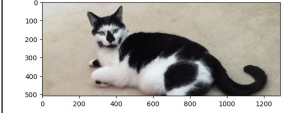
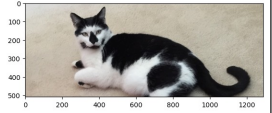
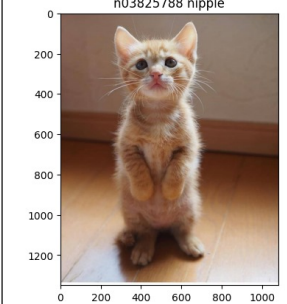
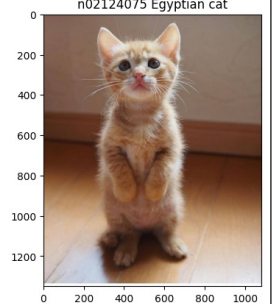
Attacked Image	Retrieved Image
	
	
	
	
	
	
	
	

TABLE 4.5: FGSM attacked image defended through Zero-Shot Noise2Noise



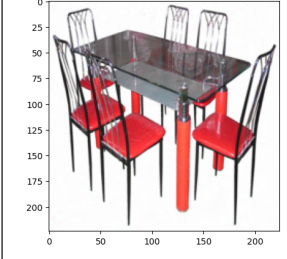




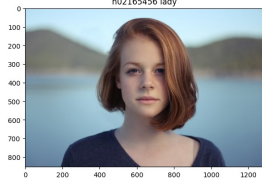




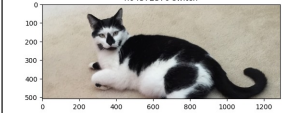
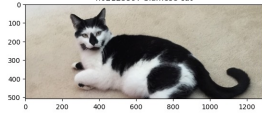
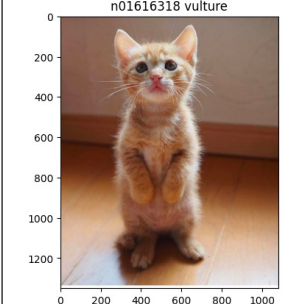

Attacked Image	Retrieved Image
<p>Perturbed label = n03961711 plate rack</p> 	<p>Original label = n03201208 dining table</p> 
<p>n03791053 motor scooter</p> 	<p>n04285008 sports car, sport car</p> 
<p>n03977966 police van</p> 	<p>n04285008 sports car, sport car</p> 
<p>n03595614 jersey</p> 	<p>n02165456 lady</p> 
<p>n03594945 landrover</p> 	<p>n03792782 mountain bike</p> 
<p>n03691459 loudspeaker</p> 	<p>n03124171 baby</p> 
<p>n04372370 switch</p> 	<p>n02123597 Siamese cat</p> 
<p>n01616318 vulture</p> 	<p>n02124075 Egyptian cat</p> 

TABLE 4.6: DeepFool attacked image defended through Zero-Shot Noise2Noise

## 4.5 Tabular Comparision

S. No.	Original	FGSM	DeepFool	DIPDefend	Noise2Noise
1	Dining Table	Cradle	Plate rack	Dining Table	Dining Table
2	Sports car	Convertible	Motor Scooter	Sports car	Sports car
3	Sport car	Amphibian	Police van	Sport car	Sport car
4	Lady	Band aid	Jersey	Lady	Lady
5	Mountain Bike	Motor Scooter	Landrover	Mountain Bike	Mountain Bike
6	Baby	Nipple	Loudspeaker	Baby	Baby
7	Siamese cat	Switch	Oboe	Siamese cat	Siamese cat
8	Egyptian cat	Nipple	Vulture	Egyptian cat	Egyptian cat

TABLE 4.7: Comparision between the predictions of different attack and defence models as compared to the Original image

Note: The attacked images were passed through the defence models to get the output. Also this is just a test dataset inspired from PASCAL VOC

## Chapter 5

# Conclusions and Future Work

Defence against adversarial attacks, particularly in the context of self-driving cars, is a key and demanding issue that necessitates robust and dependable solutions. Throughout our Bachelor's Thesis Project (BTP) research and implementation, we concentrated on protecting against two popular adversarial attack methods: Fast Gradient Sign Method (FGSM) and DeepFool. Deep Image Prior (DIP) Defend, and Zero-Shot Noise2Noise (ZS-N2N) were the defences chosen for our investigation. The overall purpose was to improve the security and resilience of self-driving cars, which are prone to hostile manipulation.

In our analysis, we discovered that adversarial approaches are successful at perturbing input data, particularly in the image domain. The powerful adversaries FGSM and DeepFool were used to assess the vulnerabilities of autonomous car systems. These attacks can jeopardise the decision-making processes of the vehicle's artificial intelligence, potentially resulting in dangerous driving situations.

Our defences of choice, DIP Defend, and ZS-N2N, represented several techniques to dealing with hostile attacks. To counter hostile perturbations, DIP makes use of the inherent structure of neural networks and employs an approach to detect and neutralise adversarial attacks, whereas ZS-N2N presents a novel zero-shot learning paradigm for defence, demonstrating promise without the use of a pre-existing dataset.

We tested the robustness of these defences against FGSM and DeepFool attacks using extensive experimentation. Our findings indicated the advantages and disadvantages of each defence mechanism. DIP-Defend proved the ability to recover from some adversarial manipulations by using the neural network's intrinsic structure.

In the face of hostile attacks, the unique Zero-Shot Noise2Noise defence demonstrated potential, coinciding with our goal of developing novel and effective defence mechanisms for autonomous cars. In the absence of paired adversarial instances, ZS-N2N's unique strategy of training on single noisy images, inspired by Noise2Noise, displayed resilience and adaptability.

Finally, our BTP contributes to the expanding field of autonomous vehicle security by offering insights into adversary attack vulnerabilities and presenting novel defence tactics. As autonomous vehicles grow more prevalent in our daily lives, improving their resistance to hostile attacks becomes critical for safety and reliability. Our findings establish the groundwork for future study into developing more advanced defence mechanisms to protect autonomous systems in real-world scenarios.



# Bibliography

- [1] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” *arXiv preprint arXiv:1412.6572*, 2014.
- [2] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “Deepfool: a simple and accurate method to fool deep neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.
- [3] D. W. B. C. J. L. Y. J. S.-T. X. Tao Dai, Yan Feng, “Dipdefend: Deep image prior driven defense against adversarial examples,” in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020.
- [4] D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Deep image prior,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9446–9454, 2018.
- [5] Y. Mansour and R. Heckel, “Zero-shot noise2noise: Efficient image denoising without any data,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14018–14027, 2023.
- [6] A. Kurakin, I. J. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” in *Artificial intelligence safety and security*, pp. 99–112, Chapman and Hall/CRC, 2018.
- [7] D. Wang, W. Yao, T. Jiang, G. Tang, and X. Chen, “A survey on physical adversarial attack in computer vision,” *arXiv preprint arXiv:2209.14262*, 2022.
- [8] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results.” <http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html>.
- [9] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, “A survey on adversarial attacks and defences,” *CAAI Transactions on Intelligence Technology*, vol. 6, no. 1, pp. 25–45, 2021.
- [10] A. Sharma, Y. Bian, P. Munz, and A. Narayan, “Adversarial patch attacks and defences in vision-based tasks: A survey,” *arXiv preprint arXiv:2206.08304*, 2022.