Random Filter-Switching-based Defense Against Decision-based Adversarial Attacks on Machine Learning

By

Rashad Khalid

00000275963

Supervisor

Dr. Muhammad Jawad Khan

Department of Robotics and Artificial Intelligence

School of Mechanical and Manufacturing Engineering (SMME)

National University of Sciences and Technology (NUST)

Islamabad, Pakistan

June 2022

Random Filter-Switching-based Defense Against Decision-based Adversarial Attacks on Machine Learning

By

Rashad Khalid

00000275963

Supervisor

Dr. Muhammad Jawad Khan

Supervisor's Signature: _____

A thesis submitted in conformity with the requirements for the degree of *Master of Science* in Robotics and Intelligent Machines Engineering

Department of Robotics and Artificial Intelligence

School of Mechanical and Manufacturing Engineering (SMME)

National University of Sciences and Technology (NUST)

Islamabad, Pakistan

June 2022

Declaration

It is certified that the final copy of MS Thesis written by Rashad Khalid (Registration No. 00000275963), of Department of Robotics and Intelligent Machine Engineering (SMME) has been vetted by undersigned, found complete in all respects as per NUST statutes / regulations, is free from plagiarism, errors and mistakes and is accepted as a partial fulfilment for award of MS Degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in this dissertation.

_____

Rashad Khalid,
00000275963

Copyright Notice

This thesis is dedicated to *my beloved parents*

Abstract

In the AI and machine learning research field, adversarial machine learning(AML), a technique that tries to deceive models using erroneous data, is becoming a major concern. By exploiting the inherent vulnerability of ML models' data reliance, AML can be used to generate adversarial attacks. Researches have shown that a small perturbation in input image can create disastrous results for an autonomous car system e.g. miscalssifying stop sign as speed limit sign near school. To counter these adversarial attacks, several defense mechanisms have been proposed. Some of the most prominent defenses are adversarial training, pre-processing-based defenses, Generative Adversarial Network-based defenses. However, most of these defenses are either computationally expensive or become in-effective under the white-box threat model or against the decision-based attacks (Adversarial attacks that exploit the final decision of the attack under black-box settings). Therefore, there is a dire need to develop efficient defense mechanisms that can effectively counter the attacks while maintaining the classification accuracy. In this thesis, we propose to develop a computationally efficient and effective defense mechanism that effectively counters the score-based and decision-based adversarial attack under black-box settings while maintaining the classification accuracy on clean images.

Acknowledgments

I would like to thank ALLAH almighty, my parents and my family especially my uncle because without their support I would not be there. In the end, I would like to pay special thanks to Dr. Hasan Sajid and Faiq Khalid, I could not complete this thesis without their kind guidance.

Contents

List of Figures

List of Tables

List of Abbreviations and Symbols

Abbreviations

ML	Machine Learning
DNN	Deep Neural Network
IoT	Internet of Things
AML	Adversarial Machine Learning
Adv.	Adversarial
ParDec	Parallel Decision-based Attack

Chapter 1

Introduction

In the AI and machine learning research field, adversarial machine learning(AML), a technique that tries to deceive models using erroneous data, is becoming a major concern. By exploiting the inherent vulnerability of ML models' data reliance, AML can be used to generate adversarial attacks. According to the National security commission on artificial intelligence, very little ML research is focused on safeguarding ML models against adversarial attacks. Researchers have shown that a small perturbation in the input image can create disastrous results for an autonomous car system, e.g., misclassifying a stop sign as a speed limit sign near the school. By adding imperceptible noise in the undergoing-test image, medical analysis can be compromised by classifying malignant moles as benign. With each passing day, adversarial attacks are increasing as the world is inclining towards ML-based systems. Based on the attacker's access to ML models' information, adversarial attacks can be classified into two types White box and Black Box attacks.

If an attacker has access to the ML model's internal parameters along with inputs and outputs, it is known as a white-box attack. Commonly used attacks generate adversarial examples under white-box settings are [1, 3–5]. As these all attacks works under white-box stetting these can easily defended by [6], defensive distillation [7, 8] and pre-processing defenses [9, 10]. In a real-world scenario, producing adversarial instances from data is difficult due to the attacker's lack of access to ML models' parameters or the training process. For this, score-based attacks are developed under black-box settings, which use probability scores of the ML model [11–14]. These attacks can be nullified by attack mentioned by Tramer or by the use of above-mentioned gradient-

based attacks in model stealing settings. Currently, decision-based evasion attacks has been proposed by [15**?** , 16], which uses final predicted decision of the ML model to generate adversarial examples. These all attacks use a random search algorithm to find the adversarial sample on the classification boundary, which increases the number of queries as well as the convergence time of the attack. FaDec [17] proposed to use the half-interval algorithm instead of the random search algorithm to find the adversarial example on the classification boundary. Although FaDec is near to practical scenarios, if the number of queries is limited to 100 or 200, then it fails to produce adversarial examples with acceptable perturbation. Moreover, it takes more time to converge in a time-constrained environment. For cloud-based ML systems, each query comes with a cost, which can increase the cost of attack many folds in monetary terms if queries are not limited. These observations raise a research question about how to develop an attack that uses a limited number of queries while minimizing the adversarial noise. Fault injection attacks can be performed on ML systems depending on their configurations. In summary, the state-of-the-art attacks have the following limitations:

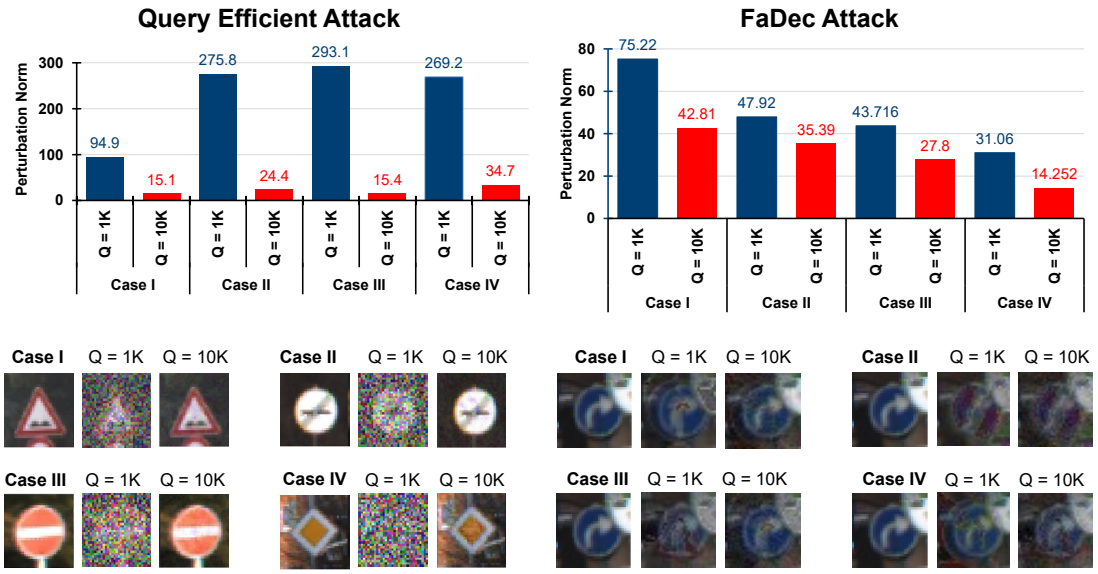

Figure 1.1: The motivation analysis shows the convergence time and number of queries required for a decision-based attack to perform successful misclassification.

- Most state-of-the-art black-box attacks require a large number of queries. To elaborate this limitation, we analyzed two the fastest state-of-the-art decision-based attacks, i.e., query efficient attack [16] and FaDec [17], as shown in Fig. 1.1. These results show that the imperceptibility of the adversarial perturbation increases as

the increase in allowed queries. Therefore, the attacker needs more resources and more number of queries, which can be defended by limiting the queries per user.

- Most state-of-the-art does not consider the complete pipeline of the ML-based system.

These limitations raise an important research question, "Can new decision-based attacks be devised that can operate successfully in real-time resource-constrained applications while ensuring imperceptibility and robustness of attacks?"

1.1 Parallel Decision-based attack

To address the above-mentioned research question, we proposed using the man-in-middle concept to parallelize the queries for developing a query-efficient and cost-effective decision-based attack called ParDec. The parallelism of the queries leads to a significant reduction in the number of required queries and attack convergence time. The ParDec consists of the following two steps:

1. *parallelism of the queries:* In this phase, the attack generate multiple images near the target images and then perform the half-interval search algorithm to find a adversarial image near the classification boundary.

2. *Optimization of Adversarial Noise:* In this phase, random noise is added to the selected sample to generate multiple perturbed images. Then the closest image from the target image is selected, and the classification boundary is selected. Then half-interval search algorithm is applied to that image. This process is repeated until the optimized adversarial image is generated.

To evaluate the effectiveness of ParDec, we performed this attack on the DNN model trained on GTSRB and CiFAR-10 datasets. The results of experiments show that ParDec is achieving significantly higher (more than 400%) imperceptibility with 5x a smaller number of queries.

1.2 Defenses against Adversarial Attacks

For defenses there are many defenses in the literature review. There are multiple types of the defenses, e.g., training on Adversarial images, squeezing the features, GAN-based attacks, Defense GAN attacks, augmenting the datasets, Quantization activation dynamically, and pre-processing filter-based defenses. All of these defenses are not resource constraints and cannot be applied to real-world scenarios. But our proposed defense offers resource-constraint applications, which exclude the process of adversarial training on adversarial examples, masking and training a separate model. Most of the attacks tackle white-box and black-box attacks as a combined, while others use a separate. We checked our defense in black box setting and decision-based attack and evaluated it against the state-of-the-art adversarial machine learning decision-based attacks. In summary, the state-of-the-art defenses exhibit the following limitations:

1. Most of the defenses require re-training, which in most of the cases is costly and also have an impact on the clean classification accuracy.
2. Some of the defenses are only applicable to the known adversarial attacks.

These limitations raise an important research question, "How to defend against efficient decision-based adversarial attack with minimum overhead?"

1.3 Random filter Switching-based Defense

To counter the above-mentioned research challenge, we propose to expand the existing pre-processing-based defense by randomly switching the filter configuration and type of filter in the pre-processing layer, called RaFiS. The main motivation behind this idea is that the adversarial image passes through a different pre-processing layer for every query. This leads to an increment in the convergence time and number of required queries. It is important to note that the increment in the number of filters in the pre-processing layer increases the attack complexity but it also increases the cost.

To evaluate the effectiveness of RaFiS, we defended the DNN model trained for GTSRB and CiFAR-10 against the proposed ParDec and the fastest decision-based attack, FaDec. The experimental results show that in all cases, RaFis nullify the attack by

preventing it from misclassification. However, In the case of FaDec, RaFiS decreases the perturbation norm by 21% and increases the perturbation norm by 6.667 times.

1.4 Novel Contributions

In summary, this thesis has the following novel contributions:

1. **Parallel Decision-based attack:** We proposed to use the concept of man-in-middle to parallelize the queries for developing a query-efficient and cost-effective decision-based attack called ParDec (see Chapter 4).

2. **Random filter Switching-based Defense:** We propose to expand the existing pre-processing-based defense by randomly switching the filter configuration and type of filter in the pre-processing layer, called RaFiS (see Chapter 6).

1.5 Organization of the Thesis

For the flow of the thesis, the first chapter (see Chapter 1) is about introduction, the second chapter (Chapter 2) gives a detailed overview of the background and terminologies used in the thesis, the third chapter (Chapter 3) gives the literature review of the past work done in the area. The fourth chapter (Chapter 4) gives the methodology of the proposed attack. The fifth chapter (Chapter 5) presents the results of the proposed attack, the sixth chapter (see Chapter 6) discusses defense methodology, and the seventh chapter (see Chapter 7) discusses the defense results. Finally, the eighth chapter (see Chapter 7.2.2) concludes the thesis.

Chapter 2

Background

In this chapter, we discuss some preliminaries that facilitate the reader's understanding of the thesis's key concepts, results, and observations. Towards this, we provide the details of parameters related to threat models, which are assumed in the adversarial attacks on neural networks. Moreover, we briefly discuss the parameters that are used to define and measure the human imperceptibility related to adversarial noise.

2.1 Threat Models

Machine learning systems, especially neural network-based systems, are used in many safety-critical applications. e.g., healthcare, autonomous driving, etc. These systems are by nature not secure because of their dependency on data, which can be manipulated to perform misclassification. Several researchers have exploited this behavior to design adversarial attacks. The efficacy of these adversarial attacks depends on a set of assumptions and parameters. These assumptions and parameters are often referred to as threat models for adversarial attacks, as shown in Fig. 2.1. In any threat model, the attacker is typically called an adversary. An adversary can change the configurations of these assumptions and parameters to generate different forms of attack for a specific scenario. For example, if an attacker has access to the training process, then its parameters configurations will be different from those who do not have access to the training process. These different scenarios of configurations are known as threat models. For ML security, a threat model consists of the following set of parameters.

- **Attacker's Knowledge** is the details or information about an ML-based system that

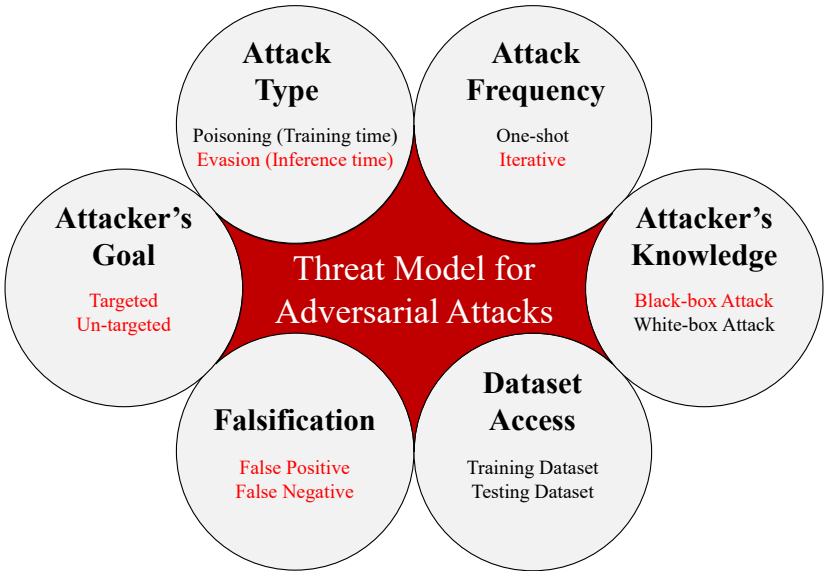

Figure 2.1: A set of assumptions and parameters is typically used to make the threat model for performing any adversarial attack. The red highlighted text represents the assumptions and parameters assumed in the proposed attack and defense.

is accessible to the adversary, i.e., how much an attacker has access to inputs, ML-based model, and outputs. If an attacker can manipulate inference by adding noise to the input. If an attacker has access to ML model parameters, it can compromise the model for wrong outputs by adding a bug in the model's architecture. If an attacker has access to the ML model's output probability vectors, it can exploit these probabilities to generate adversarial examples. The attack can be categorized as follows based on the attacker's knowledge:

- ◇ **White-Box Attack:** An attacker has full access to inputs, ML model, and outputs. To perform these kinds of attacks, an attacker can exploit information of inputs, ML model architectures, and outputs separately or in the combination of any or all parameters.
- ◇ **Black-Box Attack:** An attacker can only access inputs and outputs of an ML system. It has no access to model architecture or parameters. In such a scenario, the attacker must generate its attack by using inputs and inferred results of the ML module.

- **Attacker's Goal** is defined as the malicious intent of the attacker, which it wants to obtain from a particular adversarial attack. Based on the different payloads of the attack, It can be classified into two categories given below.

- ◇ **Targeted Attack:** In this category of attack, an attacker tries to misclassify the input into a specific class by performing the attack. This attack typically modifies ML model parameters, poisons inputs, and exploits outputs. For example, the stop sign in traffic signs can be misclassified into a 60 mph speed sign which may lead to catastrophic consequences.
- ◇ **Un-Targeted Attack:** An attacker's goal is to misclassify the input into any available class other than ground truth. An attacker can redouble the prediction error by decreasing the prediction score of the true class during the attack. For example, the stop sign in traffic signs can be misclassified into any traffic sign except the stop sign.

- **Attacker's Frequency:** Attacker can perform the attack in a single query or multiple queries through the ML model, known as attacker's frequency. Based on this query scenario, an attack can be classified into two types.
  - ◇ **One-Shot Attack:** In this type of attack, the attacker performs the attack in a single query through the ML model. The attack can only be optimized once in a one-shot attack.
  - ◇ **Iterative Attack:** In this type of attack, the attacker performs the attack in multiple queries through the ML model. The attack is optimized in each iteration of the model query. Iterative attacks have a relatively slow convergence rate, but attack efficiency is significantly better than one-shot attacks.

- **Attack Falsification:** In this type of attack, the attack is classified based on the types of misclassifications given below.
  - ◇ **False-positive attack:** In this type of attack, the prediction of the ML model is falsely classified as the positive class. For example, a benign mole, a negative sample, can be falsely classified as a malignant mole, a positive mole.
  - ◇ **False-negative attack:** In this type of attack, the prediction of ML molded is falsely classified to negative class. For example, a malignant mole, a positive sample, can be falsely classified as a benign mole, a negative mole.

- **Attack Type** is defined as the phase of the ML system architecture on which an attack is performed. This depends on the attacker's knowledge, i.e., how much the attacker has access to ML model information related to inputs, model architecture,

and outputs. Depending upon the available information, the attack can be classified as follows.

- ◇ **Training-phase attack:** In this type of attack, the attacker tries to compromise the model by poisoning training data. This attack can only be performed if the attacker has access to the training process. For example, if an organization outsources the training process, the adversary (from the outsourced organization) has access training process and can poison the data.
- ◇ **Inference-phase attack:** In this type of attack, the attacker tricks the model inference by exploiting model parameters or model architecture. This attack can only be performed if the attacker has access to ML model parameters and architecture.
- ◇ **Hardware of ML model:** In this type of attack, the attacker tries to malfunction the ML system at the hardware level in any cyber-security system.

2.1.1 White-box Attack

In the white-box attack, an attacker has full access to input, trained ML model architecture, and predicted probabilities of labels. For example, an attacker can exploit the information available at any of the above-mentioned information and perform an adversarial attack, as shown in Fig. 2.2(b). If an attacker has access to gradients of the ML model, the gradients can be used to generate an attack, known as gradient-based attacks. For example, Fast Gradient Sign Method (FGSM) [1], a white-box attack, generates attacks by adding the perturbation noise in the direction which has maximum effect on ML model inference. The direction is determined by using the cost function of the model(available in a white-box setting).Iterative-FGSM (iFGSM) [18], a white-box attack performs targeted attack, i.e., misclassifying the image to a specific class(target class). It utilizes the model's gradients to find the direction of adding noise in each step.

It is important to note that the above-mentioned white-box attacks need to know about the model information, e.g., parameters and gradients. They require a white-box setting to generate an adversarial attack.

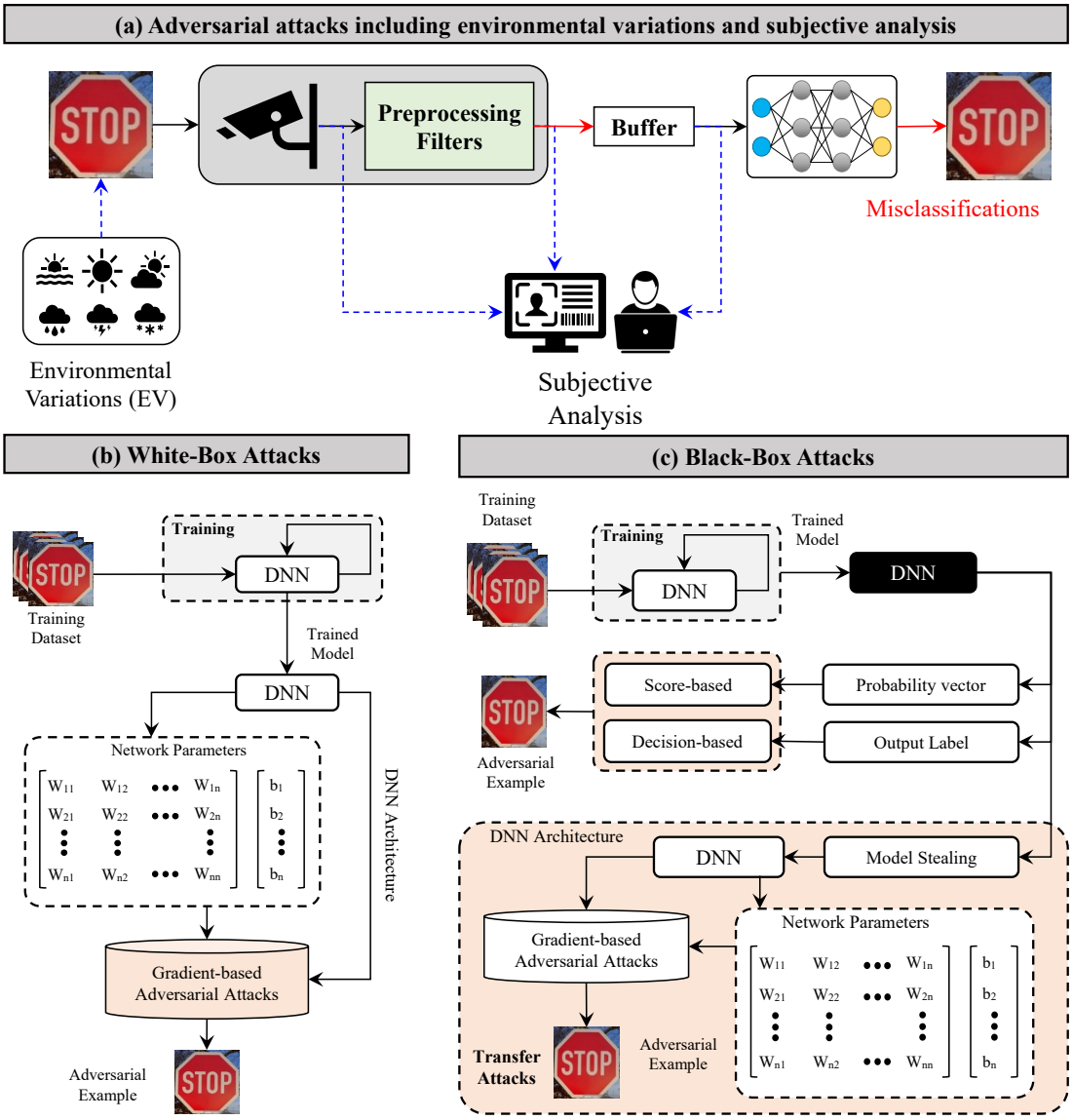

Figure 2.2: (a) Adversarial attack is defined as a crafted imperceptible noise to misclassify the ML-based System. (b) In white-box attacks, the attacker has complete access to the model parameters and other key components of the neural network. In this attack, the attacker derives the cost function, which is the difference between the ground truth and attack target; and propagates it layer-by-layer. (b) In a black-box attack, attacker access is limited to the input and output of the ML model.

2.1.2 Black-box Atatck

In the black-box attack, an attacker only has access to an ML model's input and output. There is no access to the probability of output, as shown in Fig. 2.2(c). In this adversarial setting, the adversary can only change the input of the ML model to malfunction the model. Typically, based on the attacker's access to the output, these attacks can be classified into score-based attacks and decision-based attacks. In the score-based attack,

the attacker has access to the top-1 probability of the output and can exploit it to generate the adversarial attacks by doing multiple queries. In the decision-based attack, the attacker has access to the top-1 label, hence making it very difficult to generate the adversarial noise. For example, in decision-based attacks, first, the attacker tries to alter the input by adding large noise, which can misclassify model inference [17]. The attacker's goal in such an attack is to find an adversarial example with a different label than the actual class near the classification boundary. For this, first, a random example having a label other than the actual class is picked. Second, an adversarial example is generated by moving a random example towards a classification boundary such that the label of the adversarial example is different from the actual class. Lastly, the distance between the generated adversarial example and the target example should be minimum in moving the random example so that the minimum amount of noise will be added to the clean image.

It is important to note that each iteration costs one query to the model in moving random examples toward the classification boundary. Therefore, a large number of queries are required to converge adversarial examples near the classification boundary. Such attacks can struggle in query-restricted environments, but they operate in realistic environments. Note that all the white-box attacks can be implemented in the black-box setting if combined with the model stealing attacks, known as transfer attacks (see Fig. 2.2(c)).

2.1.3 Threat Model for the Proposed Attack and Defense

In this thesis, to analyze the security of the neural network with practical assumptions, we assumed the decision-based black-box settings. In this setting, the attacker has access to the top-1 label of the ML classification. The set of assumptions based on the threat model discussed in Section 2.1 are given below (see red highlighted text in Fig. 2.1):

- ◇ **Attacker's Knowledge:** Adversary has no access to the training dataset, ML model parameters, model gradients, and classification probabilities. Adversary can only access input and also have access to the top-1 label of the output.
- ◇ **Attacker's Frequency:** The proposed attack can either be targeted or un-targeted.
- ◇ **Attacker's Frequency:** The proposed attack is an iterative attack.

◇ **Attack Falsification:** The proposed attack can misclassify negative class to positive class and positive class to negative class. Hence it can generate False-positive attacks as well as False-negative attacks. For example, speed more than 40 mph is a negative class, and other is a positive class. This attack can malfunction 40 mph speed sign to a 60 mph speed sign and vice versa for the high-speed motorway.

◇ **Attack Type:** The proposed attack exploits the model inference by adding perturbation in the input. Moreover, this attack can also work on the hardware level in any cyber-security system. Hence, it can attack the ML system's inference phase and hardware level.

2.2 Human Imperceptibility

The key goal in the adversarial attack is to add imperceptible noise that can lead to misclassification. Human imperceptibility in adversarial ML is defined as difficulty in perceiving the added noise in clean input. It is used to check the quality and stealthiness of the attack; the more the attack is imperceptible, the more that attack is effective and stealthy. Therefore, all adversarial attacks try to maximize imperceptibility. Any attack perceptible to human can not produce the desired result in practical scenarios. For example, the subjective analysis (see Fig. 2.2(a)) can detect perceptible noise like the manual checking in face recognition systems. Similarly, in a traffic sign of 60 mph, if noise added by the attack is perceptible to a human, then this perturbed image can be reported as malicious input by any human, which will make the attack ineffective. Therefore, to measure human imperceptibility, researchers have used several parameters, and some of them are discussed below:

- **Perturbation Norm** is known as the mean square difference between attacked image and the clean image. This is the most commonly used parameter to ensure imperceptibility in adversarial noise. It is denoted by d. Mathematically it is shown as follows.

$$d = 1/n \sum_{i=1}^{n} (y_i - y_i')^2 \tag{2.2.1}$$

where y_i and y_i' are the i_th pixel from the clean input image and adversarial image, respectively. **n** is the total number of pixels in the images. Human imperceptibility

should be near "0 " for high imperceptibility.

- **Cross Cor-relation Coefficient** is known as the probability of the linear relationship between the given two images. It has range between 0 and 1; 0 for minimum human imperceptibility and 1 for maximum human imperceptibility. For two same images **CC** will be '1'. It is denoted by CC. Mathematically, it is shown as follow.

$$d = \sum (x_i - x_i')(y_i - y_i') / \sqrt{\sum (x_i - x_i')^2 \sum (y_i - y_i')^2} \tag{2.2.2}$$

where r = correlation coefficient
x_i = pixel values of clean image
x' = mean of pixel values in clean image
y_i = pixel values of clean image
y' = mean of pixel values in perturbed image

- **Structural Similarity Index:** Structure Similarity Index is interpreted as perceptual similarity between two reference image and distorted or perturbed image. It is used to quantify degradation caused by data compression in an image. It is calculated using contrast, luminance and structure comparison. It is denoted by **SSIM**. It has range between 0 and 1. For maximum human imperceptibility, SSIM of two images should be 1.

$$SSIM = (2\mu_x\mu_y + c_1)(2\mu_x y + c_2) / (\mu_x^2 + \mu_y^2 + c1)(\sigma_x^2 + \sigma_y^2 + c2) \tag{2.2.3}$$

Note that in this work, we have used all three parameters to ensure imperceptibility in adversarial attacks. The rationale for this is that the perturbation norm cannot detect a sudden change in a few pixels. Therefore, to cover this limitation, we incorporated SSI and CCI for the evaluation of adversarial attacks and their defenses.

Chapter 3

Literature Review

In this chapter, we briefly discuss the impact of adversarial attacks on ML-based systems and existing literature on adversarial attacks and their respective defenses.

3.1 Adversarial Attack on Machine Learning

In this digital age, Machine Learning (ML) applications, especially safety-critical applications, are expanding in various fields as the number of collected data increases. Machine Learning algorithms follow the technique of extricating information from the data and using it for specific purposes. This shows that ML algorithms are highly dependent on training data. This dependency can be exploited to perform security attacks like generating adversarial examples, as shown in Fig. 2.2. Many ML algorithms struggle to perform against adversarial examples and misclassify these examples. This is because of the intrinsic dependency of ML algorithms on training data [1]. Hence, ML-based safety-critical applications are becoming vulnerable to simple but effective security attacks. Several techniques have been proposed to exploit these security flaws, known as adversarial attacks. Mainly, these attacks have been classified into poisoning and evasion attacks [2].

3.1.1 Poisoning Attacks

If the adversary has access to the training dataset and training process, it can manipulate both the training process and the training dataset, as discussed in Section 2.1. The attacks performed by such an adversary are known as poisoning attacks. These attacks

perform misclassification by manipulating the training dataset, training algorithm, or un-trained model (before or during training). As this noise poisoning in the training dataset and training algorithm act as a cause to generate the attack, they are also known as causative attacks. These attacks can be categorized into two types.

- **Dataset Poisoning:** In this type of attacks, attacks are formulated by adding random or crafted noise in the training dataset. As perturbation is added through backdoor channel, these attacks are also known as backdoor attacks. In this, attacker tries to maximize the classification error or target misclassification. Poisoning Attack on SVM target to misclassify Support Vector Machines inference by adding adversarial labels in the training data. The adversarial labels are generated by introducing carefully crafted noise in the training data to generate targeted attack. Targeted Clean-Label Poisoning [19] proposed to generate clean-label targeted poisoning attack, which is transferable. To generate poisonous images, this attack trains substitute models on training dataset for adversarial objective function. Watermarking [20] starts the attack by making apparently non-poisonous images open-source on web and wait for the victim to add this in its training dataset. This attack can be used to perform targeted attack. BadNets explore the aspects of backdoor channel attacks, when an adversary trains back-door network which have the same accuracy as of original trained network but misbehaves on some specific examples, controlled by adversary. Dynamic Backdoor Attacks [21] improved the backdoor channel attacks by introducing dynamic triggers.

  Limitations: Although these dataset poisoning attacks are very effective, in these attacks adversary needs to access the training dataset of the ML model, which is very difficult in the practical scenario. In addition, these attacks can easily be defended by limiting access to the training dataset. For example, an organization can distribute its training process among multiple users while outsourcing to limit access.

- **Model Poisoning attacks** are formulated when the attacker slightly modifies the architecture of the ML model to misclassify the inferred results or maximize the classification error. One of the examples of model poisoning attacks is weight poisoning [22], which starts the attack when a victim downloads untrusted pre-trained weights from the internet. These compromised model weights can be used to open backdoor channel attacks. Similarly, another example is the Local Model Poisoning

Table 3.1: A brief comparison of the state-of-the-art adversarial attacks (Evasion attacks) on ML-based systems [2].

Adversarial Attacks		Iterative/ One-shot	Targeted/ Un-Targted	Imperceptibility Parameter
Gradient-based Attacks	Limited memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) [1]	Iterative	T	l_2 norms
	Fast GradientSign Method (FGSM) [1]	One-shot	T/U	l_0, l_2, l_∞ norms
	Iterative FGSM (iFGSM) [1]	Iterative	T/U	l_0, l_2, l_∞ norms
	Basic Iterative Method (BIM)	Iterative	T/U	l_0, l_2, l_∞ norms
	Jacobian-based Saliency Map Attack (JSMA)	Iterative	T	l_0 norm
	Carlini & Wagner l_2 attack [3]	Iterative	T	l_2 norm
	Carlini & Wagner l_∞ attack [3]	Iterative	T	l_∞ norm
	DeepFool [4]	Iterative	U	l_2 norm
	Universal Perturbations	Iterative	U	l_p norm
	NewtonFool	Iterative	U	Tuning parameter
	TrISec [5]	Iterative	T/U	SSI, CC
Transfer Attacks	Ensemble Transfer		T	l_2, l_∞ norms
	FGSM transfer	Iterative	U	l_0, l_2, l_∞ norms
Score-based Attacks	Zeorth Order Optimization (ZOO) [11]	Iterative	T	l_2 norm
	Local Search	Iterative	U	
	HopskipJump [23]	Iterative	T/U	l_2 norm
Decision-based Attack	Query Efficient [16]	Iterative	T/U	l_2 norm
	Decision-based [15]	Iterative	T/U	l_2 norm
	FaDec-Attack [17]	Iterative	T	l_2 norm, SSI, CC
	Multi-Query Attack (This Work)	Iterative	T/U	l_2 norm

attack, which proposes to poison parameters of local models in a federated learning system that is used to formulate an attack on a global model.

Limitations: One of the biggest challenges in the model poisoning attack is that it requires access to the parameters, including the weight of the trained ML model. However, it is challenging to get access to the model's parameters in real-world scenarios. These attacks can be defended by deploying any ML model in black-box form, which prohibits the user from accessing the parameters of the model.

3.1.2 Evasion Attacks

In these type of attacks, a perturbation is added in the input of the trained ML model during inference stage, which can misclassify the model's inference. This perturbation is is known as adversarial noise. These attacks can be used to generate targated and un-targeted attacks. Due to evasive nature during inference of these attacks, they are also known as evasion attack. These attacks are classified into three categories (given in Table 3.1).

- **Gradient-based Attacks:** These attacks use the parameters of the ML model in

computing the gradients to generate adversarial noise. Fast Gradient Sign Method (FGSM) [1], a gradient-based attack, uses cost function with input, model parameters, and output to find out the gradient direction in which perturbed noise will have the greatest effect, as shown in Fig. 3.1(a). After that, a small adversarial noise is added in the acquired direction of the gradient in a single iteration. Iterative Fast Sign Method [18] is a variant of FGSM, and it adds perturbation in an iterative manner while generating the attack. Its cost function corresponds to a specific target. Hence it performs targeted misclassification. Jacobian Saliency Map Attack determines the derivative with respect to all input nodes to construct a saliency map, as shown in Fig. 3.1(b). This map can be used to determine the perturbation, which can generate a successful attack using the minimum number of input nodes. Carlini and Wagner attack [3] is a gradient-based attack, which minimizes the added noise with respect to a specific label and optimizes the objective function of misclassification of the targeted label. TrISec [5] uses a back propagation algorithm on a pre-trained ML model, without any information about the training data set, to generate an adversarial attack. Similar to the Iterative Fast Sign Method, this attack also considers the perturbation finding as an optimization problem.

Projected Gradient Descent(PGD) considers the perturbed noise as a large-scale constrained optimization problem of loss landscape on multiple data sets. It uses projected gradient descent to explore large spaces in a loss landscape. Auto-PGD improved step size, objective function, and proposed parameter-free attack. It can be used to generate target and un-targeted attacks. Iterative Frame Saliency [24] computes gradients through the classifier and optical flow to generate an attack for action recognition. For computing optical flow gradient, it used FlowNet2 as it estimates optical flow between successive frames. DeepFool [4] improved the approximation of the optimal perturbation vectors in FGSM while computing gradients. Universal Perturbations tries to find the perturbation vectors that fool the model on almost all images from a specific data distribution. This attack usually works in an iterative manner and usually performs un-targeted attacks. Newton Fool introduced a gradient descent algorithm for performing the attack and computing adversarial examples. It also improved the metric to check the imperceptibility of adversarial examples by introducing tuning parameters. Feature Adversaries tries to find small perturbations to the source image given a source image iteratively by using internal layers of DNN.

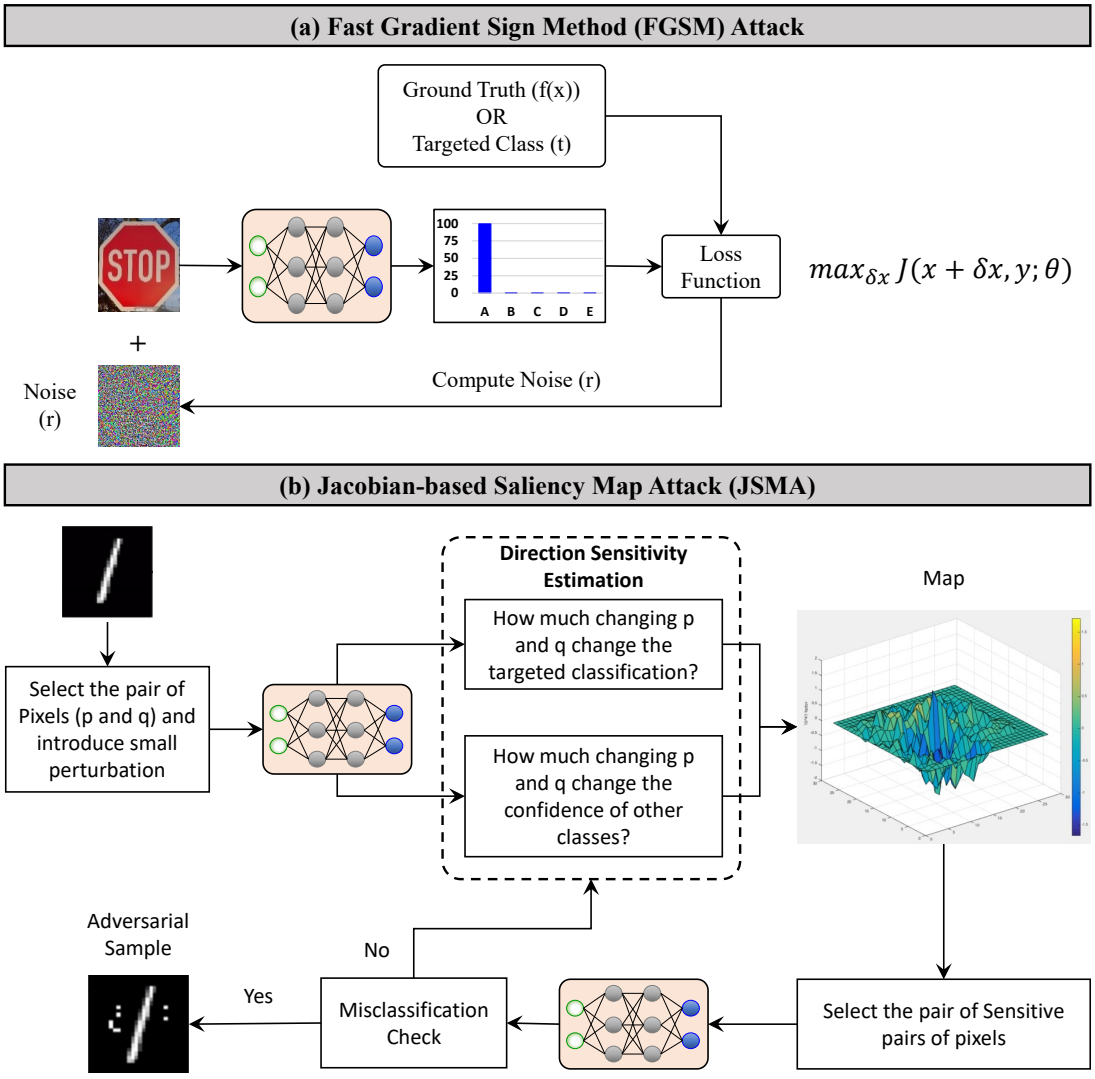

Figure 3.1: A pictorial view of gradient-based evasion attacks. (a) FGSM Attack (b) JSMA Attack

Adversarial Patch [25, 26] proposed the attack to generate an adversarial patch in case an adversary is not restricted to imperceptibility. Elastic-Net [27] focused on imperceptibility metric L_1 norm instead of L_2 and L_∞, while generating attack and showed attack's transferability. DPATCH [28] extends the attack of the adversarial patch to manipulate bounding box regression and object classification to generate location-independent targeted and un-targeted attacks. It also shows the attack's transferability between different ML networks. Wasserstein Attack [29] used Wasserstein distance as imperceptibility norm instead of L_p norm while computing the gradients of the classifier. It can perform targeted and un-targeted attacks. Shadow Attack [30] generates adversarial examples by focusing on manipulating certificates

issued by certified classifiers along with model classifiers.

Limitations: Most of the gradient-based attacks mentioned above require information about ML model parameters or gradients [3–5, 18, 31–33]. Hiding the gradients of the ML model, known as Gradient masking, can counterbalance these gradient-based attacks [6]. Moreover, deployed ML model does not provide access to gradients in a realistic environment.

- **Score-based Evasion Attacks:** In this type of attack, an adversary has access to the scores of prediction or probabilities of the ML model. Any change in the input corresponds to the change in the prediction scores and can be used to compute the strength and direction of perturbation. Zeroth Order Optimization attack [11] used confidence scores of output with input to compute gradients of the ML model for generating adversarial examples for a targeted attack. This attack does not use the internal information of the model. Local Search [12] considers the ML model as a black-box model, adds significant perturbation in a random set of pixels, and uses a greedy search algorithm to make added perturbation small and minimize the score of the true class label. This attack performs un-targeted attacks. Square Attack performs a random search near the classification boundary and uses perturbation updates in square form to generate the attack. Copy and Paste attack [13] used the approach of copying patches of specific properties from other images to original images and manipulating these patches with respect to changes in confidence scores to generate adversarial examples. It also focuses on making the attack query-efficient. It can be used to perform targeted and un-targeted attacks. One pixel attack [14] proposed to generate an attack by adding perturbation in one pixel or a few pixels with respect to probabilities of labels.

  Limitations: In a realistic environment, an adversary can only access the ML model's inputs and outputs. For example, cloud-based ML services like Google Cloud AutoML, Amazon, Microsoft Azure and IBM Watson provide black-box access to their trained model. For this, black-box attacks have been developed. Papermnot et al. [34] trains substitute model with the help of synthetic input and inferred output. Gao et al. [35] developed a score-based strategy to find and modify the most important words that can malfunction Deep Neural Network(DNN). C. Guo et al. used confidence scores of inference to generate adversarial examples. However, this attack

requires confidence scores continuously. F. Tramer et al. [**?**] introduce Ensemble Adversarial Training, which can neutralize score-based attacks by hiding information about scores probabilities. Cao. et al. compute gradients using transfer-based prior for generating an attack in the black-box setting. Suya utilizes the techniques of transfer as well as score-based attacks. All above-mentioned black-box attacks and [11, 12, 36] uses output score probabilities for computing gradients or stealing models for developing adversarial examples. Model stealing attacks can be countered by using above mentioned white-box dense techniques like gradient masking, pre-processing filtering, and defensive distillation.

- **Decision-based Evasion Attacks:** If the attacker does not have access to the output model probabilities, then researchers have developed the attacks that require an only a top-1 label. These attacks act in black-box settings and depends solely on the decision model. The process starts with introducing random noise in the input image, which causes misclassification. Then, added noise is reduced to make it imperceptible while conforming to the misclassification. These attacks are also known as boundary attacks as they try to find adversarial examples of classification boundaries. HopskipJump [23] generates targeted and un-targeted attacks by computing gradient estimation for perturbation by using binary search on the classification boundary. Query Efficient Attack [16] formulate the attack by performing the random walk on the classification boundary and using the zeroth order optimization algorithm to optimize the cost function, which is not always continuous in score-based attacks, to reduce the number of queries. Decision-based Attack [15] proposes to explore the classification boundary by using a random search for generating adversarial attacks near the classification boundary. Geometry-Inspired Decision-based (qFool) started the attack by introducing random perturbation and then exploited the geometric properties of the decision boundary to compute gradient estimation direction for generating targeted and un-targeted attacks. Query Efficient Boundary Attack (QEBA) improved the attack by proposing a framework to consume less number of queries for gradient estimation on classification boundary. This framework improves the number of queries by reducing the dimension of higher-dimension data while computing gradient estimation. FaDec [17] reduced the number of queries by using an iterative half-interval search instead of a random search algorithm to explore classification boundaries. It also improves imperceptibility by using distance-based

gradient sign estimation along with a half-interval search algorithm.Y. Dong et al. generate an attack by exploring the classification boundary by modeling the local geometry of search directions near the boundary. J. Chen et al. incorporate gradient estimation in searching adversarial examples near the classification boundary.

Limitations: Above mentioned decision-based attacks use the inferred decision of ML model instead of scores probability vector. Most of these decision-based attacks use a random search algorithm and multiple reference images to generate imperceptible attacks near the classification boundary, which notably increases the number of queries to the ML model.

3.1.3 Limitations of state-of-art-adversarial Attacks

To reduce the number of queries in decision-based evasion attacks, Fadec [17] proposed to use a half-interval search algorithm instead of a random search algorithm in exploring classification boundaries and also uses one sample example as a reference image instead of multiple reference images. It is applicable in practical scenarios to some extent, but in a query-restricted environment, if queries are restricted to 100 or 200 for a single user instead of 1000, Fadec fails to generate adversarial images with acceptable imperceptible noise. Moreover, Fadec takes more time in the timing-constrained environment as its queries depend on previous query results during the gradient search part. Therefore, in this work, we have proposed a more efficient attack that perform successful attack with very limited number of queries.

3.2 Defense against Adversarial Attacks

To counter the above mentioned attacks, various defenses have been proposed, as summarized in Table 3.2. Training ML model on generated adversarial examples is introduced by [1] known as adversarial examples. It follows the assumption that adversarial examples to misclassify one ML model will create malfunction on other ML models. Thus, this defense will work only against known attacks and struggle against unknown attacks.

Table 3.2: Comparison of methodology the state-of-the-art defenses for adversarial attacks on ML-based systems

Defenses	Brief Methodology	Threat Model	
		White-Box	Black-Box
Adversarial Learning [1]	Trains the CNNs for generated adversarial examples	✓	✓
Feature Squeezing [37]	Input transformation	✓	✓
BReLU + GDA [38]	Input transformation with adversarial learning	✓	✓
APE-GAN [39]	Input transformation with adversarial learning using GANs	✓	✓
Defense-GAN [40]	Input transformation	✓	✓
Defensive Distillation [6? –8]	Gradient masking	✓	
Data Augmentation [41]	Improve the diversity of the training data	✓	✓
Dynamic Quantization Activation [42]	Quantized activation with gradient masking	✓	✓
SSCNets [9]	Add pre-processing layer before training	✓	✓
QuSecNets [43]	Add pre-processing layer before training	✓	✓
RandFil (This work)	Add pre-processing filter before each query	✓	✓

3.2.1 Adversarial Training:

One of the very naive methods is to train the ML model on generated adversarial examples, introduced by [1], known as adversarial training, as shown in Fig. 3.2. It assumes that adversarial examples of misclassifying one ML model are transferable to other ML models. Thus, this defense will work only against known attacks and struggle against unknown attacks. The main purpose of adversarial training is to train a model on adversarial examples by adding them to training data [1, 44, 45]. For this, a defender generates a lot of adversarial examples by using existing attacks and augmenting its training data with [46]. By using generated adversarial examples, the ML model can be trained with a modified objective function in a way that model will predict the same output for clean and perturbed example [1].

Limitations: One of the biggest drawbacks of adversarial training is that it only works for known adversarial attacks. Although it shows some resilience toward unknown attacks, it does not nullify misclassification or significantly improve perceptibility. Moreover, the adversarial training approach does not work against black-box attacks, which generate adversarial examples using locally trained models as the original model is trained on adversarial examples generated on the original model [34]. Adversarial training can be bypassed by applying random probabilities on an instance, and then any existing attack is performed on it [**?**].

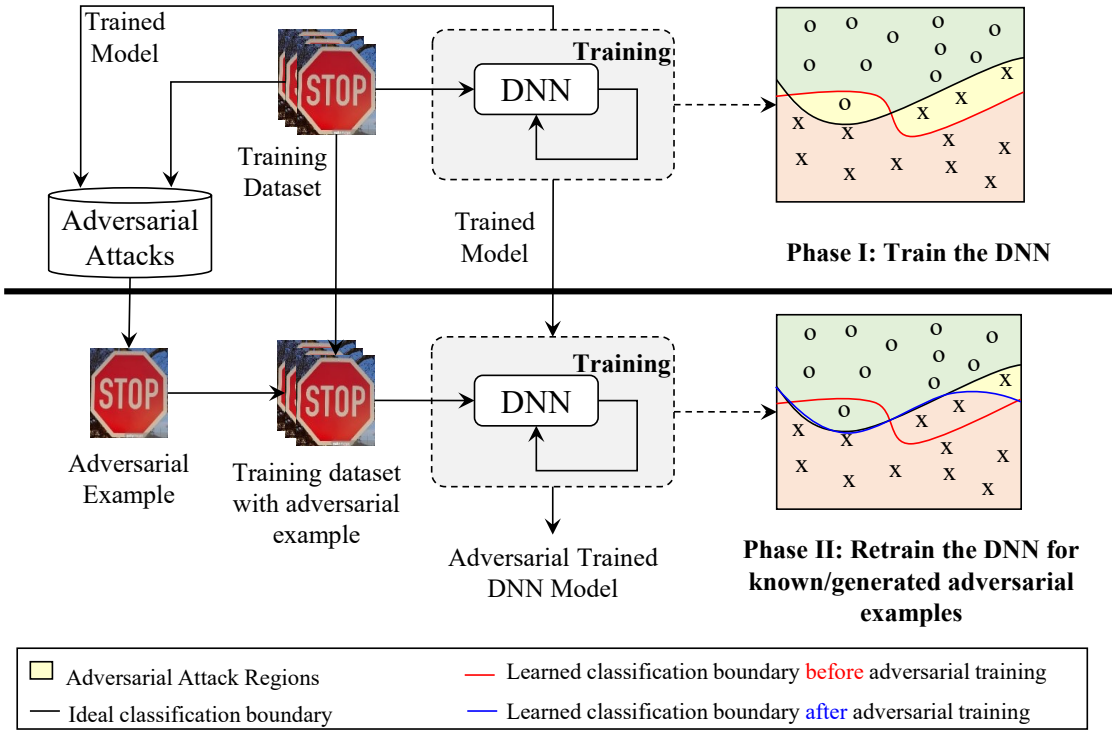

Figure 3.2: A pictorial view of methodology for adversarial training [1].

3.2.2 Gradient Masking:

Gradient-based Attacks mentioned in Section 3.1.2 use information about model parameters' gradients to generate attacks, as shown in Fig. 3.3. Such attacks can be neutralized by hiding the information about gradients, known as gradient masking or gradient hiding [6**?**].

Limitations: All gradient masking defenses mentioned in Section 3.2.2 could be fooled by learning a substitute model having gradients and generating attacks with it [34]. Moreover, any black box that does not need model parameters information can break gradient masking defense [17].

3.2.3 Defensive Distillation:

Defensive distillation is the process of using two-step ML models. First, an ML model is trained for the classification into hard and soft labels, and then the soft labels are given to the second ML model with the same architecture, as shown in Fig. 3.3. The distilled model makes output robust against adversarial examples [7, 8]. Moreover, in defensive distillation, labels are smoothed and converted into soft targets. ML model is

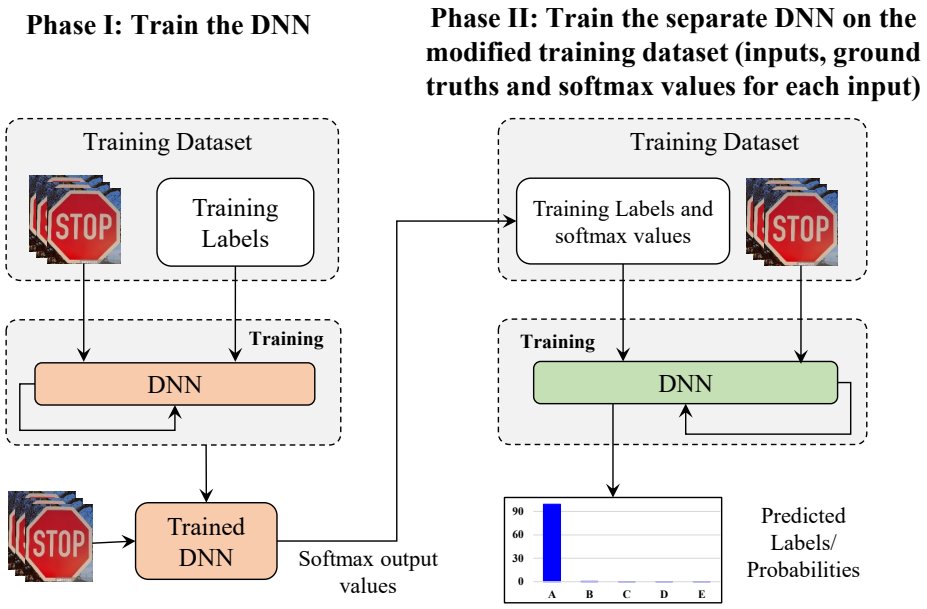

Figure 3.3: A pictorial view of the Gradient Masking and Defensive Distillation-based defenses against adversarial attacks [6**?**].

trained on these modified values. The results of defensive distillation are improved in Papernot2017.

Limitations: Recent developments in black-box attacks, i.e., score-based and decision-based attacks, can avoid defensive distillation and its smoothing method [3, 34].

3.2.4 Pre-processing-based Defenses:

In pre-processing-based defenses, input is pre-processed to filter out the perturbations that are added by the attacks [9, 10], as shown in Fig. 3.4. Some of the pre-processing-based defenses are given below.

- **Feature Squeezing:** Feature squeezing defensive technique hardens the model by using two heuristics. First, it reduces the pixel' color depth by using fewer colors to encode the colors. Second, it uses a filter to smooth multiple inputs [37].

  Limitations: This technique defends the ML model against adversarial attacks, but empirically they also reduce the accuracy of models on clean examples significantly.

- **Input transformation-based defenses:** In this type of defense, input is transformed into the manifold of training data, and the ML model is trained into it. MagNet proposed a defensive approach that uses a classifier on the last layer of ML

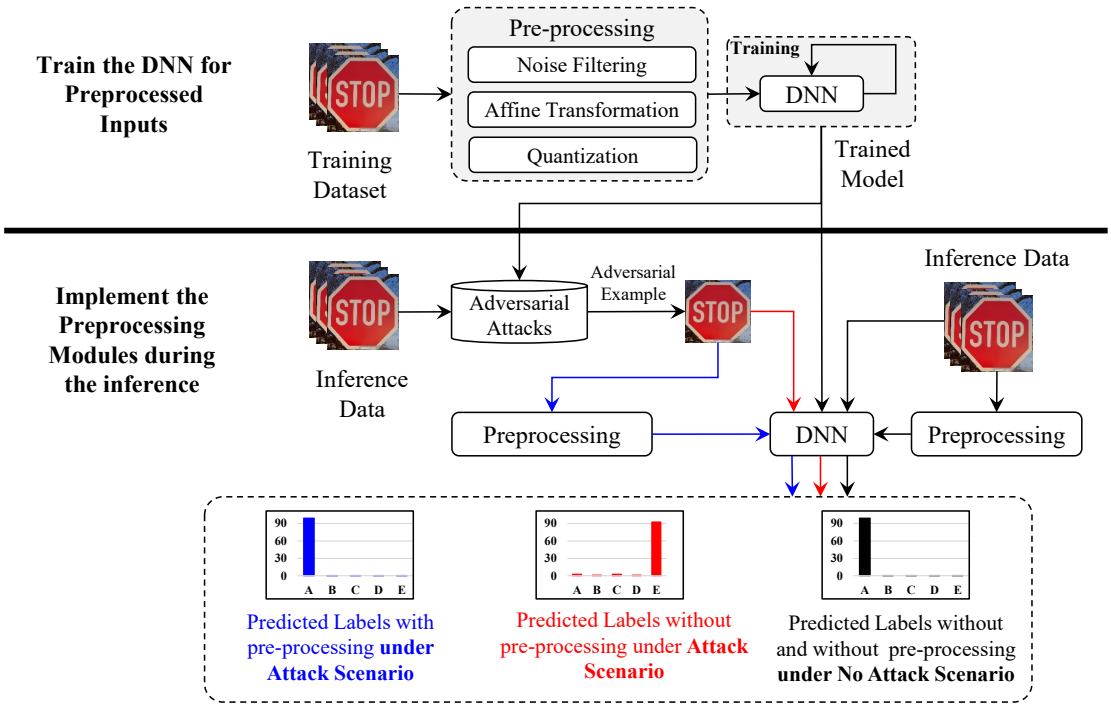

Figure 3.4: A pictorial view of the pre-processing-based defense against adversarial attacks.

model and checks whether a test example is adversarial. It checks by using a trained model how a example under test differs net from normal examples by measuring distance. If that distance is more than a specific threshold, classify it as an adversarial example and reform it using reformer [47]. APE-GAN and Defense-GAN proposed to leverage the power of GAN to reduce adversarial noise. In this, input is projected to the generator of GAN, which can differentiate between adversarial and normal examples, before sending it to the ML model.

Limitations: MagNet [47], APE-GAN [39], and Defense GAN [40] require a trained model, GANs, for detecting an example as adversarial or normal. They cannot be performed on resource-constrained applications. Moreover, MagNet defense cannot perform well in white-box attacks where attacks know all about the parameters of an ML model. For APE-GAN and Defense-GAN, GANs are difficult to train if they are not trained well. They can reduce the accuracy of the ML model significantly.

- **SScNets:** A relatively better resource-constrained defense SScNets which performs edge detection using one convolution filter and pre-processes the input using one sigmoid and one multiplication layer.

  Limitations: SScNets reduce features from an input image by extracting features

based on the filters. Experiments have shown that it can increase the robustness of the ML model on adversarial examples, but it reduces the accuracy of the ML model on clean inputs.

In summary, most of the existing defenses against adversarial attacks cannot be directly applied to decision-based attacks, especially in multi-resource decision-based attacks. Therefore, there is a dire need to develop a simple, efficient, and effective solution to defend against decision-based adversarial attacks.

Chapter 4

ParDec: Parallel Decision-based attack

In this chapter, we present and provide the detailed explanation of the proposed attack, which uses the concept of man-in-the-middle cyber attack to perform parallel-query based adversarial attack.

4.1 ParDec

The goal of our proposed methodology is to create minimum adversarial noise or perturbation, which can perform random misclassification of target image (in case of untargeted attack) or perform targeted miscalssification of target image(in case of targeted image). Pictorial view of the proposed methodology is presented in Fig. 4.1.

1. First, it chooses a reference image *reference image (I_B)*(see step 1 from 4.1), whose label is different from *target image (I_A)*, from the input(camera).

2. Second, it selects(three images in given example) multiple images having minimum perturbation (belongs to class A) with reference image (I_B)((see step 2 from 4.1)), if adversary has its own collected dataset otherwise it will skip this step.

3. Third, it applies half-interval search algorithm to find the example i.e., reference image ($I_i s$) near the classification boundary(see step 3 from 4.1). Example near classification boundary means that generated example has δ_{min} distance with the classification boundary.

4. After that, it adds perturbations from multiple sources in the generated example, both in positive and negative directions((see step 4 from 4.1)) such that label of generated image (I_is) is different from the target image (I_A).

5. Then, it applies half-interval search algorithm in parallel i.e., on multiple sources to bring the perturbed examples near classification boundary (see step 4 from 4.1).

6. It selects the example, from the perturbed examples generated in step 4, having the minimum perturbation with *target image (I_A)*(see step 5 from 4.1).

7. It again performs the step 4 and step 5 till it finds generated image (I_is) with Δ_{max} distance with target image (I_A) or the number of queries reaches to limited allowed number of queries $Qmax$ (see step 6/results from 4.1).

4.2 Mathematical Formulation of Multi-Query Attack

To compute the adversarial example, we use the cost function from the state-of-the art decision-based attack(FaDec) which is the improved version of the cost function defined by [3], which is given as.

$$cost = c \times (f(X_{adv})! = f(X_{target})) + \sum (X - X_{adv})^2 \tag{4.2.1}$$

X_{adv}, X_{target} and c are adversarial image, targeted image and constant. The reason for selecting this cost is that in this perturbation norm is minimized between target image, X_{target} and adversarial image, X_{adv}. we choose c as 1 in 4.2.1, because large value of c increases convergence time. The gradients of the cost function given below. if the updated adversarial example does not belong to the target class then $X = \frac{X_{target}+X_{adv}}{2}$ and the computed gradient is.

$$\frac{\partial cost}{\partial X_{adv}} = X_{adv} - X_{target} \tag{4.2.2}$$

If the instant adversarial example belongs to targeted class then $X = X_{target}$ and the computed gradient is.

$$\frac{\partial cost}{\partial X_{adv}} = 2 \times (X_{adv} - X) \tag{4.2.3}$$

For the gradients of the cost function in 4.2.2 and 4.2.3 , the new adversarial gradient update will be:

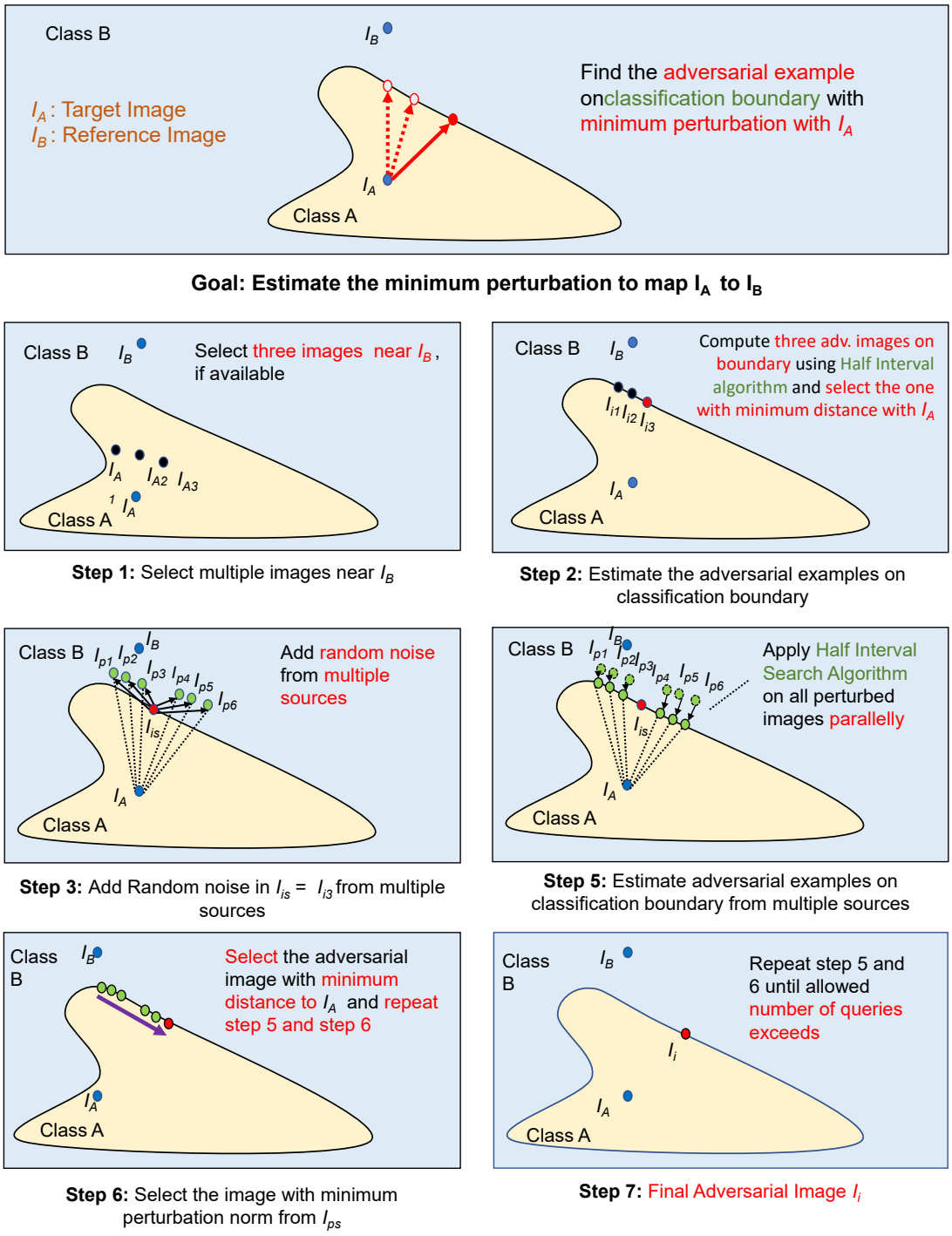

Figure 4.1: Visualization of the step-by-step methodology of the proposed ParDec.

$$X_{adv,new} = X_{adv,old} - \alpha \times \frac{\partial cost}{\partial X_{adv}} \tag{4.2.4}$$

The cost function mentioned in 4.2.4 searches the adversarial example linearly on classification boundary. This cost function is updated by —— To compute the adversarial

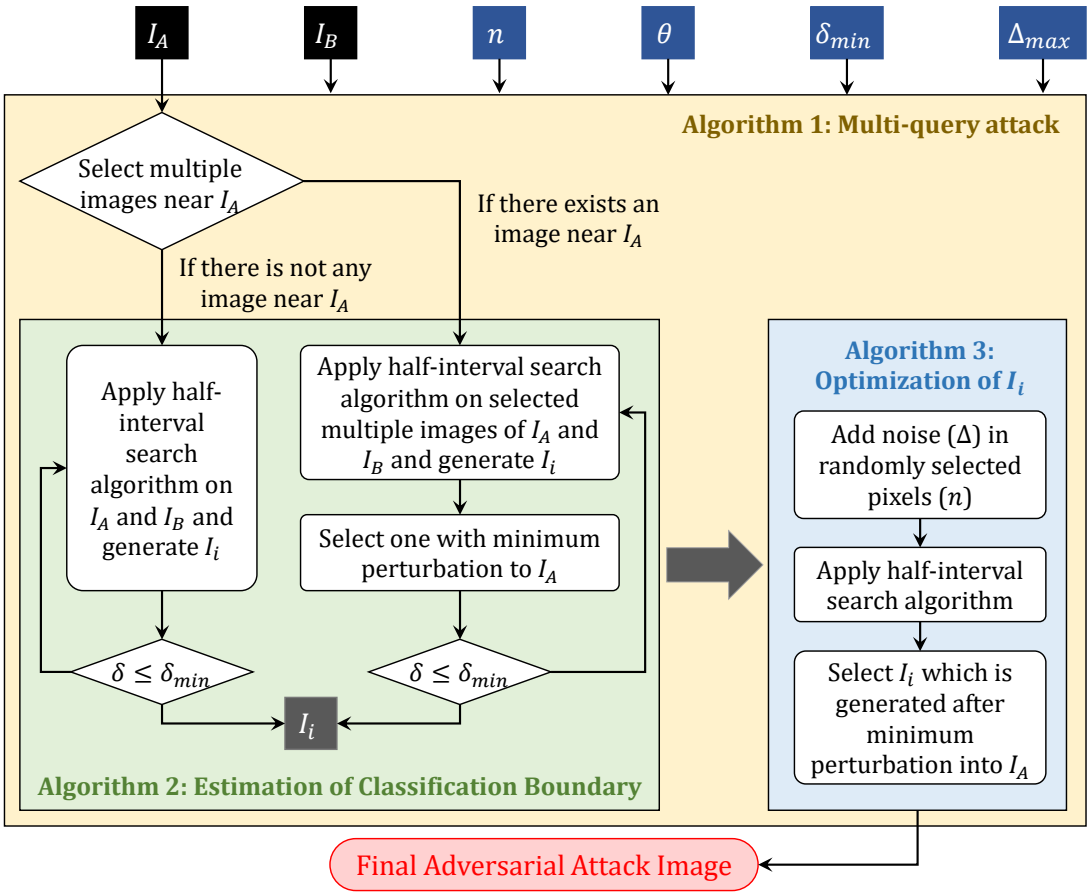

Figure 4.2: Algorithmic flow of the proposed multi-query attack

example, we use the cost function mentioned in the state-of-the art decision-based attack(FaDec).

$$cost = \sum (X_{adv} - X)^2 \tag{4.2.5}$$

The cost function mentioned in 4.2.4 use half-interval search algorithm shown in 2. When we apply half-interval search algorithm, adversarial example generated linearly causes infinite oscillation at the classification boundary. To address this problem, this cost function uses maximum allowed perturbation δ_{min} near classification boundary 2. To optimize this cost function by the use of stochastic Zeroth-Order Optimization. Firstly, n number of pixels in X_{adv} are selected. Secondly, random perturbations are added in the selected pixels to calculate $\bar{X}_{adv}$. The zeroth-order gradient for 4.2.5 follows.

$$\frac{\partial cost}{\partial X_{adv}} = \frac{\sum (X_{adv} - X)^2 - \sum (\bar{X}_{adv} - X)^2}{X_{adv} - \bar{X}_{adv}} \tag{4.2.6}$$

$$X_{adv,new} = X_{adv,old} - \lambda \times \frac{\partial cost}{\partial X_{adv}} \tag{4.2.7}$$

Algorithm 1 Attack flow of the proposed ParDec Attack

Input:
I_A = Target image; I_B = Reference image;
Δ_{max} = Maximum tolerable l_2 square distance;
Q_{max} = Query restrictions; n = Number of pixels to be perturbed;
θ = Perturbation in a pixel; δ_{min} = maximum perturbation value;
Output:
I_i = Adversarial Image;

1: Select multiple images near I_A
2: Compute $I_is = I_{iselect}$ using Algo.2;
3: **repeat**
4: Update $I_i = I_{it}$ using Algo. 3;
5: **until** $(\sum(I_A - I_i)^2 > \Delta_{max})$ & $Q \leq Q_{max}$

"λ" is the factor that controls the jump in

4.3 Estimating the Adversarial Example I_i on the Classification Boundary

First, we outline the problem statement of estimating the sample near classification boundary in the given below goal.

Goal: Suppose I_A, I_B and δ_{min} are the target image (class: A), random (reference) image (class: other than A) and maximum allowed perturbation margin. The goal of this given algorithm is to generate a perturbed image I_{is} near classification boundary having tolerable δ_{min} distance from the classification boundary. It uses I_As, if available otherwise will it use I_A as input to the image. Mathematically, it can be defined as:

$$\exists\ I_{is} : f(I_{is})\ \neq\ f(I_A)\ \wedge\ max(I_{is} - I_A)\ \leq\ \delta_{min} \tag{4.3.1}$$

This algorithm tries to find the I_is near the classification boundary having tolerable δ_{min} distance from the classification boundary. To this, it calculates I_pi by computing the mean image between *target image* I_A and *reference image* I_B (see line 1 in 2). If the label of *generated image* I_pi is equal to I_A it replace the I_B and if the label of *generated image* I_pi is equal to I_B it replace the I_A. (see line 5-8). The algorithm performs this process iteratively until maximum distance of I_is from the I_A is less than δ_{min} while $f(I_{is})$ is not equal to $f(I_A)$.

Algorithm 2 Estimating the sample image near Classification Boundary

Input
I_A = Target image; OR I_{As} = Selected images near I_A
I_B = Reference image; δ_{min} = Max. Allowed Perturbation;
Output
I_{si} = Adversarial Image;

1: $I_{si} = \frac{I_{Ai}+I_B}{2}$;
2: **repeat**
3: $label_i = f(I_{Ai})$;
4: Q(query) = Q(query) +1
5: **if** $label_i \equiv f(I_{Ai}$ **then**
6: $I_{Ai} = I_{si}$;
7: **else**
8: $I_B = I_{si}$;
9: $\delta = max(I_A - I_{si})$;
10: **until** $\delta_i \leq \delta_{min}$
Select I_{is} with $min.(I_A - I_{si})$

} in parallel for i

4.4 Optimize the perturbed image I_is on the Classification Boundary

Goal: Suppose I_A, I_B and δ_{min} are the target image (class: A), random(reference) image (class: other than A) and maximum allowed perturbation margin. The goal of the given algorithm is to minimize the distance of the perturbed image I_i to the target image I_A, while label of I_i is not equal to I_A, to make sure imperceptibility of generated adversarial image I_is. Mathematically, it can be defined as:

$$\forall\ I_i\ min(I_i - I_A) : f(I_i)\ \neq f(I_A) \tag{4.4.1}$$

For this, this algorithm starts with initializing I_0 image by setting all pixels to zero(see line 1 in 3). After that, it randomly selects n pixels from I_0 and set them to maximum value. After that it adds perturbation in in positive and negative direction in I_i parallel process, to generate multiple perturbed images(see line 3-6 in 3). After that, half interval search from 2 is applied in parallel process to bring perturbed images near to classification boundary.(see line 8 2). Moreover, the magnitude of added random noise is decreased by half after 10 queries to converge to minima on classification boundary(see line 7 in 3). Finally, from all generated images I_it it chooses the image with minimum distance with target image I_A.

All above-mentioned algorithms are integrated to make final adversarial example I_i shown in 1. First, it computes I_s using 2. Then it performs 3 iteratively to optimize

the added perturbation until ($\sum(I_A - I_i)^2 > \Delta_{max}$)) and number of allowed queries crosses to Q_{max}.

Algorithm 3 Moving Adversarial examples to min. perturbation

Input
I_A = Target image;
n = Number of pixels to perturb;
θ = Relative Perturbation in each pixel;
$factor$ = random noise factor;
$scale_p = [1, 50, 100, 1, 50, 100]$
Output
I_{is} = Adversarial Image;

Define a zeroed I_0 of size I_i;
Choose n number of pixels in I_0 and equal them to the maximum values of the pixel;
if $t < thresh.$ **then**
    $I_{pt} = I_i + factor * scale_p[t]$;
else
    $I_{pt} = I_i - factor * scale_p[t]$;
$factor = \frac{factor}{2}$; after 10 queries
Update $I_i t = I_{i2}$ using Algo. 2;
Select I_{it} with $min.(I_A - I_{it})$;

} in parallel for t

Chapter 5

Experimental Results for Proposed Attack

5.1 Experimental setup

To evaluate the proposed Multi-query attack, we performed multiple un-targeted attacks using the experimental setup given below.

1. **Datasets:** GTSRB, CIFAR-10

2. **DNN for GTSRB:** Lambda (lambda t: t/255.0 - 0.5) Conv2D(3, 1x1) - Conv2D(16, 5x5, (2, 2)) - Conv2D(32, 3x3) - MaxPool2D((2, 2), (2, 2)) - Conv2D(64, 3x3) - Conv2D(128, 3x3) - Flatten() - Dropout (0.4) - Dense (128) - Dropout(0.7) - Dense(34) – Dropout - softmax()

3. **DNN for CIFAR-10 (cifar10vgg):** Conv2D(64, 3x3) - Conv2D(64, 3x3)- MaxPool2D((2, 2)) - Conv2D(128, 3x3) - Conv2D(128, 3x3) - MaxPool2D((2, 2))- Conv2D(256, (3, 3))- Conv2D(256, (3, 3))- Conv2D(256, (3, 3))- MaxPool2D((2, 2)) -Conv2D(212, (3, 3))- Conv2D(512, (3, 3)) -Conv2D(512, (3, 3))-MaxPool2D((2, 2)) - Dense(512) - Dense(10) - Softmax()

4. **Training parameters for DNN (DNN for CIFAR-10):** Epoch = 250; Batch Size = 128; Activation = relu, Optimizer = Adam; Learning Rate = 0.0001; Decay = 1×10^{-6}.

5. **Number of parallel sources** : 6 sources are used in given expermental setup.

5.2 Evaluation Parameters

To evaluate the effect of hyper parameters on Multi-query attack, we use the following parameters and their values are given below:

1. δ_{min} is the maximum allowed perturbation in the target image I_A i.e., the distance between the actual classification boundary and the estimated classification boundary. It is calculated as the maximum distance between two samples in algorithm 2 (line 9). Note, we vary is from 1 to 15, i.e., 1, 5, 10, and 15.

2. n is the number of random pixels to be perturbed for introducing perturbation in the initial adversarial image I_i in algorithm 2(line 2). Note, we vary is from 5 to 50, i.e., 5, 10, 20, 30 and 50.

3. θ is the relative perturbation in each pixel i.e., magnitude of the perturbation in each pixel randomly selected. Note, we vary is from 0.0392 to 0.1962, i.e., 0.0196, 0.0392, 0.1176, 0.1962. These are same values used in FaDec attack paper [17].

5.2.1 Metrics of Imperceptibility Evaluation

To evaluate the imperceptibility, we use the evaluation metrics given below.

1. **Perturbation Norm (d)** is most commonly used parameter and it measures the mean square difference between two images (adversarial image and clean input image). Note, lower the value of perturbation norm higher the imperceptibility (explained in Section 2.2).

2. **Cross Co-relation Coefficient (CC)** is the probability of the linear dependencies between two images. It is calculated by using the Pearson's correlation coefficient. It has range between 0 and 1; "0" for minimum human imperceptibility and "1" for maximum human imperceptibility (explained in Section 2.2).

3. **Structural Similarity Index (SSIM)** is perceptual similarity between two given images (reference image and perturbed image). It has range between 0 and 1. For maximum human imperceptibility, SSIM should be "1" (explained in Section 2.2).

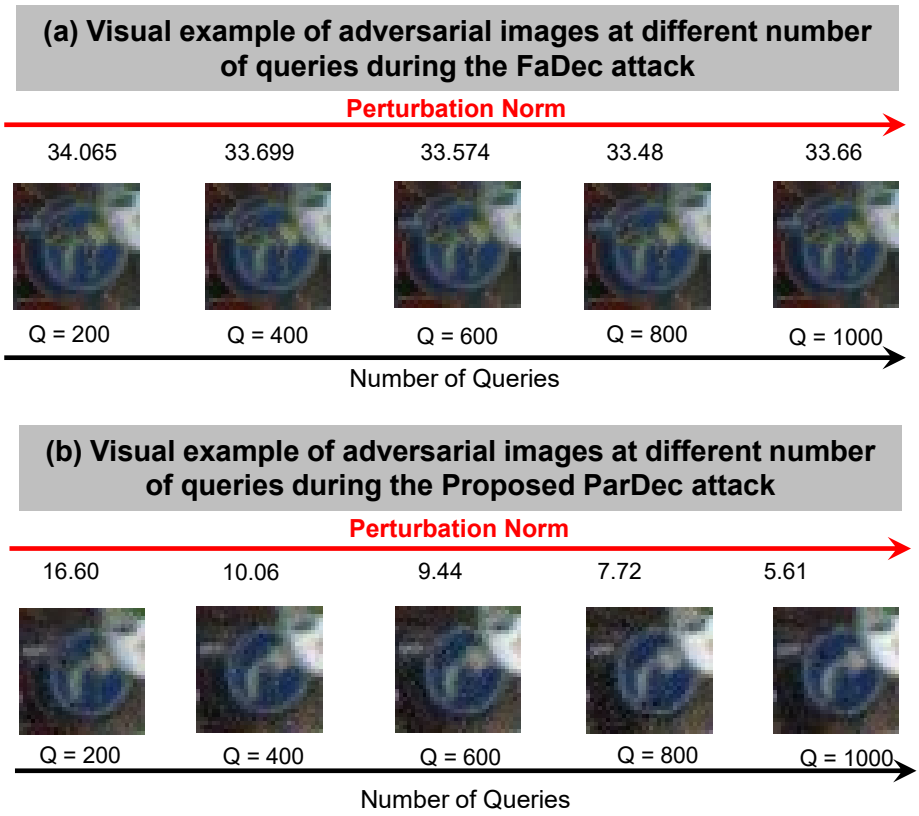

Figure 5.1: Visual example of adversarial images at different number of queries during the decision-based adversarial attacks, i.e., FaDec and ParDec.

5.2.2 Evaluation and Discussion

Number of Queries (n) Multi-query attack and FaDec attack are evaluated against same sample images and evaluation parameters from Section 5.2 for 100000 queries shown in Fig. 5.3. It can be shown that Multi-query attack converges at 10000 queries, while FaDec is still converging after 100000 queries. It can be shown that Multi-query attack converges approx. 10 times faster than FaDec which is the state-of-the art decision-based attack [17]. Fig. 5.1 shows that perceptibly of the adversarial noise in the case of ParDec is far less than the in the case of FaDec attack. Hence, it shows that the proposed Pardec attack converges approximately, 5× than to the fastest state-of-the-art attack (FaDec).

Max. allowed perturbation(δ_{min}) Multi-query attack and FaDec attack are evaluated against same sample images and δ_{min} is changed except all other evaluation parameters as shown in fig. **??**. It can be shown as the value of δ_{min} is increased perturbation norm(d) of adversarial image increases, which results in the decreasing the quality of

Figure 5.2: Different examples attack cases, e.g., proposed ParDec and state-of-the-art FaDec attacks on different samples of dataset.

adversarial image. This is due to the fact that increasing the δ_{min} increases the margin to add perturbation in target image which allows Multi-query attack to converge at higher perturbation. Same results in observed for FaDec.

Max. allowed perturbation(θ) Multi-query attack and FaDec attack are evaluated against same sample images and θ is changed except all other evaluation parameters as shown in fig. 5.3.By increasing θ trend similar to increasing the δ_{min} is observed. An optimal value of θ for GTSRB dataset is 20. Moreover, smaller values of *theta* gives stable convergences.

Key Insights By analyzing all the results in Figs. 5.3 and 5.4, we made the following key observations:

1. ParDec is achieving significantly higher (more than 400%) imperceptibility with 5x a smaller number of queries. Hence, it can be unimplemented on a very resource-constrained devices, like battery-operated edge device in IoT.

2. It is also observed that targeted attack takes more time to converge as compared to un-targeted attack.

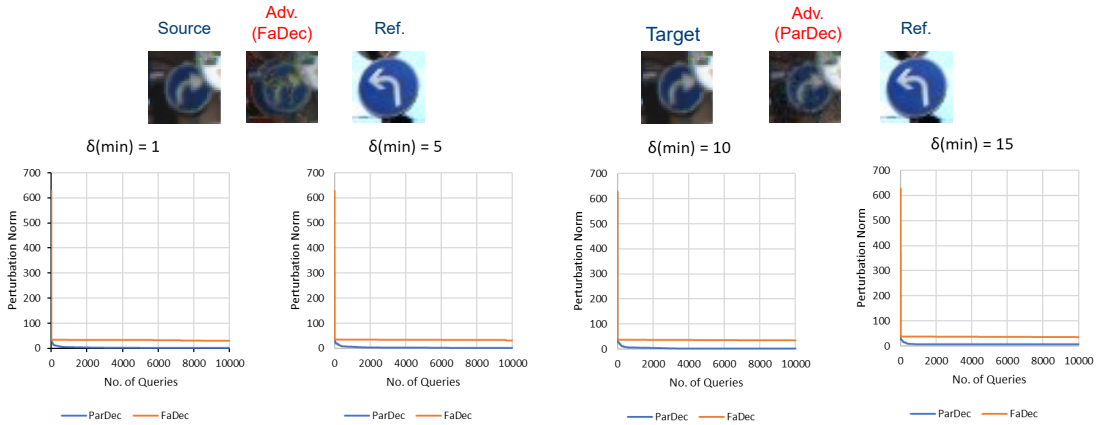

Figure 5.3: The effect of changing the $\delta_m in$ on the convergence of the proposed ParDec attack and the state-of-the-art FaDec attack. These results show that in all cases the ParDec is converging much faster than FaDec, and achieving very high imperceptibility (low perturbation norm.)

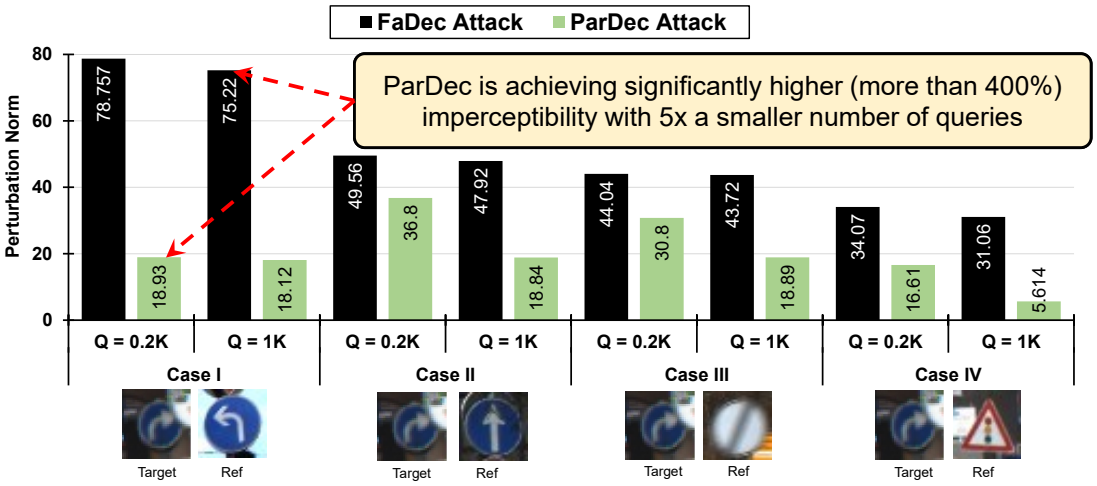

Figure 5.4: Experimental results to compare the the number of quesries required to converge the proposed attack and the fastest state-of-the-art attack, i.e., FaDec.

3. Increasing the number of parallel sources can further decrease the number of queries.

Imperceptibility Imperceptibility of different adversarial images as number of queries increases is shown in Fig. 5.3. In this figure, it is observed that in all cases the ParDec is converging much faster than FaDec, and achieving very high imperceptibility (low perturbation norm). The detailed results of ParDec for the 50 percent of the GTSRB dataset are given in the Appendix A.

Chapter 6

RaFiS: Random filter Switching-based Defense

In this chapter, we present and provide the detailed explanation of the proposed defense against decision-based adversarial attacks, which randomly switches the existing pre-processing-based defense.

6.1 RaFiS

To counter the limitations of the mentioned state-of-the-art defenses, we propose to use random filters switching as pre-processing step. The methodology to use random filters is explained below.

1. Suppose X is an colored image with three channels $X = [x^0, x^1, ...x^k]$, where x^k is the number of cahnnel of the input image, $k = 2$ in aboce mentione case. A filter F used to extract features will have the form.

$$F = [f^0, f^1, ..., f^k] \tag{6.1.1}$$

   where f^k is the *k-th* channel of the filter.

2. First, convolution is applied between image and the randomly selected filter F for extracting the edges of the input image.

$$E = X \circledast F \tag{6.1.2}$$

E is a gray-scale image of extracted edges X.

3. Second, strong edges are selected from the E from 6.1.2 for a threshold, t_k. All the edges less than the t_k will be set to 0 while other will be set to 1 i.e., ignoring the weaker edges and selecting the stronger edges. Sigmoid function(σ) is used to select stronger edges and is given below.

$$\sigma(x) = \frac{1}{1 + e^{-}x} \tag{6.1.3}$$

The output with stronger edges S is obtained after applying threshold on 6.1.3 and is given below.

$$S = \sigma(E, t_k) \tag{6.1.4}$$

4. After that, the output S from 6.1.4 is multiplied by with the input image channel-wise to preserve color information.

$$I = X \times S \tag{6.1.5}$$

5. Finally, I is given as input to the ML model.

6. It is important to mention that filter F is selected randomly from the set of filters consists of multiple filters e.g. laplacian, high-pass and low-Pass filters etc.

Chapter 7

Experimental Results for Proposed Defense

7.1 Experimental setup

To evaluate the Random Filter-Switching-based Defense, we performed multiple untargeted attacks using the experimental setup given below.

1. **Datasets:** GTSRB, CIFAR-10
2. **Attacks :** Multi-query attack and FaDec atatck
3. **DNN for GTSRB:** Lambda (lambda t: t/255.0 - 0.5) Conv2D(3, 1x1) - Conv2D(16, 5x5, (2, 2)) - Conv2D(32, 3x3) - MaxPool2D((2, 2), (2, 2)) - Conv2D(64, 3x3) - Conv2D(128, 3x3) - Flatten() - Dropout (0.4) - Dense (128) - Dropout(0.7) - Dense(34) – Dropout - softmax()
4. **DNN for CIFAR-10 (cifar10vgg):** Conv2D(64, 3x3) - Conv2D(64, 3x3)- MaxPool2D((2, 2)) - Conv2D(128, 3x3) - Conv2D(128, 3x3) - MaxPool2D((2, 2))- Conv2D(256, (3, 3))- Conv2D(256, (3, 3))- Conv2D(256, (3, 3))- MaxPool2D((2, 2)) -Conv2D(212, (3, 3))- Conv2D(512, (3, 3)) -Conv2D(512, (3, 3))-MaxPool2D((2, 2)) - Dense(512) - Dense(10) - Softmax()
5. **Set of Filters:** Three filters in set of filters, which are Laplacian, High-Pass Filter and Low-Pass Filter.
6. **Number of parallel sources** : 6 sources are used in given expermental setup.

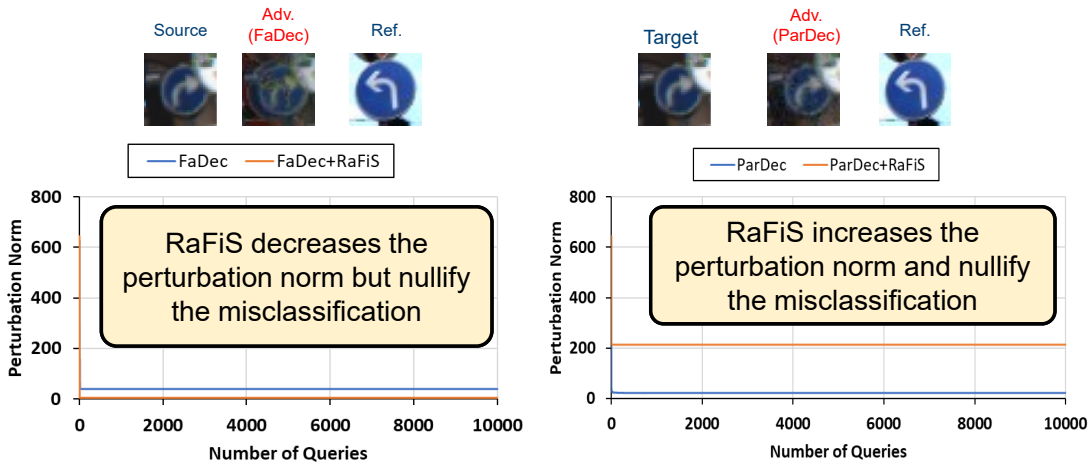

Figure 7.1: Experimental evaluation of the proposed RaFiS, which shows it decreases the perturbation norm in FaDec but it increases the perturbation norm in ParDec case.

7.2 Evaluation Parameters

To evaluate the Random Filter-Switching-based Defense, we use the following parameters.

1. **Threshold for edges(t_k):** is the threshold for selecting the edges on the convolved image i.e., edges less than t_k will be "0" and others will be set to "1".

7.2.1 Evaluation Metrics for defense success

To evaluate out proposed defense, we use the following the metrics.

1. **Label changing of Adversarial Image:** if the label of the adversarial image is same as the target image after applying the attack, it is defined success of the defense. For a successful defense, label changing of adversarial image should be "0".

2. **Number of queries:** If the given defense prevents the attack to generate adversarial example, which can misclassify the target image, in the given number of queries is considered as the successful defense.

3. **Perturbation Norm(d):** If a defense fails to prevent label changing of the adversarial but if restricts the perturbation norm to the value which is easily perceptible to human eye can be considered as the successful attack.

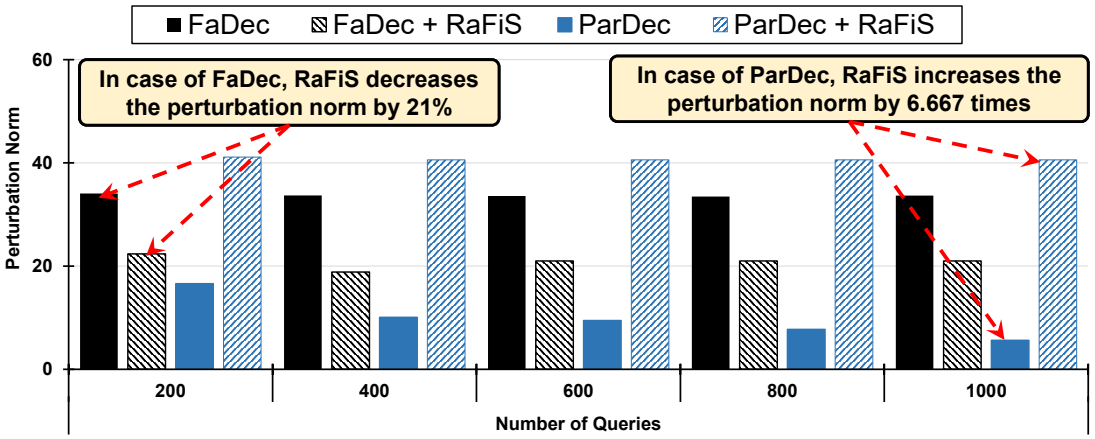

Figure 7.2: This experimental results show the impact of RaFiS on perturbation norm during the attack with respect to number of queries. From analyzing this, we observe that in case of FaDec, RaFiS decreases the perturbation norm by 21% but in case of ParDec, RaFiS increases the perturbation norm by 6.667 times.

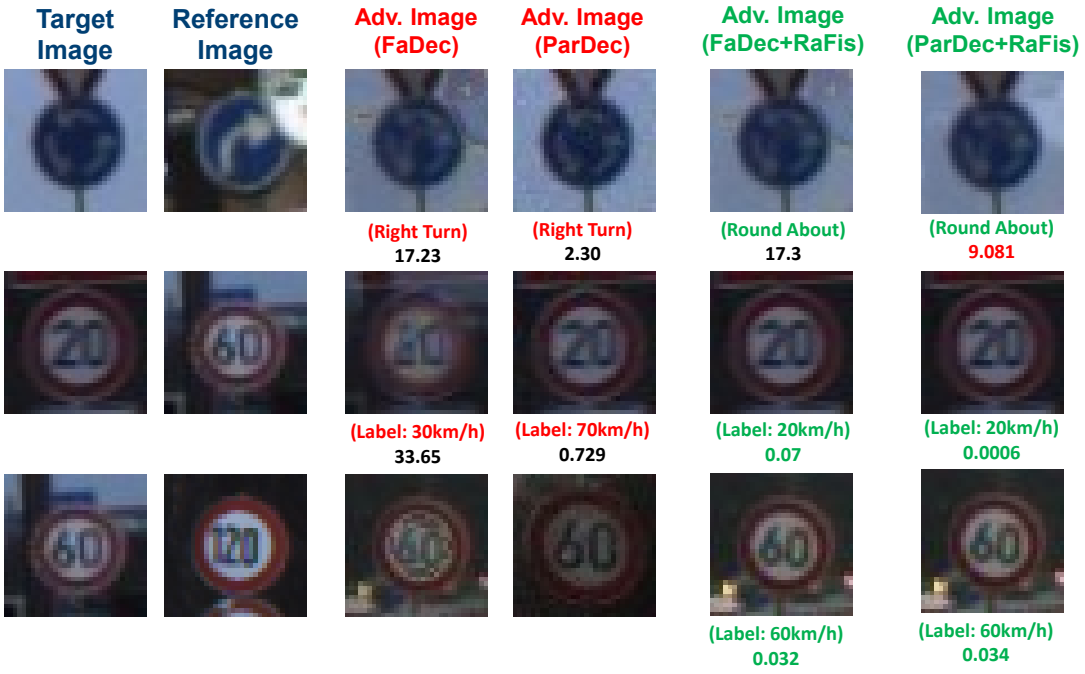

Figure 7.3: Different examples of defense agaisnt multiple attack cases, e.g., proposed ParDec and state-of-the-art FaDec attacks on different samples of dataset.

7.2.2 Evaluation and Discussion

All the attacks performed in section 5.1 are tested against the Random Filter-Switching-based Defense and the results are discussed below.

1. Multi-query attack is performed on same sample images but defended by Random Filter Switching based defense, and evaluation parameters from section 5.2 for 100000

queries shown in Fig. 7.1. It can be seen that performing Multi-query attack converges at perturbation norm of 39, which is approximately 20 times more than without defense, while keeping the label of adversarial image same as target image, which shows the success of our proposed. It prevents the label changing perturbed image and resists the decrease in the perturbation norm.

2. FaDec attack is performed on same sample images but defended by Random Filter Switching based defense, and evaluation parameters from section 5.2 for 100000 queries shown in fig. 7.1. It can be seen that that FaDec attack converges at perturbation norm of 21, which is twice than the one without defense, while keeping the label of the adversarial image same as target image.

Key Insights

1. For using random filtering in each query, the perturbation added by the attack is filtered out in each iteration of attack. In start, perturbation norm decreases significantly because the magnitude of the added perturbation norm is large initially and reduces as the iterations of attack increases.

2. If high pass filter is selected form the set of filters it performs sharpening of the edges which is further refined by the edge triggering threshold t_k. This helps to remove the perturbation added near the edges of the target image.

3. If Laplacian filter is selected, it performs the same process of edge detection as high pass filter. It removes the perturbation added near the edges of in the target image.

4. If the low pass filter is selected from the set of filters, it will reduce the perturbation in the areas of adversarial image other than edges as it smooths the image.

5. Our proposed defense evaluated on GTSRB and CIFAR-10 and it successfully defended the test images without changing the label of adversarial images to 100000 queries.

It is important to note that the increment in the number of filters in the pre-processing layer increases the attack complexity but it also increase the cost.

Conclusion

In this thesis, we first developed a novel approach to perform a Multi-query attack. It finds initial adversarial image near classification boundary by using half interval search algorithm and optimize the added perturbation in initial adversarial image from multiple sources by using half interval search-based algorithm in parallel to converge faster and selects the adversarial image based on perturbed norm with target Image. We evaluate it on multiple samples from GTSRB and CIFAR-10 datasets. Multi-query attack converges 10x faster when compared to the state-of-the art decision-based attack, i.e. FaDec [17]. For Perturbation norm, we showed that Multi-query attack converges to low perturbation norm as compared to FaDec, which provides high human imperceptibility for adversarial image. Moreover, we also proposed a Random Filter-Switching-based Defense against decision-based adversarial attacks. It uses the mechanism of switching filters randomly in pre-processing step of DNN inference. For each query, it selects a filter randomly form the set of pre-defined filters and performs convolution and thresholding on the input image to reduce the noise or make the noise perceptible. We evaluate Random Filter-Switching-based Defense on Multi-query attack and FaDec for multiple samples of GTSRB and CIFAR-10. We showed that Random Filter-Switching-based Defense prevents the label changing of the adversarial image and resists the decrease in perturbation norm on Multi-query attack and FaDec attack.

References

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

[2] Faiq Khalid, Muhammad Abdullah Hanif, and Muhammad Shafique. Exploiting vulnerabilities in deep neural networks: Adversarial and fault-injection attacks. *arXiv preprint arXiv:2105.03251*, 2021.

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

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

[5] Faiq Khalid, Muhammad Abdullah Hanif, Semeen Rehman, Rehan Ahmed, and Muhammad Shafique. Trisec: Training data-unaware imperceptible security attacks on deep neural networks. In *2019 IEEE 25th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, pages 188–193. IEEE, 2019.

[6] Ian Goodfellow. Gradient masking causes clever to overestimate adversarial perturbation size. *arXiv preprint arXiv:1804.07870*, 2018.

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

[8] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and

Debdeep Mukhopadhyay. A survey on adversarial attacks and defences. *CAAI Transactions on Intelligence Technology*, 6(1):25–45, 2021.

[9] Hassan Ali, Faiq Khalid, Hammad Ali Tariq, Muhammad Abdullah Hanif, Rehan Ahmed, and Semeen Rehman. Sscnets: Robustifying dnns using secure selective convolutional filters. *IEEE Design & Test*, 37(2):58–65, 2019.

[10] Faiq Khalid, Muhammad Abdullah Hanif, Semeen Rehman, Junaid Qadir, and Muhammad Shafique. Fademl: Understanding the impact of pre-processing noise filtering on adversarial machine learning. In *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 902–907. IEEE, 2019.

[11] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, pages 15–26, 2017.

[12] Nina Narodytska and Shiva Prasad Kasiviswanathan. Simple black-box adversarial perturbations for deep networks. *arXiv preprint arXiv:1612.06299*, 2016.

[13] Thomas Brunner, Frederik Diehl, and Alois Knoll. Copy and paste: A simple but effective initialization method for black-box adversarial attacks. *arXiv preprint arXiv:1906.06086*, 2019.

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

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

[16] Minhao Cheng, Thong Le, Pin-Yu Chen, Jinfeng Yi, Huan Zhang, and Cho-Jui Hsieh. Query-efficient hard-label black-box attack: An optimization-based approach. *arXiv preprint arXiv:1807.04457*, 2018.

[17] Faiq Khalid, Hassan Ali, Muhammad Abdullah Hanif, Semeen Rehman, Rehan Ahmed, and Muhammad Shafique. Fadec: A fast decision-based attack for adver-

sarial machine learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.

[18] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018.

[19] Chen Zhu, W Ronny Huang, Hengduo Li, Gavin Taylor, Christoph Studer, and Tom Goldstein. Transferable clean-label poisoning attacks on deep neural nets. In *International Conference on Machine Learning*, pages 7614–7623. PMLR, 2019.

[20] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. *Advances in neural information processing systems*, 31, 2018.

[21] Ahmed Salem, Rui Wen, Michael Backes, Shiqing Ma, and Yang Zhang. Dynamic backdoor attacks against machine learning models. *arXiv preprint arXiv:2003.03675*, 2020.

[22] Keita Kurita, Paul Michel, and Graham Neubig. Weight poisoning attacks on pretrained models. *arXiv preprint arXiv:2004.06660*, 2020.

[23] Jianbo Chen, Michael I Jordan, and Martin J Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. In *2020 ieee symposium on security and privacy (sp)*, pages 1277–1294. IEEE, 2020.

[24] Nathan Inkawhich, Matthew Inkawhich, Yiran Chen, and Hai Li. Adversarial attacks for optical flow-based action recognition classifiers. *arXiv preprint arXiv:1811.11875*, 2018.

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

[26] Aishan Liu, Xianglong Liu, Jiaxin Fan, Yuqing Ma, Anlan Zhang, Huiyuan Xie, and Dacheng Tao. Perceptual-sensitive gan for generating adversarial patches. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 1028–1035, 2019.

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

[28] Xin Liu, Huanrui Yang, Ziwei Liu, Linghao Song, Hai Li, and Yiran Chen. Dpatch: An adversarial patch attack on object detectors. *arXiv preprint arXiv:1806.02299*, 2018.

[29] Eric Wong, Frank Schmidt, and Zico Kolter. Wasserstein adversarial examples via projected sinkhorn iterations. In *International Conference on Machine Learning*, pages 6808–6817. PMLR, 2019.

[30] Amin Ghiasi, Ali Shafahi, and Tom Goldstein. Breaking certified defenses: Semantic adversarial examples with spoofed robustness certificates. *arXiv preprint arXiv:2003.08937*, 2020.

[31] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*, pages 372–387. IEEE, 2016.

[32] Muhammad Shafique, Mahum Naseer, Theocharis Theocharides, Christos Kyrkou, Onur Mutlu, Lois Orosa, and Jungwook Choi. Robust machine learning systems: Challenges, current trends, perspectives, and the road ahead. *IEEE Design & Test*, 37(2):30–57, 2020.

[33] Jeff Jun Zhang, Kang Liu, Faiq Khalid, Muhammad Abdullah Hanif, Semeen Rehman, Theocharis Theocharides, Alessandro Artussi, Muhammad Shafique, and Siddharth Garg. Building robust machine learning systems: Current progress, research challenges, and opportunities. In *Proceedings of the 56th Annual Design Automation Conference 2019*, pages 1–4, 2019.

[34] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519, 2017.

[35] Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE, 2018.

[36] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and J Doug Tygar. Adversarial machine learning. In *Proceedings of the 4th ACM workshop on Security and artificial intelligence*, pages 43–58, 2011.

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

[38] Valentina Zantedeschi, Maria-Irina Nicolae, and Ambrish Rawat. Efficient defenses against adversarial attacks. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 39–49, 2017.

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

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

[41] Antreas Antoniou, Amos Storkey, and Harrison Edwards. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017.

[42] Adnan Siraj Rakin, Jinfeng Yi, Boqing Gong, and Deliang Fan. Defend deep neural networks against adversarial examples via fixed and dynamic quantized activation functions. *arXiv preprint arXiv:1807.06714*, 2018.

[43] Faiq Khalid, Hassan Ali, Hammad Tariq, Muhammad Abdullah Hanif, Semeen Rehman, Rehan Ahmed, and Muhammad Shafique. Qusecnets: Quantization-based defense mechanism for securing deep neural network against adversarial attacks. In *2019 IEEE 25th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, pages 182–187. IEEE, 2019.

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

[45] Chunchuan Lyu, Kaizhu Huang, and Hai-Ning Liang. A unified gradient regularization family for adversarial examples. In *2015 IEEE international conference on data mining*, pages 301–309. IEEE, 2015.

[46] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[47] Dongyu Meng and Hao Chen. Magnet: a two-pronged defense against adversarial examples. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pages 135–147, 2017.

Appendices

Appendix A

To compute overall accuracy or success rate of our attack on given GTSRB dataset, we evaluate our proposed attack ParDec on 50 percent of GTSRB dataset. We select image from each class and mapped it on all other classes (which 42 in every case), e.g., we select class1 which is 20 Kmph from dataset and mapped it into all other classes except class1. For valid adversarial images, we set **perturbation norm <= 15** and **number of queries**=200 for generated adversarial images in this experimental setup. We calculate accuracy or success rate of the attack on for each class by taking average of the accuracy, when of the target class is mapped onto all other classes. After that, we take the mean of the accuracy for each target class, which was calculated in previous step. For example, for each class, we calculate the accuracy of the selected class 1 against all other classes and take its mean. For the given dataset which has 43 class of road signs, we take mean of 42(one less than the overall classes) accuracy values for target class. When we calculate accuracy of all classes, i.e., 43, we have 43 accuracy values. Finally we take the mean of these 43 values to compute the overall accuracy. All above mentioned, steps are performed on FaDec and ParDec attack and their comparison is given in the table A.1. From the Table A.1, it can be seen that ParDec achieves 2.7 times more accuracy than FaDec, state-of-the-art decision-based attack.

Table A.1: Accuracy or Success Rate Comparison FaDec vs ParDec

FaDec		**ParDec**	
Target Class	**Accuracy(%)**	**Target Class**	**Accuracy(%)**
0	50	0	86.03
1	0	1	0.7
2	7.5	2	28.1
3	0	3	0.3
4	0	4	0.4
5	0	5	38.1
6	2.5	6	25.4
7	30.7	7	68.5
8	25	8	49
9	0	9	14.6
10	17	10	39
11	8	11	20
12	32.2	12	73.1
13	0	13	0
14	0	14	0.8
15	12.3	15	37.5
16	0	16	20
17	0	17	0
18	0	18	1
19	0	19	8
20	0	20	19.7
21	7.1	21	22.7
22	9.5	22	28.1
23	5.5	23	14.6
24	27.8	24	76
25	11.1	25	30.2
26	22.2	26	64.4
27	17.1	27	31.2
28	0	28	0.2
29	12	29	26
30	37	30	54
31	21	31	32.7
32	22	32	44.5
33	5.7	33	10.5
34	6.1	34	22
35	3	35	16.5
36	0	36	17.5
37	0	37	16.7
38	0	38	3.26
39	0	39	11.3
40	32	40	45
41	5.9	41	35
42	7.12	42	31
Overall Average Accuracy	**10.17023**	**Overall Average Accuracy**	**27.06023**

Appendix B

For the proposed defense, RaFis, all the adversarial images generated in A are defended in 1000 number of queries.