Fall 2023

A System for the Detection of Adversarial Attacks in Computer Vision via Performance Metrics

Sarah Reynolds
reynos23@my.erau.edu

A System for the Detection of Adversarial Attacks in Computer Vision via Performance Metrics

Sarah Reynolds

Department of Electrical Engineering and Computer Science

Embry-Riddle Aeronautical University

Daytona Beach, FL

reynos23@my.erau.edu

Master's Thesis

Fall 2023

Table of Contents

Table of Figures

1. Abstract

Adversarial attacks, or attacks committed by an adversary to hijack a system, are prevalent in the deep learning tasks of computer vision and are one of the greatest threats to these models' safe and accurate use. These attacks force the trained model to misclassify an image, using pixel-level changes undetectable to the human eye. Various defenses against these attacks exist and are detailed in this work. The work of previous researchers has established that when adversarial attacks occur, different node patterns in a Deep Neural Network (DNN) are activated within the model. Additionally, it is known that CPU and GPU metrics look different when different computations are occurring. This work builds upon that knowledge to hypothesize that the system performance metrics, in the form of CPUs, GPUs, and throughput, will reflect the presence of adversarial input in a DNN. This experiment found that external measurements of system performance metrics did not reflect the presence of adversarial input. This work establishes the beginning stages of using system performance metrics to detect and defend against adversarial attacks. Using performance metrics to defend against adversarial attacks can increase the model's safety, improving the robustness and trustworthiness of DNNs.

Keywords: adversarial defenses, DNNs, CNNs, neural networks, adversarial attacks, system performance metrics

2. Introduction

The field of computer vision has developed rapidly in recent years, with computer vision models able to decipher images with an incredibly high level of accuracy [1]. The current success would not have been possible without the architectural development of Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs). Despite the incredible performance of these computer vision models, which have been researched and adapted across domains, DNNs remain susceptible to adversarial attacks [2].

Adversarial attacks perturb images, or make minimal changes to an image's composition, resulting in a DNN making incorrect predictions even though these perturbations are practically undetectable to humans [2]. There are a variety of defenses against adversarial attacks that exist, which are described in Section 3. However, these defenses are not comprehensive, and there is no guarantee that a model is safe against adversarial attacks, as new attack methods are frequently developed. Until these deep learning models show appropriate robustness against adversarial attacks, they must be used cautiously and not with safety critical systems [3].

In a world where the power of computer vision holds so much potential, these Deep Learning (DL) models are rising in popularity. The general public can use them without regulation and with minimal policy guidance, which causes the safety risk of adversarial attacks to be significant. This problem, as it is related to the structure of DNNs and not just computer vision, threatens a wide range of state-of-the-art Artificial Intelligence (AI) practices. In order to counteract the threat of adversarial attacks, this work proposes a novel strategy for detecting the presence of an adversarial attack, with the hopes that it can be used to prevent adversarial attacks in the future. This work fits into a wide array of mitigations against adversarial attacks, from

building robust models to detecting the occurrence of attacks to recovering from these attacks with various methods.

The purpose of this experiment is to test the theory that adversarial attacks cause a significant effect on the system performance metrics during inference. Specifically, four adversarial attack techniques will be studied on two different benchmark data sets to determine the significance of this difference and the potential to use it as an adversarial defense mechanism. This work attempts to answer the following questions:

> RQ1. To what extent does the presence of an adversarial attack reflect in the system performance metrics?
>
> RQ2. How effective is monitoring the system performance metrics in stopping an adversarial attack in real-time?

The approach presented in this work analyzes the system performance metrics in order to detect adversarial attacks. Two benchmark image datasets, MNIST [4] and CIFAR-10 [5], are used to train and test a CNN model. These datasets are used to train a CNN made with Tensorflow, and then the testing dataset is tested using a variety of adversarial attack methods (Fast Gradient Sign Method [6], Projected Gradient Descent [7], Carlini-Wagner [8], and DeepFool [9]). These adversarial attacks were tested and compared against non-adversarial images to determine if an adversarial attack affected the system performance metrics during inference.

This study provides a preliminary study into a potential defense for adversarial attacks. This experiment was structured within a specific environment with a limited number of models and datasets. Therefore, the findings of this study may not hold for alternate model structures.

Despite these limitations, this study addresses the potential of discovering a new method for detecting adversarial input into Deep Neural Networks (DNNs). Exploring novel defense solutions like the one presented in this work is a step towards making deep learning, and therefore artificial intelligence, safer and more trustworthy for the general public and safety-critical systems.

3. Background

This work addresses the impact of adversarial attacks on DNNs. Though existing defenses can protect against some adversarial attacks, the current state of DNNs is that they are not robust against adversarial attacks. Specific properties of the DNNs and prior knowledge about system performance metrics form the theory that suggests that system performance metrics can be used to detect the presence of an adversarial attack.

3.1. Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs)

Computer vision is a series of tasks built with the concepts of deep learning and neural networks. CNNs are a version of DNNs that have been researched and developed since the 1980s [10]. DNNs, as depicted in Figure 3.1, have multiple hidden layers between the input and the output layers [11]. In computer vision, the input represents the pixels that make up a visual image; the output is the conclusions drawn based on that image. The neurons process that input based on various weights and activations determined in the training process.

Unlike traditional DNNs, CNNs make use of limited connections between neurons in what are called convolutional layers [12]. Having fewer connections limits the number of parameters for the model to calculate, allowing the CNNs to process and make predictions on larger images more effectively. The difference between the convolutional layers and the fully connected layers can be seen in Figure 3.2. CNNs and the variations of CNNs are the standard for computer vision tasks due to their high level of performance [12].

DNNs, CNNs, and their derivatives form the basis for most state-of-the-art computer vision tasks. Within that, a wide variety of specific architectures have been developed and tested against

each other to improve the performance and accuracy of the models [13]. Most of these algorithms have incredibly high levels of accuracy.

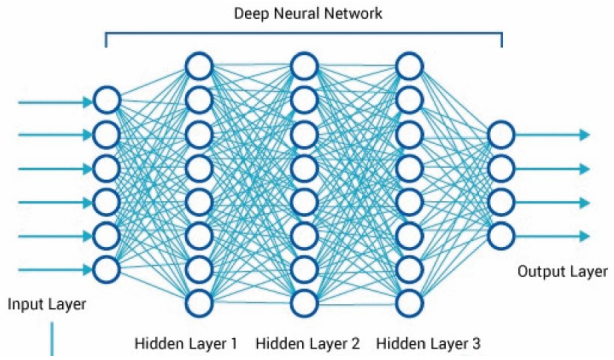


Figure 3.1. shows the connections and representation of a Deep Neural Network (DNN) [14]

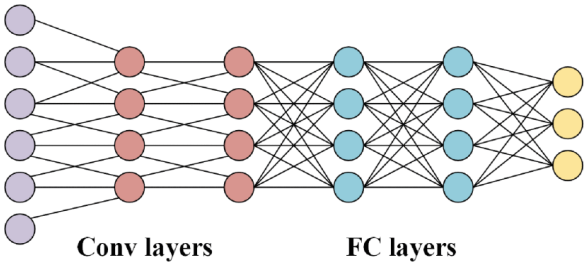


Figure 3.2. shows how convolutional layers compare to fully connected (FC) networks used in traditional deep learning [10]

The complexity that allows these models to interpret complex data accurately comes with the drawback of being difficult to interpret black boxes [15]. This lack of explainability of these models makes their outputs challenging to trust, a feeling compounded by their vulnerability to adversarial attacks [16].

Some traits of these neural networks have been discovered. The conception of node-pruning showed that some nodes within the DNN are either redundant or have little influence over

accurate prediction [17]. These less valuable nodes, however, do tend to be affected more by adversarial inputs [18] [19].

3.2. Adversarial Attacks

Adversarial attacks are not a new threat to AI but rather an ongoing danger. Other researchers have determined [20] that the first description of an adversary purposefully changing data to affect the results of a system was in a 2004 paper on the result of using adversarial manipulations to bypass a textual spam filter [21].

Adversarial attacks occur in computer vision when an input image is distorted, perturbed, or changed so that it is misclassified by the model [22]. These changes are usually indistinguishable to the human eye, yet are one of the significant risks for implementing computer vision tasks in safety-critical environments [23]. These attacks have been a concern for applying deep neural networks over the last decade [16]. Adversarial attacks pose an even greater danger because an attack that works effectively on one model can be effective on a different model, even if the attack was not created for that model [16] [24] [25]. This concept is known as transferability.

There are many variations of adversarial attacks [23]. Some techniques allow for general misclassification, while others target a specific misclassification to occur and cause change to a particular category. Some techniques require access to the model, thus specifically generating adversarial images for a particular model. Other attacks are more general and can be applied to images regardless of the method used for detection and classification.

Improving adversarial attacks has been a popular research topic, as researchers attempt to find the most minimal perturbation that can fool a model, which is also the most dangerous and effective attack [9]. Perturbation size can be used as a measure of robustness, as a model

vulnerable to more minor perturbations is considered less robust than a model that only misclassifies when presented with more significant perturbations.

In addition to the adversarial attack strategies described above (which address the alteration of pixels and other digital changes to the images), there is also a concern that there are natural perturbations or physical objects that change the results from computer vision models [26]. In some cases, these issues naturally occur in the world and exist without malicious intent. In other cases, objects in the physical world are manipulated to no longer be classified by existing models. For example, small black and white stickers can be added to stop signs, and while a human can still read these, computer vision models fail to perform [27]. These contribute to various adversarial attacks that can affect computer vision models.

Most frustratingly to researchers, it is unclear why adversarial examples exist [28]. Part of the complexity in determining the cause of adversarial examples is due to the black-box nature of DNNs. The lack of understanding surrounding adversarial examples is closely tied to the lack of explainability surrounding DNNs. Nevertheless, it signals that DNNs cannot mimic the human decision and learning process as closely as researchers would like.

3.2.1. Adversarial Attack Types

The focus of this work is to detect evasion attacks. In evasion attacks, the model is already trained, and the attack occurs at run time [28]. However, in the field of adversarial attacks, there are more opportunities to attack the model. For example, poisoning attacks, where adversarial data is mixed with the training data to create exploitation paths for future attacks, occur during the model's training [28].

Another distinction between attack types is the goal of the misclassification. If the adversary wants to change the classification to a specific target, the attack is a targeted attack [28]. Alternatively, the attack is untargeted if the only goal is wrong input without a specific class in mind.

If the attacker knows the model's structure and uses the knowledge of that structure to fine-tune the attack, that is considered a white-box attack [28]. If the attacker lacks the structural knowledge of the model and can only access the input and output of the model, the attack is a black-box attack.

The goal of the defense mechanism defined in this work is to defend existing DNNs, unlike other defenses that involve changing the structure of the model from the beginning. Therefore, this work is concerned with detecting evasion attacks. Within the evasion attacks, untargeted and white-box adversarial methods represent typical attacks. Details regarding the attacks replicated in this study can be seen in Section 3.2.2.

3.2.2. Known/Published Adversarial Attacks

This section describes a series of modern and effective adversarial attacks. These adversarial attacks are all replicated in this work to test and evaluate the proposed defense method.

3.2.2.1. Fast Gradient Sign Method (FGSM)

The Fast Gradient Sign Method (FGSM) was introduced in 2014 [6]. This adversarial attack is an example of an untargeted white-box attack [29]. The perturbations are considered small, yet further minimal perturbations have been determined, as shown in Figure 3.4. In this method, a gradient of the loss is used to determine how much each pixel in the input contributes to the loss,

which is used to determine what pixels to change in an input image to generate an adversarial attack.

3.2.2.2. Projected Gradient Descent (PGD)

Projected Gradient Descent [7] is another example of an untargeted, white-box attack [29] published in 2017. This attack is iterative, unlike the one-step method seen in FGSM. The iterations make this method slower, as it calculates a gradient of the loss and uses the gradient to create a perturbation that is projected on an image. However, the iterative nature aims to fine-tune the perturbation to be more effective.

3.2.2.3. Carlini-Wagner (CW)

The Carlini-Wagner (CW) is an adversarial attack published in 2017 [8]. This attack, like the two previous, is a white box attack, but it can be used as a targeted approach [29]. This approach treats finding the slightest perturbation as an optimization problem. Three algorithms (L_0, L_2, and L_∞) were developed and compared in the original publication. The most robust attack is the L_2 attack, the version implemented in this paper.

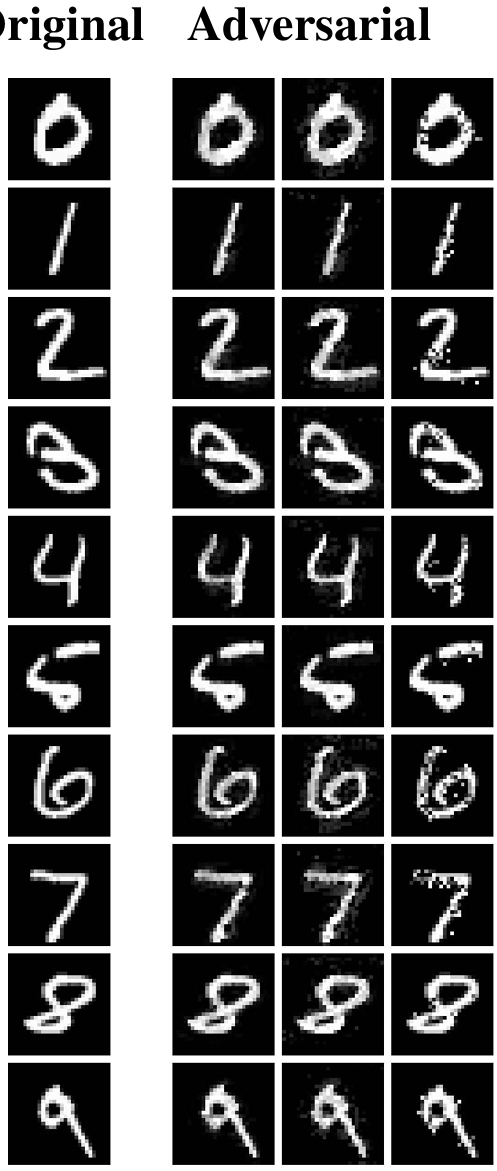


Figure 3.3. The Carlini-Wagner adversarial attack is shown on the MNIST dataset. The three attacks, L_0, L_2, and L_∞, are shown under the "Adversarial" heading from left to right. This image is adapted from [8].

3.2.2.4. DeepFool (DF)

DeepFool [9] is a method for adversarial perturbations published in 2016. This attack is an advanced example of an untargeted white box attack [29]. The algorithm of DeepFool addresses

that previous adversarial perturbations were not finding the minimal change needed for an attack. The difference in perturbation found by DeepFool and FGSM is shown in Figure 3.4. Using a smaller perturbation makes the adversarial attack more subtle and, therefore, more challenging to detect.

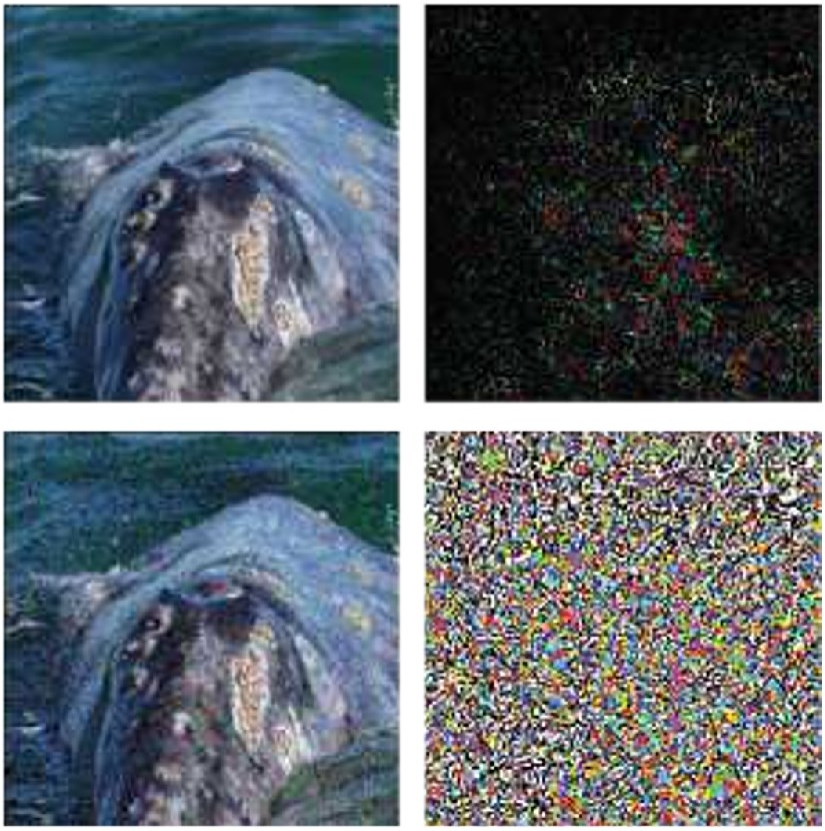


Figure 3.4. The difference between noise needed for DeepFool's perturbation (top) compared to an FGSM perturbation (bottom). Adapted from [9].

3.2.2.5. Other Attacks

A wide variety of attacks exist in the literature but are not replicated in this work. This work selected some of the most popularly discussed and modern adversarial attack techniques. An example of an attack not replicated in this study is the HopSkipJump (HSJ) attack [30], which is

an example of a black box attack that can be either targeted or untargeted [29]. This attack only has access to the model's output but uses changes in the classification to estimate the gradient descent and then optimize the perturbation. This attack is valuable in showing how black box attacks can work, but as discussed in Section **Error! Reference source not found.**, relying on hiding a model to prevent white box attacks from occurring is not a viable strategy.

Further work on this subject should consider other adversarial attack strategies such as HSJ, universal perturbations [31], and physical perturbations [32]. In addition, attacks and defenses should be further addressed on other DNNs unrelated to computer vision.

3.2.3. Considerations on the Threat of Adversarial Attacks

The security of these models is a significant issue and is closely tied to cybersecurity. In one instance, attackers could determine the white box structure of a CNN model by exploiting a GPU side-channel [33]. Between this capability and the fact that adversarial examples can be transferred effectively between models [16] [24] [25], it is clear that assuming that the model structure is secret is not an effective method of defending against adversarial attacks. That assumption would violate the cybersecurity principle that security through obscurity is inadequate.

There are also concerns about publicly generated data sets. The theory is that an exploit can be introduced to the publicly available data set, and all models trained with this dataset can be exploited later through a backdoor attack [34]. These concerns are related to poisoning attacks, not evasion attacks, and are therefore outside the scope of consideration for this work.

3.2.4. The Effects of Adversarial Attacks

Knowing that adversarial attacks exist, it only takes creativity and imagination to determine the possible effects and dangers of using these attacks. In the medical domain, experts have considered the following possibilities [20]:

- In terms of textual data, if the insurance agency uses a DL model to make decisions, medical codes can be manipulated to result in a fraudulently low payment for the patient or increased payment for the doctors.
- In the computer vision domain, it has been suggested that future systems that potentially use patient images to validate doctor diagnoses can be manipulated to validate the results that the doctor decides on.

In both of the above cases, since the input to the DL model is provided by a human, it is easy to insert malicious content in that no "hacking" of the system has to occur. However, a new attack surface for accessing these models is developed, with Graphical Processing Units (GPUs) being offered as a cloud service for consumers to run their DNNs on [35].

3.3. Adversarial Defenses

Researchers have developed and tested a wide range of defenses against adversarial attacks. These defenses have been tested and are effective against some adversarial attacks. As experts suggest, following the best defense practices is essential in using DNNs. For example, like much of the cybersecurity domain, it is more difficult to fully defend a system than to find one attack that works [20].

3.3.1. Detecting Adversarial Images

The first category of adversarial defenses focuses solely on detecting the presence of adversarial attacks.

Some models use classifiers trained separately to detect adversarial attacks [36] [37] [38] [39]. Attempts to look at the scale of the feature attribution scores are effective, if inefficient, to scale to more complex models [39]. Autoencoders can be used to detect anomalies in images that are caused by adversarial attacks [40] [41]. Open-set recognition can be adapted to allow adversarial examples to be considered a part of the open set [42].

Recently, a technique was developed to detect adversarial attacks using random perturbations [43]. It was discovered that adding random noise within a small magnitude would not affect non-adversarial images but could cause adversarial images to revert to the original classification. With that, the change of classification after the noise was added could also be used to alert to an adversarial attack.

A modern defense technique, DeepFense, uses a varying number of defenders to watch various layers of the DNN. These defenders look for uncommon node activations characteristic of an adversarial attack [19]. The number of defenders used in a model can be adjusted based on resource and performance time constraints. In establishing that adversarial attacks activate different node patterns than non-adversarial input, this work contributed significantly to the theory surrounding this research process.

Researchers are evaluating ensemble approaches that are adaptable to include new detection methods as they are developed [44]. Once detection occurs, the model must decide how to handle it. Some models reject adversarial attacks rather than classifying them [41]. Alternatively,

denoising can be used on the image so that it can still be processed and classified [42]. Other methods of handling adversarial attacks once they are detected are described in the following sections.

3.3.2. "Fixing" Adversarial Images

The second category of defenses looks to return adversarial inputs to the original, non-adversarial input, at least to the extent that the model does not misclassify the images.

Some defenses look at ways to remove the adversarial attack from the image. One method involves removing noise from images [45]. Further work uses GANs to remove the perturbations [46] [47], a solution that is most effective on trivial perturbations [46]. In other work, CNNs detect adversarial examples and remove the noise before it is provided as input to the classification model- allowing the defense to work universally with any classification model [48]. After the detection technique, Magnet uses a detector to determine if the input is adversarial. If adversarial input is found, an autoencoder reduces noise and ideally removes the perturbation [37].

Research has determined that image transformations can be used against adversarial attacks, showing that the adversarial attacks (FGSM, Deepfool, and Carlini & Wagner) were no longer effective when the input underwent image transformations before being fed to the model [49]. The most successful image transformations were total variation minimization and image quilting, as the adversary cannot use them during their attack due to the random property of both transformations.

3.3.3. Alternate Model Structures

Another category of defenses involves making architectural changes to a DNN to be more robust against attacks [50]. Structural changes to the model, such as bounding the ReLU activations in the architecture, are used so that the effects of small perturbations cannot propagate and grow throughout the network layers [51]. Alternatively, the model structure can be changed to learn more robust features as a method of adversarial defense [52].

One strategy of defense against adversarial attacks is defensive distillation [53]. Derived from a technique to reduce the computational complexity of DNNs [54], defensive distillation uses two models to predict instead of one. The first model is trained as expected, but the last layer (a softmax layer) is then fed into a second DNN, from which the classification predictions are derived. Since this technique reduces overfitting and allows models to generalize to new inputs, the models are also less sensitive to adversarial attacks.

3.3.3.1. Adversarial Training

Adversarial training is a method of increasing the robustness of the model by using adversarial inputs during model training. It is a method that changes the structure of the model because it changes the training data used.

The concept began when it was realized that geometric transformations can affect model performance, and it was discovered that CNNs should be trained on data augmented by these transformations [55]. This discovery has expanded to recommendations for adversarial perturbations on a subset of the training data [9]. It is found that training with Gaussian noise can increase the robustness of a model and requires less work than training model-specific adversarial examples [51].

Physical perturbations or perturbations implemented in the natural world rather than digitally have not been analyzed to a great extent in terms of adversarial training [56]. This exposure is essential because models are only robust against these physical perturbations when similar examples have been found during training.

Adversarial training has been found to have drawbacks. It has been found that adversarial training can decrease the model's performance on non-adversarial images, an effect that is amplified if adversarial perturbations are too excessive [9]. Excessive adversarial training can reduce the overall robustness of the model [9]. Special care should be taken when determining the adversarial attacks and perturbations to use during training.

3.3.4. Effectiveness of Defenses

Not all defenses work as cleanly as the original researchers would hope. Rebuttals [57] [8] show that claims of robustness [51] [37] [46] did not hold when alternative perturbations and adversarial strategies were used. Much like when dealing with software bugs, it is impossible to guarantee that a model is fully robust against adversarial attacks, as novel adversarial attacks are being developed regularly [7].

Even on the MNIST dataset, actual robustness is an unsolved problem. That highlights that we may be even further off than we think from making other, more complex images resistant to adversarial attacks [58]. The lack of robustness means deep learning is inappropriate for safety-critical operations [59]. New detection methods must be determined until robustness can be guaranteed.

3.4. Model Performance Metrics

Other work has established categories for what performance metrics matter when dealing with deep learning models. From work on training these deep learning models, the following categories for performance metrics have been established: CPU utilization, GPU utilization, peak memory consumption, energy consumption, and throughput [60]

The TensorFlow Profiler [61] is an assortment of tools that characterize hardware usage when doing deep learning. The abilities of this tool allow the developer to view profiles of the computing operations and an overview of the model training process [62]. The TensorFlow Profiler and similar tools are most commonly used to identify bottlenecks during the model training process [63]. Inference performance is when the model is given data to categorize and occurs in real-world situations when the model is used as a part of a system.

The Profiler is one of the best tools available, as it integrates with existing frameworks and models using TensorFlow to get performance data. However, other techniques are possible to measure the duration and occurrence of (1) reading model parameters into the CPU, (2) copying model parameters into the GPU, and (3) executing the neural network in the GPU kernels [64].

The CPU, GPU, and throughput performance metrics are identified as the variables to monitor during the proposed defense technique. CPUs and GPUs are involved in the DNN process [65], and the usage of these metrics reflects the complexity of the computation they perform [66]. The timing of these operations has been exploited in other attacks and has revealed details about the operations occurring [67] [68]. Throughput, or the speed at which the DNN can process input, is related to the performance and speed of the CPU and GPU.

4. Related Work

Among the wide variation of defenses for adversarial attacks, few directly address the concept of node pathways and system performance. Despite this, the work that has been done supports the hypothesis proposed in this work.

4.1. Detecting Adversarial Attacks

As discussed in Section 3.3, detecting adversarial attacks is essential in ensuring security. Without detection, there is no chance of recovering from the attack. Work has been done to train a separate neural network to classify whether the data is accurate or contains adversarial perturbations [69] [70]. Other work relies on other objects in the scene to verify if a detection is legitimate or potentially the consequence of an adversarial attack [71]. People are researching new model architectures that are more robust when presented with adversarial examples [72]. K-nearest neighbor models are used post-classification to see if objects fit with other exemplary images in their class [73]. The preprocessing of images uses genetic algorithms to filter out adversarial images [74]. Image transformation successfully detects the presence of adversarial examples [75].

It has been established that adversarial examples tend to activate a different, less commonly used pattern of nodes during inference [19]. This was found through the work in developing DeepFense, a detection algorithm that utilizes patterns of node activation to detect adversarial attacks at runtime [19]. It utilizes a variable number of defense models that look at various layers within the model, allowing the detectors to detect suspicious input based on the activation space without touching the model itself, thus making it much harder for attackers to find perturbations that will not trigger the defense modules.

The DeepFense algorithm is the closest to the defense method proposed in this work. However, this proposed method will be a black-box method. Assuming that when non-commonly used nodes are activated, the model may have a higher memory consumption or a slower throughput, the proposed defense mechanism uses the performance at inference time to make the same conclusions about adversarial attacks taking novel pathways as DeepFense.

4.2. Performance Metrics for DL

The only example of work found that looked at hardware state to detect adversarial methods was in an analysis of adversarial attacks on autonomous driving models [76]. This work proposes a more detailed look at resource utilization to detect and classify adversarial attacks of all techniques. Other work has established categories for what performance metrics matter when dealing with deep learning models. From work on training these deep learning models, the following categories for performance metrics have been established: CPU utilization, GPU utilization, peak memory consumption, energy consumption, and throughput [60]

4.3. Alignment

This study is meant to contribute to existing research and protocols for the defense against evasion adversarial attacks via discussion.

The importance of performance metrics in deep learning [76] [60], along with supporting work that the pattern of node activation differs between adversarial and non-adversarial examples [18] [19] and supporting research that shows that both the CPU and GPU have different utilization under different operations all support the theory that the system performance metrics will reflect the presence of adversarial input due to the pathways activated by the adversarial input in the CNNs.

5. Experiment Design

The method detailed in this work is depicted in Figure 5.1. Image data from the MNIST and CIFAR-10 datasets were used to train separate TensorFlow CNNs. The image data subsets set aside for testing underwent transformations according to the FGSM, PGD, CW, and DF adversarial attacks. Repeated tests were done, and system performance metrics were collected. The system performance metrics were analyzed to determine if there is a statistically significant and practical difference in the system metrics between adversarial and non-adversarial input.

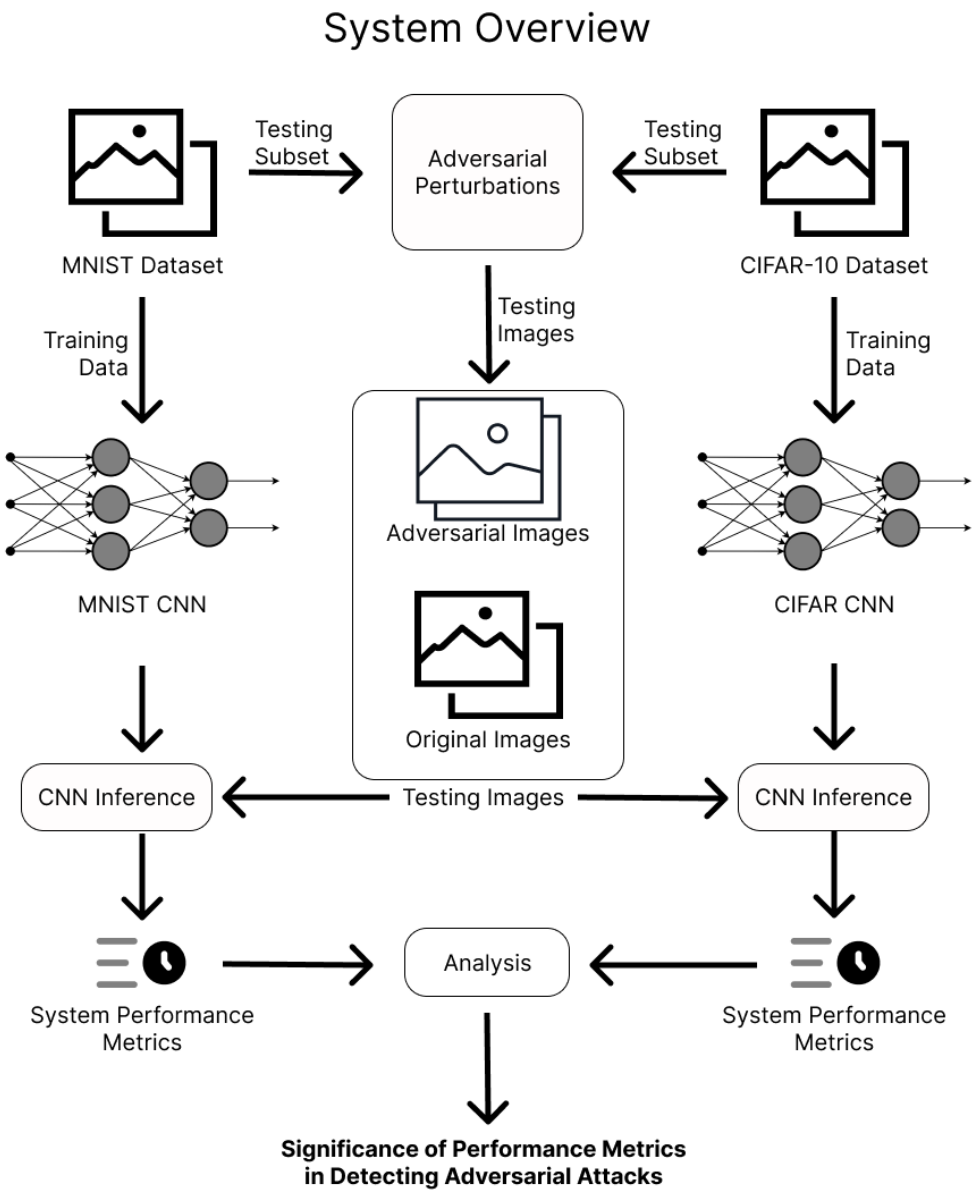


Figure 5.1 The proposed overarching method for detecting adversarial images

5.1. Environment

All experimentation was conducted via the Google Colab platform [77]. Google Colab was selected as the computational environment for several reasons. The first reason was the prospect of using multiple runtime environments, including CPUs, GPUs, and TPUs. Though this experiment uses a GPU environment, further work on this subject will include various

environments. Additionally, this environment supported the development and testing of TensorFlow models, which were chosen as a standard model configuration to run this experiment. Using Google Colab allows future experiments to be run in the same environment, and results are not bound to the researcher's machine.

5.2. Image Datasets

Two different image classification datasets were used to test the performance metrics of models on various perturbations. These datasets are the MNIST dataset and the CIFAR-10 dataset. The MNIST dataset contains grayscale images of handwritten digits. It is a simple classification dataset and is, therefore, helpful in demonstrating fundamental principles related to computer vision [4]. The CIFAR-10 dataset, consisting of 32x32 color images, was an example of a slightly more complex dataset [5]. Though the images are still small, they contain realistic scenarios and provide a slightly higher level of complexity. These datasets are recognized as benchmark datasets for the research community, allowing this work to be compared against others in the domain.

5.3. Model Selection

For each data set chosen, a CNN model was trained until the accuracy on the dataset was 90% accurate and any increase in accuracy on the validation dataset had plateaued. The CNN model used multiple convolutional, max-pooling, and fully connected layers. The model architecture was the same for each dataset, and the only difference was the training data used and the amount of training provided until each model reached the appropriate accuracy.

5.4. Attack Methods

This study used a diverse set of adversarial attack methods to test the proposed defensive technique. The details of these attacks are described in Section 3.2.2.

The FGSM is a more straightforward method that does not necessarily find the minimum perturbation but is a benchmark attack and is used to show the model's vulnerability. The PGD attack is an iterative attack that assesses the white-box structure of a model to a deeper extent than the FGSM attack. The CW attack was incorporated into our evaluation due to its optimization-driven approach, which generates more subtle and tailored adversarial inputs. Lastly, the DeepFool attack was due to its ability to find a minimal and subtle perturbation using iterative techniques. These techniques mark a set of the most commonly used attacks. Each attack is easily implemented, showing how the threat of adversarial attacks is easily accessible to the would-be attacker.

The general structure of the adversarial attack remained the same on each dataset. Some parameters were altered so that each attack showed a notable drop in accuracy when possible. The accuracy of each attack on the CNN model was measured. The model's accuracy when facing adversarial attacks is a valuable metric for showing the robustness of the model. Therefore, if an attack decreases the model's accuracy, it is a threat that needs to be detected.

5.5. Adversarial Detection

The novel detection method proposed in this paper focuses on using the model's performance metrics at inference time. A description of the various performance metrics available is discussed in Section 3.4.

Resource utilization was analyzed for the Central Processing Unit (CPU) and the Graphics Processing Unit (GPU). For each test, the average and peak memory usage for both the CPU and GPU were observed. These metrics were gathered using the psutil tool [78].

Throughput, or the capacity of the model to infer a given number of samples within a standardized unit of time, was used to indicate the speed and efficiency of the model's performance. This metric was measured in terms of the number of samples or batches per second.

A performance metric commonly used yet not implemented in this study is energy consumption due to logistical constraints. As discussed in Section 5.1, the experiments were conducted using the cloud resources provided by Google Colab. This environment prevented the collection of energy consumption data, but the benefits of using the Google Colab platform outweighed the limitations. Nevertheless, future research should consider using energy consumption as a potentially valuable performance metric when calculating this performance metric.

5.5.1. Analysis Methods

The performance metrics were first analyzed using the ANOVA test to determine statistical significance among the various adversarial inputs. The Tukey's Honestly Significand Difference test was then used to determine which groups had a significant difference and to what extent.

The ANOVA (Analysis of Variance) statistical test is a method used to analyze differences among group means in a sample. It was chosen because it can compare the means of three or more groups, making it an extension of the t-test typically used for comparing two means. This was relevant for the five categories of adversarial input that were tested. If ANOVA indicates that at least one group is different from the others, it does not specify which groups are different.

The post-hoc test, Tukey's Honestly Significant Difference (HSD), was used to pinpoint the specific groups that differ from each other. Tukey's HSD is designed to control for the type I error rate (false positive occurrence of significance) and provide pairwise comparisons between group means to determine which specific groups have significantly different means. This test determined what adversarial perturbations had a difference in system performance metrics compared to non-adversarial input.

5.5.2. Experiment 1: Comparing Adversarial and Non-Adversarial Images

In the first experiment, the batches of adversarial and non-adversarial methods were handled separately. The separation allowed us to analyze the performance metrics of inference without considering how and when adversarial examples may be introduced during real-time operation. The inference performance metrics and accuracy were collected and analyzed to determine the significance of the results.

5.5.3. Experiment 2: Mixing

In the second experiment, a more realistic scenario was performed. In this experiment, the adversarial attacks were given to the model amongst original, non-adversarial images. This approach aimed to mimic a real-life scenario where the adversarial attack is only likely to occur on some of the images. The inference performance metrics and accuracy were collected and analyzed to determine the significance of the results.

6. Results

Though each test had slightly differing results, this section details the contributing factors to the conclusion that system performance metrics do not reflect the presence of adversarial input.

6.1. Adversarial Attacks

The difference between the adversarial attacks and the original image can be seen in Figure 6.1. With both the model trained on MNIST data and the model trained on CIFAR data, they were each trained until they had at least 90% training accuracy. The training resulted in 99% testing accuracy for the MNIST model and 63% accuracy for the CIFAR data. The four adversarial attacks were applied to the testing images to degrade the accuracy of those images. The accuracy was degraded for all attacks except the Carlini Wagner attack on the MNIST dataset.

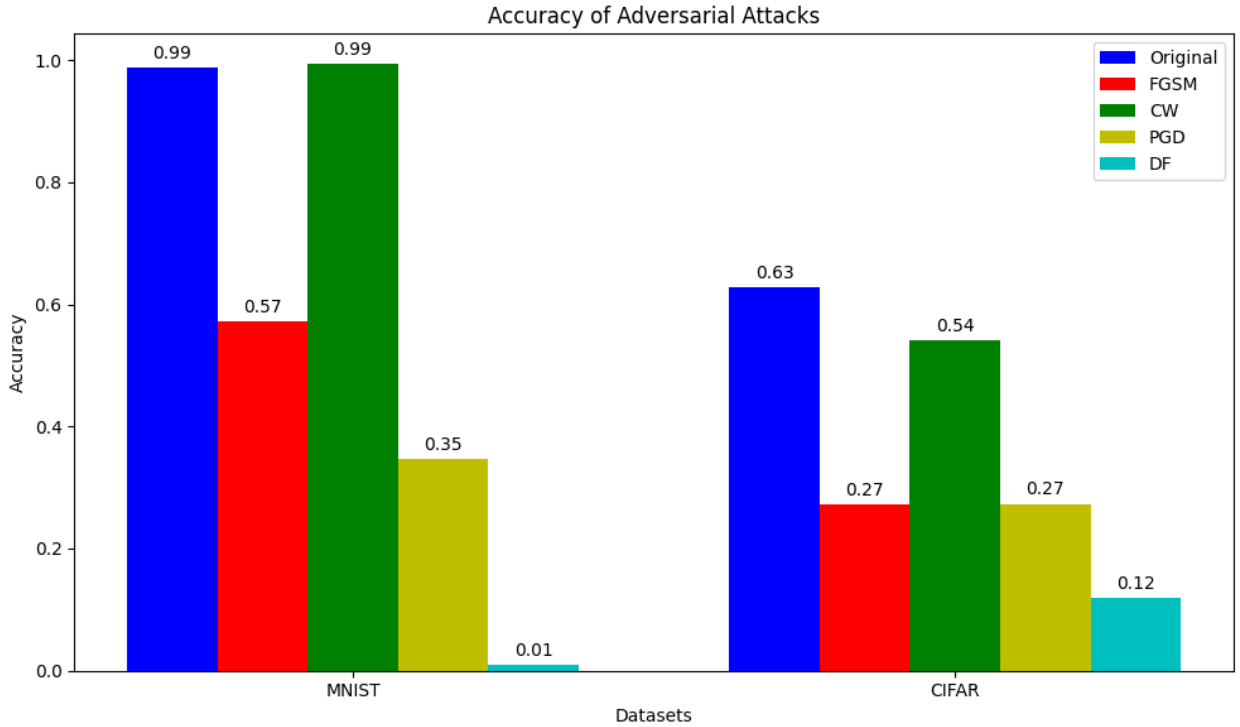


Figure 6.1 shows the accuracy of the trained models on original data compared to the four described adversarial attacks.

The high accuracy of the Carlini Wagner attacks may be due to the models' simplicity. The simplicity, therefore, prevented the method from finding an optimum perturbation. Despite this, this range of accuracies gives a variety of situations to test the hypothesis.

6.2. Experiment 1 Results

In experiment 1, adversarial and non-adversarial inputs were tested separately to get a baseline. In this experiment, for both the MNIST and CIFAR datasets, the images were inferred in batches by the model and in a singular, one-by-one inference setting.

The results shown in Figure 6.2 show the difference in inference performance metric between the adversarial and non-adversarial images during the experiment where MNIST images were provided in batches to the model. The difference between the throughput on original images when compared with FGSM and PGD attacks is statistically significant. The GPU usage was consistent across all inputs. In this test, the original images' throughput was slightly lower than the FGSM and the PGD attacks. The difference in the mean CPU memory between the original images and the four adversarial attacks is slight but statistically significant.

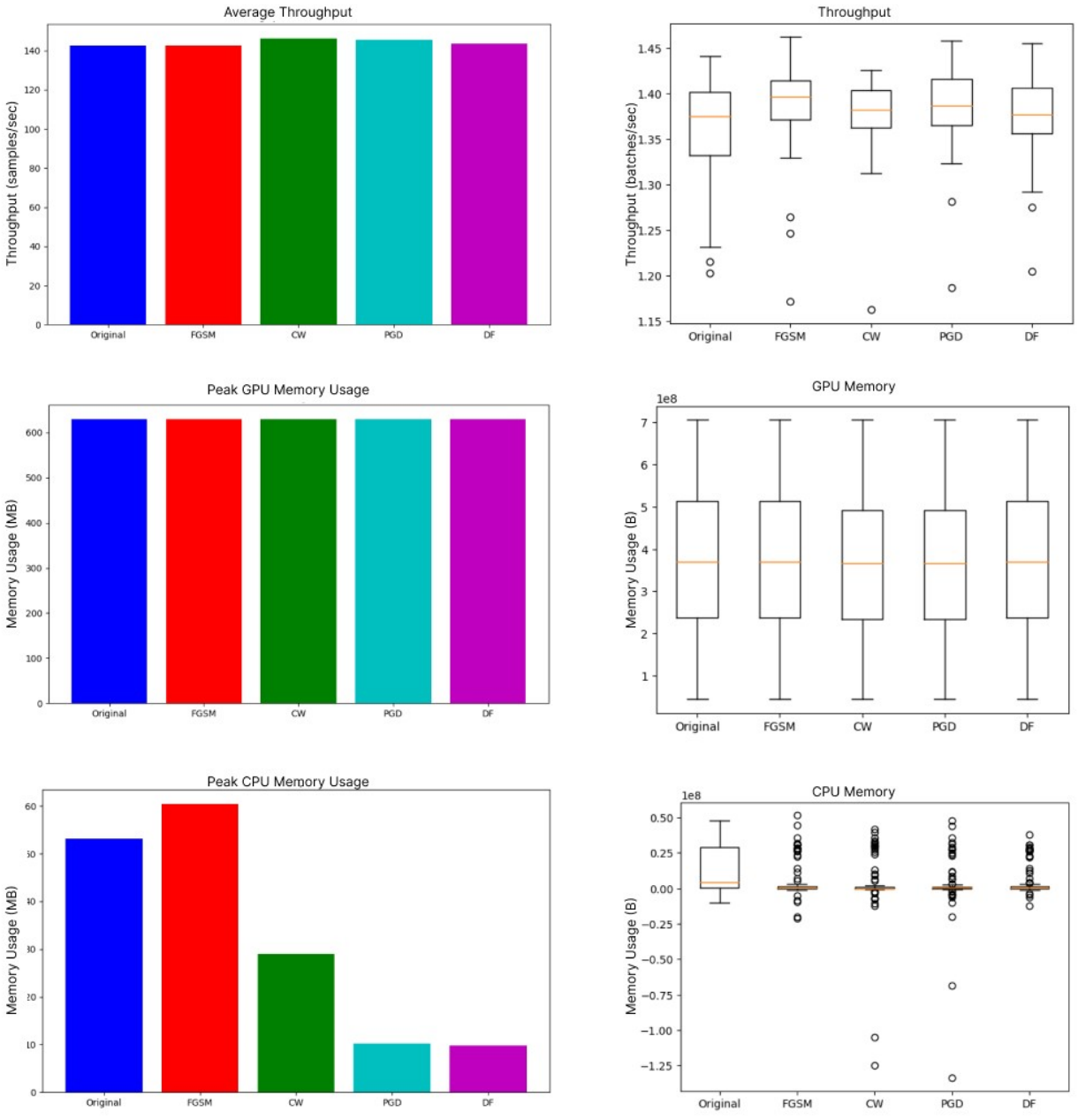
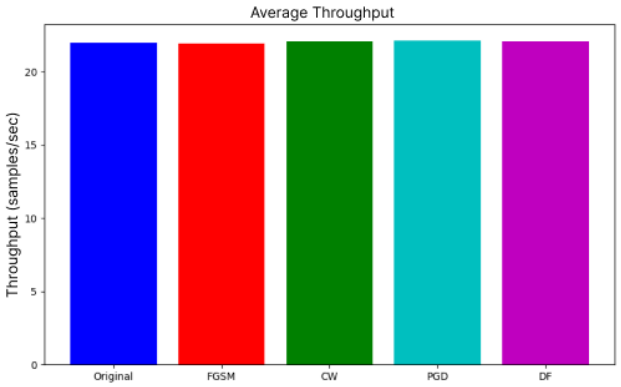
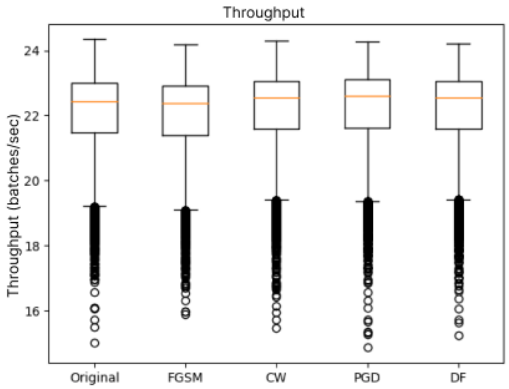


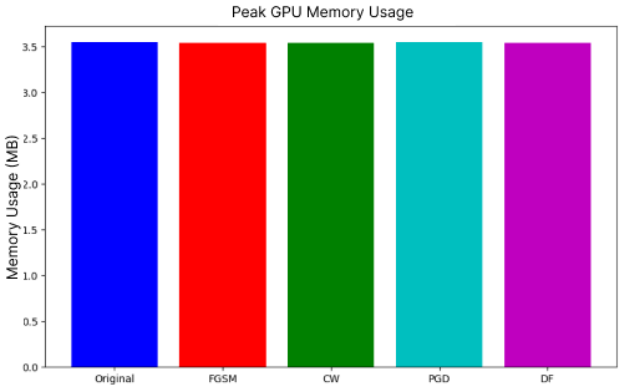
Figure 6.2 shows the performance metrics of adversarial attacks compared to the non-adversarial attacks when MNIST images were sent to the model in batches.

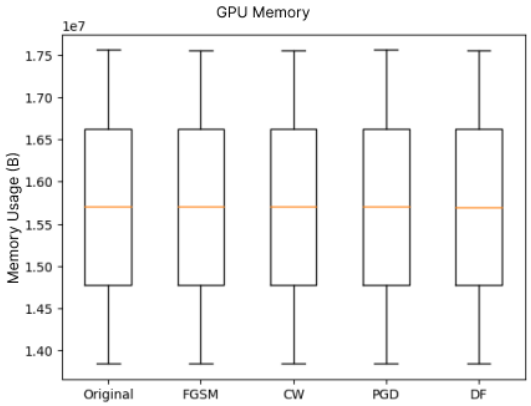
The results shown in Figure 6.3 show the difference in inference performance metric between the adversarial and non-adversarial images during the experiment where MNIST images were provided individually to the model for inference. The difference between the throughput on the

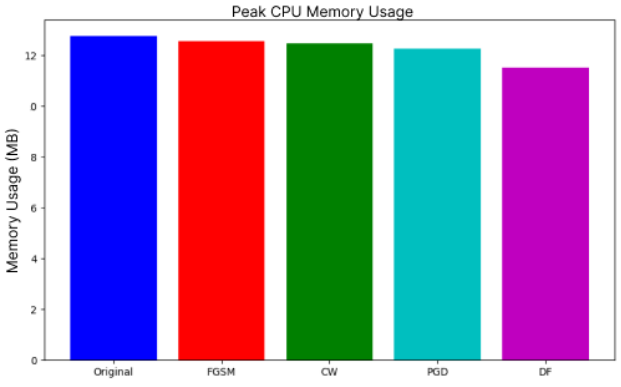
original images and those exposed to the FGSM attack significantly differed from all other throughputs. However, there was not enough variation to be statistically significant for either the GPU or the CPU usage. The original images had a slightly lower throughput than all adversarial attacks except for FGSM.











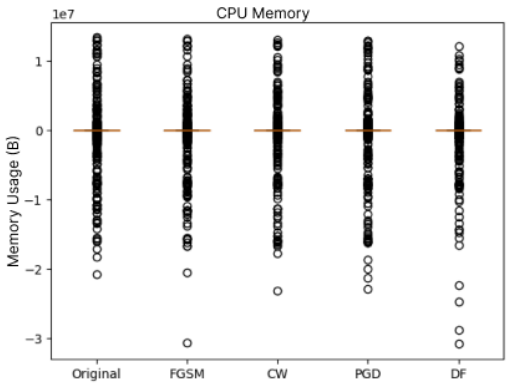


Figure 6.3 shows the performance metrics of adversarial attacks compared to the non-adversarial attacks when MNIST images were sent to the model individually.

The results shown in Figure 6.4 show the difference in inference performance metric between the adversarial and non-adversarial images during the experiment where CIFAR-10 images were provided in batches to the model. The throughput for the original images compared to all adversarial images was statistically significant, though the GPU and CPU memory usage showed no differences. The mean throughput for the original images was slightly lower than all the adversarial images.

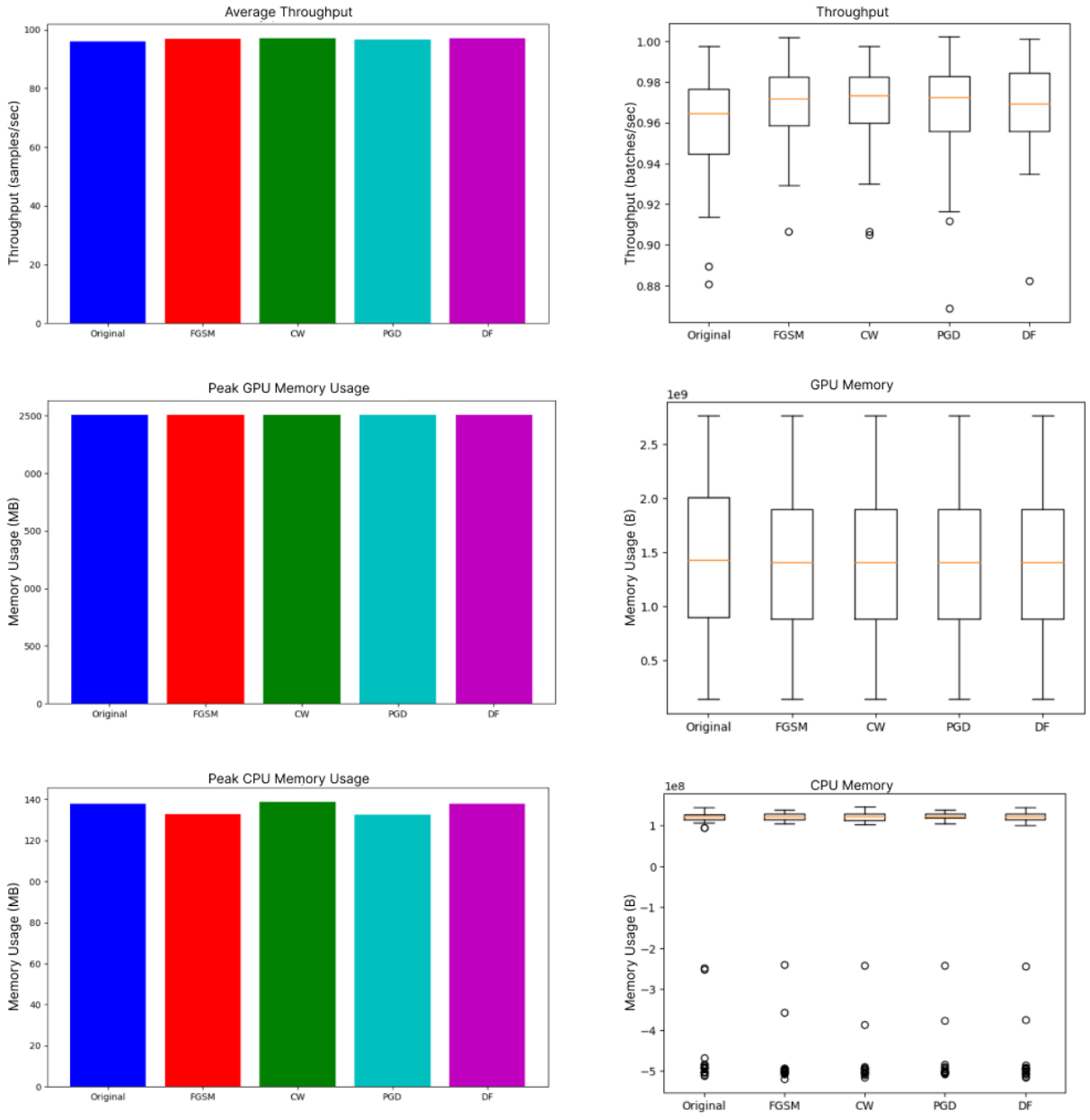
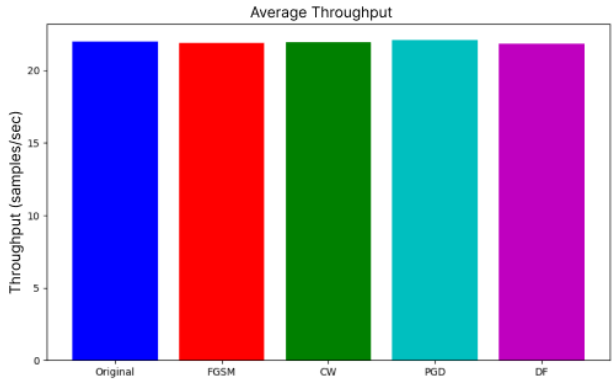
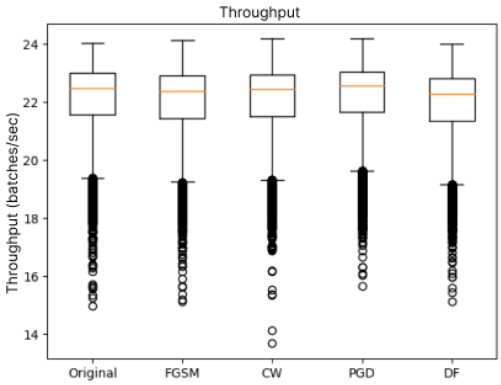


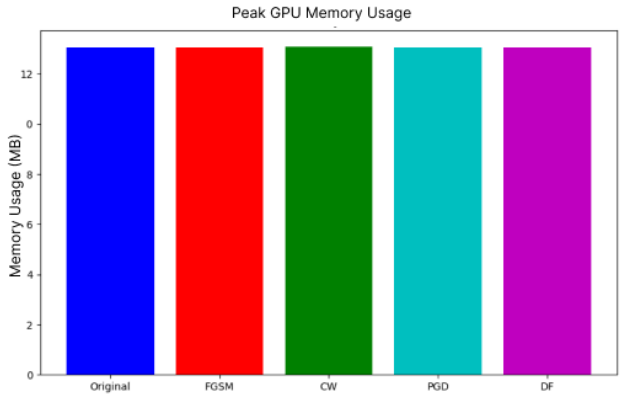
Figure 6.4 shows the performance metrics of adversarial attacks compared to the non-adversarial attacks when CIFAR images were sent to the model in batches.

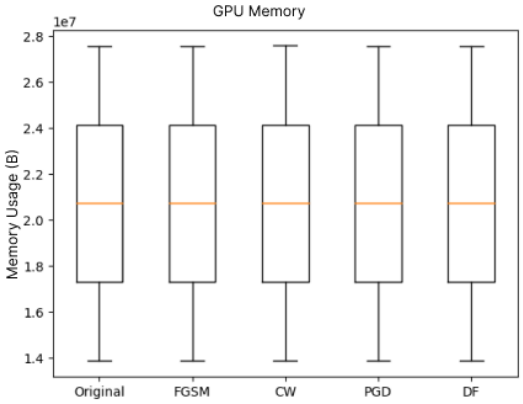
The results shown in Figure 6.5 show the difference in inference performance metric between the adversarial and non-adversarial images during the experiment where CIFAR-10 images were provided individually to the model. There was no difference in the CPU or GPU memory usage.

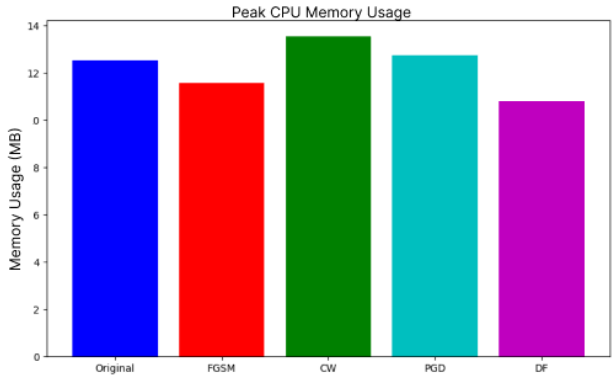
For the throughput, all averages were statistically significant in their difference except for the original images compared to the images that the CW attack had transformed. However, the mean of the original images throughput was slightly higher than the FGSM and DF throughputs and slightly lower than the PGD throughputs.











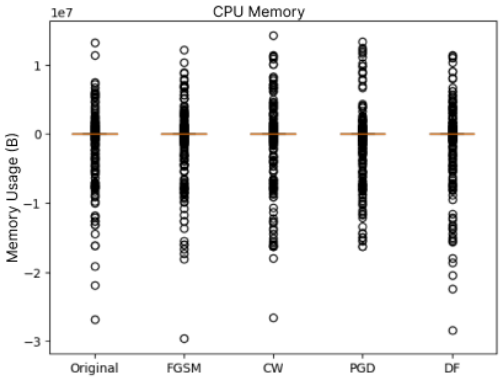


Figure 6.5 shows the performance metrics of adversarial attacks compared to the non-adversarial attacks when CIFAR images were sent to the model individually.

6.3. Experiment 2 Results

Experiment 2 replicates Experiment 1, except for the way samples were taken. Experiment 1 measured all values in standalone sessions for each adversarial category. In Experiment 2, the model was fed the original images with interspersed batches or instances of adversarial examples.

The results shown in Figure 6.6 show the difference in inference performance metric between the adversarial and non-adversarial images during the experiment where MNIST images were provided in batches to the model, with adversarial examples sprinkled amongst the original image batches. None of the system metrics measured showed differences with statistical significance.

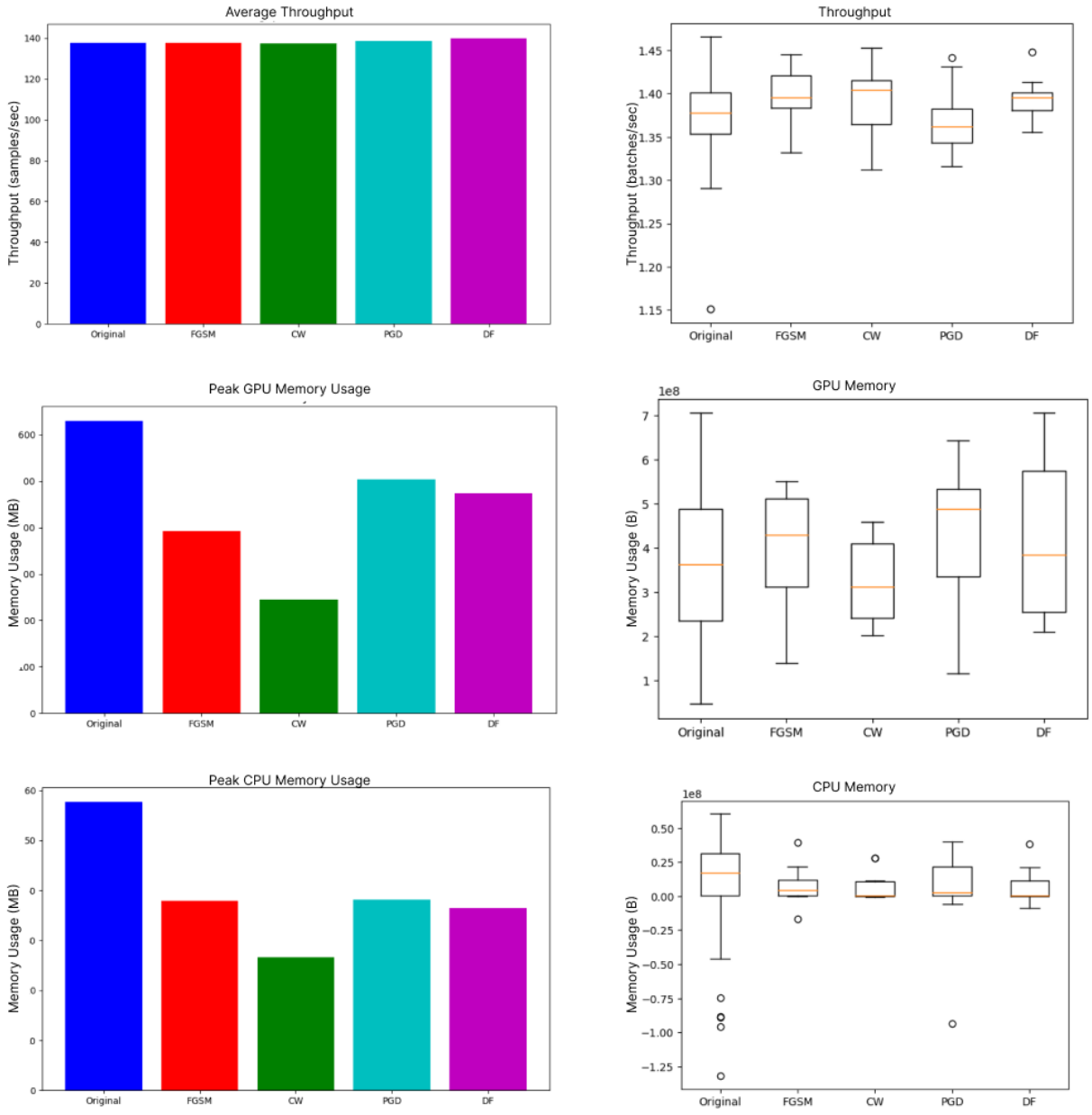
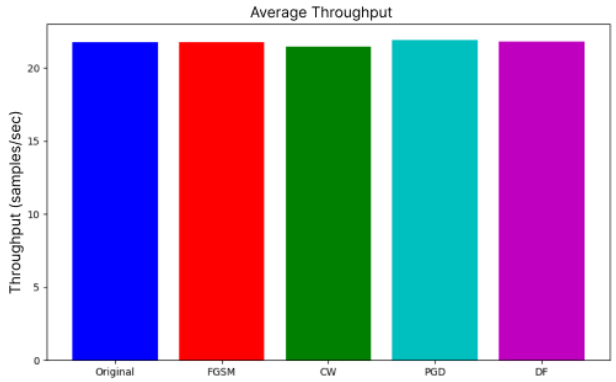
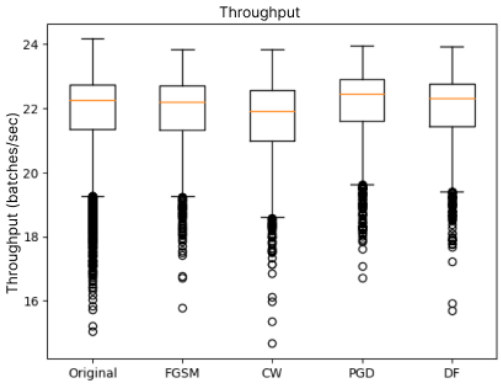


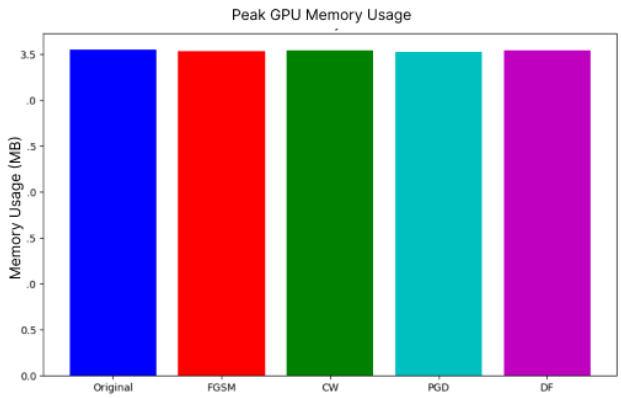
Figure 6.6 shows the performance metrics of adversarial attacks compared to the non-adversarial attacks when MNIST images were sent to the model in batches.

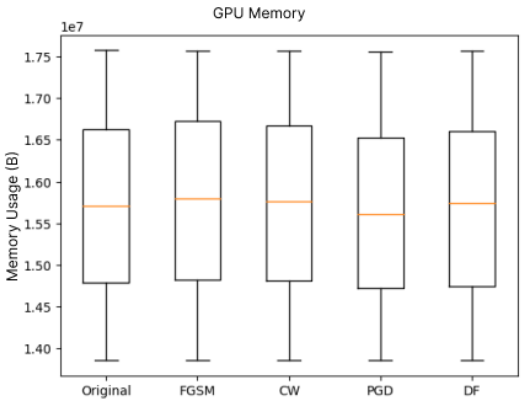
The results shown in Figure 6.7 show the difference in inference performance metric between the adversarial and non-adversarial images during the experiment where MNIST images were provided in batches to the model, with adversarial examples sprinkled amongst the original

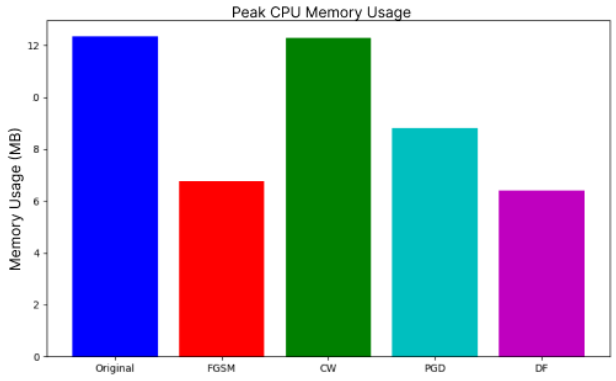
image batches. In this scenario, the throughput of original images was statistically significant compared to the CW and PGD throughputs, with the original throughput slightly higher than the CW input and slightly lower than the PDG input. Neither the GPU nor CPU memory usage showed any statistical significance.











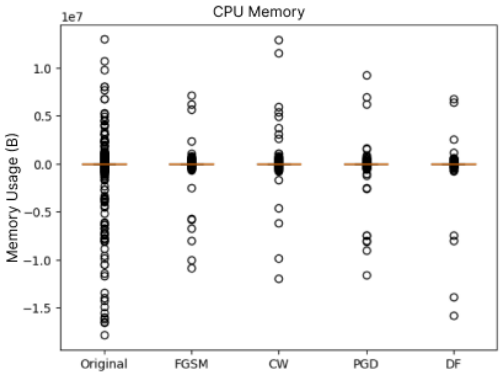


Figure 6.7 shows the performance metrics of adversarial attacks compared to the non-adversarial attacks when MNIST images were sent to the model individually.

The results shown in Figure 6.8 show the difference in inference performance metric between the adversarial and non-adversarial images during the experiment where CIFAR images were provided in batches to the model, with adversarial examples sprinkled amongst the original image batches. In this scenario, the throughput of original images was lower than the throughput of the PGD images in a statistically significant way. Neither the CPU nor the GPU memory usage was statistically significant.

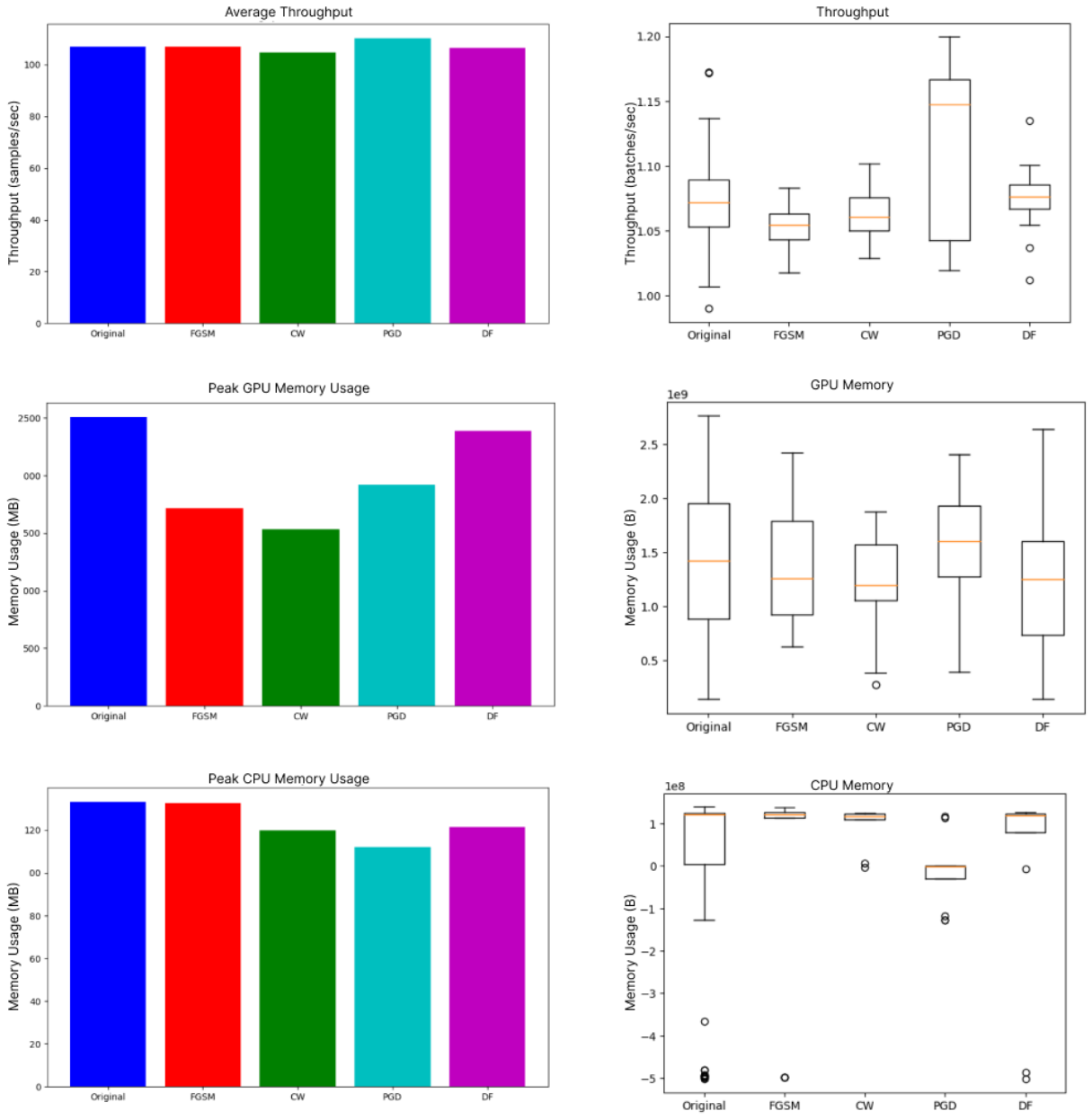


Figure 6.8 shows the performance metrics of adversarial attacks compared to the non-adversarial attacks when CIFAR images were sent to the model in batches.

The results shown in Figure 6.9 show the difference in inference performance metric between the adversarial and non-adversarial images during the experiment where CIFAR images were provided in batches to the model, with adversarial examples sprinkled amongst the original

image batches. The throughput of the original images was higher than the throughput of the CW images in a statistically significant way. None of the other metrics were statistically significant.

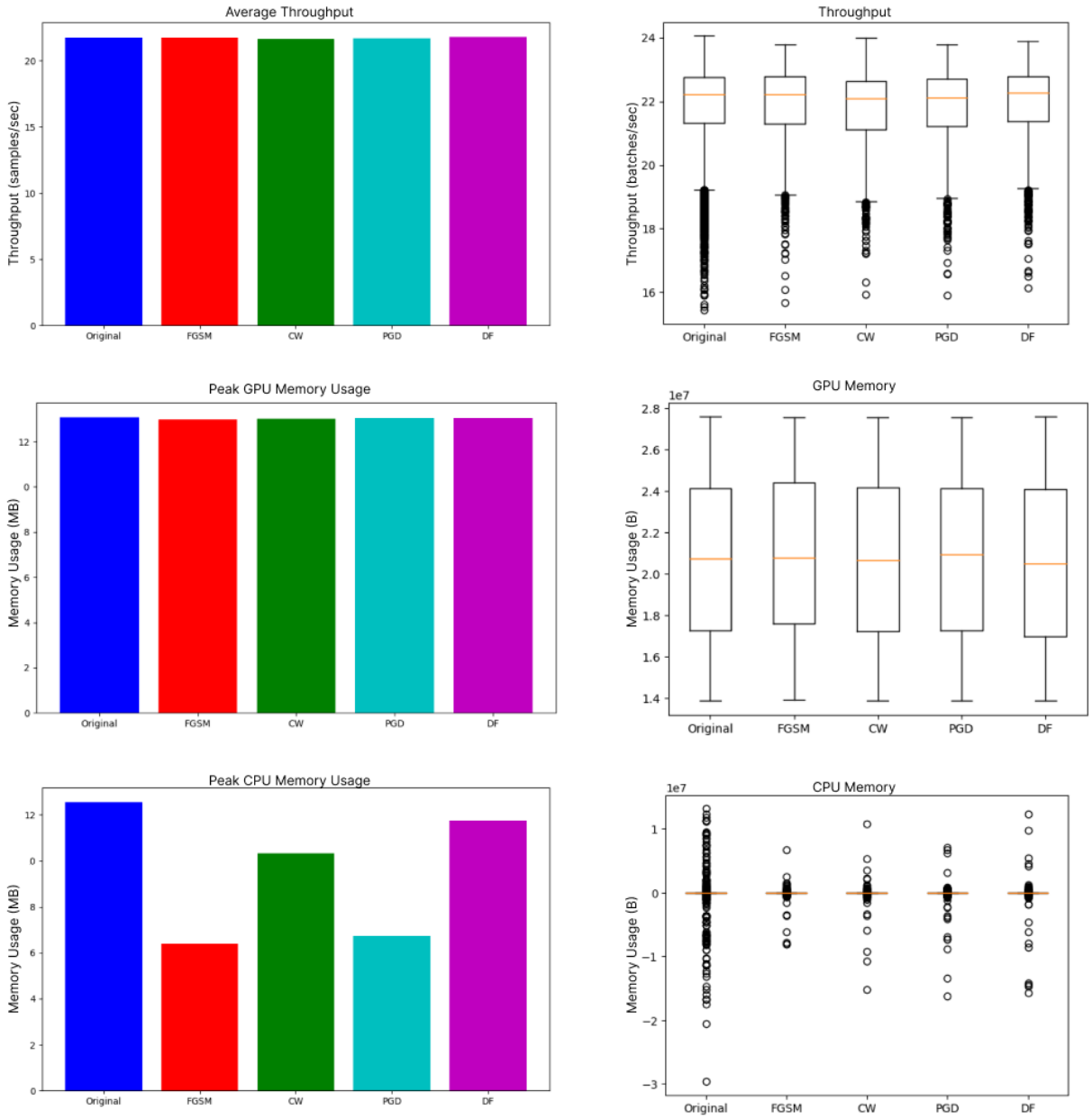


Figure 6.9 shows the performance metrics of adversarial attacks compared to the non-adversarial attacks when CIFAR images were sent to the model individually.

7. Discussion

When comparing adversarial to non-adversarial input, the results showed no statistical significance to CPU and GPU usage during the model's inference. In most of the tests, the throughput of the original images differed from at least one of the adversarial attacks statistically significantly. However, the difference in throughput is not practically significant for two reasons. First, the throughput of the original images was not consistently higher or lower than the throughput of adversarial attacks. Secondly, the difference in average was so slight that a clear threshold cannot be applied to differentiate between adversarial and non-adversarial input.

Therefore, the hypothesis was disproven that external measures of system performance metrics during model inference can detect adversarial attacks.

7.1. Future Work

Despite the initial hypothesis not being supported — that system performance metrics could effectively function as an external-to-the-model measure for detecting adversarial attacks — the foundational theory of this research still shows significant promise. Future explorations in this area will first focus on a more nuanced measurement of system performance, particularly emphasizing enhanced control over key hardware components such as CPUs, GPUs, and memory caches. This refined approach is expected to enable a more sensitive detection of anomalies indicative of adversarial attacks.

The key to this approach is the granular monitoring of hardware components. Enhanced control over CPUs, for instance, involves tracking metrics like clock speed and cache misses, offering insights into how these components respond under different computational loads, including those induced by adversarial attacks. Similarly, for GPUs, monitoring core utilization and memory

bandwidth usage can provide indications of atypical patterns during attacks. Close observation of memory cache behavior is also vital, as variations in cache hits and misses could signify attack manipulations.

Another potentially promising direction for this research is the exploration of conditional neural networks. These networks are tailored to activate specific pathways based on predetermined conditions, optimizing computational efficiency and focus. This ensures that the system only processes pathways relevant to adversarial attack detection, reducing unnecessary computational loads and boosting overall system performance.

Incorporating conditional neural networks allows for a dynamic, responsive system that adapts to varying scenarios and focuses computational resources effectively. This improves detection efficiency and reduces the likelihood of missing subtle adversarial manipulations. These networks can be fine-tuned to identify specific adversarial attack patterns, enhancing the system's ability to differentiate between legitimate and malicious inputs.

This research acknowledges the ongoing challenge of developing effective defenses for DNNs against adversarial attacks, a task that requires continuous innovation and adaptation. When a successful detection method is developed, it should not be isolated but integrated with other defensive strategies, as discussed in Section 3.3. This comprehensive approach is crucial, recognizing the complexity and evolving nature of adversarial threats in AI and machine learning. By merging various defensive techniques and refining them continually, the field progresses towards more secure and resilient DNNs, capable of withstanding the sophisticated adversarial attacks expected in the future.

8. Conclusion

This work investigated the possibility of using system performance metrics, specifically CPU usage, GPU usage, and threshold, to determine whether or not a CNN was processing adversarial input. The hypothesis that system performance metrics would reflect an adversarial attack was tested using a TensorFlow CNN, the MNIST and CIFAR-10 datasets, and four different adversarial attacks (FGSM, PGD, CW, and DF). These tests were performed in the Google Colab environment.

The experiments revealed no significant indicators in the system performance metrics that adversarial input was present. Despite this setback, future work will continue to evaluate possible ways to use the system performance metrics to detect and defend against adversarial attacks.

References

[1] T. Diwan, G. Anirudh and J. V. Tembhurne, "Object detection using YOLO: Challenges, architectural successors, datasets and applications," *multimedia Tools and Applications,* vol. 82, no. 6, pp. 9243-9275, 2023.

[2] N. Akhtar and A. Mian, "Threat of Adversarial Attacks on Deep Learning in Computer Vision: A Survey," *IEEE Access,* vol. 6, pp. 14410-14430, 2018.

[3] European Union Aviation Safety Agency, "Artificial Intelligence Roadmap 2.0: Human-centric approach to AI in aviation," 2023.

[4] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE,* vol. 86, no. 11, pp. 2278--2324, 1998.

[5] A. Krizhevsky, G. Hinton and others, "Learning multiple layers of features from tiny images," 2009.

[6] I. J. Goodfellow, J. Shlens and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572,* 2014.

[7] A. Mądry, A. Makelov, L. Schmidt, D. Tsipras and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *stat,* vol. 1050, p. 9, 2017.

[8] N. Carlini and D. Wagner, "Towards Evaluating the Robustness of Neural Networks," in *2017 IEEE Symposium on Security and Privacy (SP)*, San Jose, CA, USA, 2017.

[9] S.-M. Moosavi-Dezfooli, A. Fawzi and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[10] Z. Li, F. Liu, W. Yang, S. Peng and J. Zhou, "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 33, no. 12, pp. 6999-7019, 2022.

[11] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition, O'Reilly Media, Inc., 2019.

[12] X. Feng, Y. Jiang, X. Yang, M. Du and X. Li, "Computer vision algorithms and hardware implementations: A survey," *Integration,* vol. 69, no. C, pp. 309-320, 2019.

[13] Z.-Q. Zhao, P. Zheng, S.-T. Xu and X. Wu, "Object Detection With Deep Learning: A Review," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 30, no. 11, pp. 3212-3232, 2019.

[14] W. Abbass, Z. Bakraouy, A. Baina and M. Bellafkih, "Classifying IoT security risks using Deep Learning algorithms," in *2018 6th International Conference on Wireless Networks and Mobile Communications (WINCOM)*, Marrakesh, Morocco, 2018.

[15] G. Joshi, R. Walambe and K. Kotecha, "A Review on Explainability in Multimodal Deep Neural Nets," *IEEE Access,* vol. 9, pp. 59800-59821, 2021.

[16] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199,* 2013.

[17] T. He, Y. Fan, Y. Qian, T. Tan and K. Yu, "Reshaping deep neural network for fast decoding by node-pruning," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, 2014.

[18] L. Ma, F. Juefei-Xu, F. Zhang, J. Sun, M. Xue, B. Li, C. Chen, T. Su, L. Li and Y. Liu, "Deepgauge: Multi-granularity testing criteria for deep learning systems," *Proceedings of the 33rd ACM/IEEE international conference on automated software engineering,* pp. 120-131, 2018.

[19] B. D. Rouhani, M. Samragh, M. Javaheripi, T. Javidi and F. Koushanfar, "Deepfense: Online accelerated defense against adversarial deep learning," in *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2018.

[20] S. G. Finlayson, J. D. Bowers, J. Ito, J. L. Zittrain, A. L. Beam and I. S. Kohane, "Adversarial attacks on medical machine learning," *Science,* vol. 363, no. 6433, pp. 1287-1289, 2019.

[21] N. Dalvi, P. Domingos, Mausam, S. Sanghai and D. Verma, "Adversarial Classification," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Seattle, WA, USA, 2004.

[22] X. Huang, D. Kroening, W. Ruan, J. Sharp, Y. Sun, E. Thamo, M. Wu and X. Yi, "A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability," *Computer Science Review,* vol. 37, p. 100270, 2020.

[23] X. Yuan, P. He, Q. Zhu and X. Li, "Adversarial examples: Attacks and defenses for deep learning," *IEEE transactions on neural networks and learning systems,* vol. 30, no. 9, pp. 2805--2824, 2019.

[24] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh and P. McDaniel, "The space of transferable adversarial examples," *arXiv preprint arXiv:1704.03453,* 2017.

[25] X. Wei, S. Liang, N. Chen and X. Cao, "Transferable adversarial attacks for image and video object detection," *arXiv preprint arXiv:1811.12641,* 2018.

[26] V. Shankar, A. Dave, R. Roelofs, D. Ramanan, B. Recht and L. Schmidt, "A systematic framework for natural perturbations from videos," in *ICML 2019 Workshop on Identifying and Understanding Deep Learning Phenomena*, 2019.

[27] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno and D. Song, "Robust physical-world attacks on deep learning visual classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018.

[28] S. Zhou, C. Liu, D. Ye, T. Zhu, W. Zhou and P. S. Yu, "Adversarial attacks and defenses in deep learning: From a perspective of cybersecurity," *ACM Computing Surveys,* vol. 55, no. 8, pp. 1-39, 2022.

[29] S. Y. Khamaiseh, D. Bagagem, A. Al-Alaj, M. Mancino and H. W. Alomari, "Adversarial deep learning: A survey on adversarial attacks and defense mechanisms on image classification," *IEEE Access,* 2022.

[30] J. Chen, M. I. Jordan and M. J. Wainwright, "Hopskipjumpattack: A query-efficient decision-based attack," in *2020 ieee symposium on security and privacy (sp)*, 2020.

[31] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi and P. Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[32] D. Song, K. Eykholt, I. Evtimov, E. a. L. B. Fernandes, A. Rahmati, F. Tramer, A. Prakash and T. Kohno, "Physical adversarial examples for object detectors," in *12th USENIX workshop on offensive technologies (WOOT 18)*, 2018.

[33] J. Wei, Y. Zhang, Z. Zhou, Z. Li and M. A. Al Faruque, "Leaky dnn: Stealing deep-learning model secret with gpu context-switching side-channel," in *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2020.

[34] M. Goldblum, D. Tsipras, C. Xie, X. Chen, A. Schwarzschild, D. Song, A. Mądry, B. Li and T. Goldstein, "Dataset Security for Machine Learning: Data Poisoning, Backdoor Attacks, and Defenses," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 45, no. 2, pp. 1563-1580, 2023.

[35] S. Iserte, F. J. Clemente-Castelló, A. Castelló, R. Mayo and E. S. Quintana-Ortí, "Enabling GPU Virtualization in Cloud Environments.," in *CLOSER (2)*, 249-256.

[36] J. H. Metzen, T. Genewein, V. Fischer and B. Bischoff, "On Detecting Adversarial Perturbations," in *International Conference on Learning Representations*, 2016.

[37] D. Meng and H. Chen, "Magnet: a two-pronged defense against adversarial examples," in *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, 2017.

[38] G. Goswami, N. Ratha, A. Agarwal and R. a. V. M. Singh, "Unravelling robustness of deep learning based face recognition against adversarial attacks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[39] P. Yang, J. Chen, C.-J. Hsieh, J.-L. Wang and M. Jordan, "Ml-loo: Detecting adversarial examples with feature attribution," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[40] C. Cintas, S. Speakman, V. Akinwande, W. Ogallo, K. Weldemariam, S. Sridharan and E. McFowland, "Detecting adversarial attacks via subset scanning of autoencoder activations and reconstruction error," in *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, 2021.

[41] P. Ghosh, A. Losalka and M. J. Black, "Resisting adversarial attacks using gaussian mixture variational autoencoders," in *Proceedings of the AAAI conference on artificial intelligence*, 2019.

[42] R. Shao, P. Perera, P. C. Yuen and V. M. Patel, "Open-set adversarial defense with clean-adversarial mutual learning," *International Journal of Computer Vision,* vol. 130, no. 4, pp. 1070-1087, 2022.

[43] B. Huang, Y. Wang and W. Wang, "Model-Agnostic Adversarial Detection by Random Perturbations," in *IJCAI*, 2019.

[44] F. Craighero, F. Angaroni, F. Stella, C. Damiani, M. Antoniotti and A. Graudenzi, "Unity is strength: Improving the detection of adversarial examples with ensemble approaches," *Neurocomputing,* vol. 554, p. 126576, 2023.

[45] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," *arXiv preprint arXiv:1412.5068,* 2014.

[46] G. Jin, S. Shen, D. Zhang, F. Dai and Y. Zhang, "Ape-gan: Adversarial perturbation elimination with gan," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[47] P. Samangouei, M. Kabkab and R. Chellappa, "Defense-gan: Protecting classifiers against adversarial attacks using generative models," *arXiv preprint arXiv:1805.06605,* 2018.

[48] Y. Song, T. Kim, S. Nowozin, S. Ermon and N. Kushman, "Pixeldefend: Leveraging generative models to understand and defend against adversarial examples," *arXiv preprint arXiv:1710.10766,* 2017.

[49] C. Guo, M. Rana, M. Cisse and L. Van Der Maaten, "Countering adversarial images using input transformations," *arXiv preprint arXiv:1711.00117,* 2017.

[50] H. Qian and M. N. Wegman, "L2-nonexpansive neural networks," in *International Conference on Learning Representations*, 2019.

[51] V. Zantedeschi, M.-I. Nicolae and A. Rawat, "Efficient defenses against adversarial attacks," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, 2017.

[52] T. Dai, Y. Feng, B. Chen, J. Lu and S.-T. Xia, "Deep image prior based defense against adversarial examples," *Pattern Recognition,* vol. 122, p. 108249, 2022.

[53] N. Papernot, P. McDaniel, X. Wu, S. Jha and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *2016 IEEE symposium on security and privacy (SP)*, San Jose, CA, USA, 2016.

[54] G. Hinton, O. Vinyals and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531,* 2015.

[55] A. Fawzi and P. Frossard, "Manitest: Are classifiers really invariant?," *arXiv preprint arXiv:1507.06535,* 2015.

[56] O. Ojaswee, A. Agarwal and N. Ratha, "Benchmarking Image Classifiers for Physical Out-of-Distribution Examples Detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[57] N. Carlini and D. Wagner, "Magnet and" efficient defenses against adversarial attacks" are not robust to adversarial examples," *arXiv preprint arXiv:1711.08478,* 2017.

[58] L. Schott, J. Rauber, M. Bethge and W. Brendel, "Towards the first adversarially robust neural network model on MNIST," *arXiv preprint arXiv:1805.09190,* 2018.

[59] C. Meyers, T. Löfstedt and E. Elmroth, "Safety-critical computer vision: an empirical survey of adversarial evasion attacks and defenses on computer vision systems," *Artificial Intelligence Review,* pp. 1-35, 2023.

[60] J. Liu, J. Liu, W. Du and D. Li, "Performance analysis and characterization of training deep learning models on mobile device," in *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*, 2019.

[61] TensorFlow, "A profiling and performance analysis tool for TensorFlow," 2021. [Online]. Available: https://github.com/tensorflow/profiler.

[62] S. W. D. Chien, A. Podobas, I. B. Peng and S. Markidis, "tf-Darshan: Understanding Fine-grained I/O Performance in Machine Learning Workloads," in *2020 IEEE International Conference on Cluster Computing (CLUSTER)*, Kobe, Japan, 2020.

[63] TensorFlow, "Profile Inference Requests with TensorBoard," [Online]. Available: https://www.tensorflow.org/tfx/serving/tensorboard.

[64] M. Ji, S. Yi, C. Koo, S. Ahn, D. Seo, N. Dutt and J.-C. Kim, "Demand Layering for Real-Time DNN Inference with Minimized Memory Usage," in *2022 IEEE Real-Time Systems Symposium (RTSS)*, 2022.

[65] A. A. Awan, H. Subramoni and D. K. Panda, "An in-depth performance characterization of CPU-and GPU-based DNN training on modern architectures," in *Proceedings of the Machine Learning on HPC Environments*, 2017.

[66] V. W. Lee, C. Kim, J. Chhugani, M. Deisher, D. Kim, A. D. Nguyen, N. Satish, M. Smelyanskiy, S. Chennupaty, P. Hammarlund and others, "Debunking the 100X GPU vs. CPU myth: an evaluation of throughput computing on CPU and GPU," in *Proceedings of the 37th annual international symposium on Computer architecture*, 2010.

[67] G. Zaid, L. Bossuet, A. Habrard and A. Venelli, "Methodology for efficient CNN architectures in profiling attacks," in *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020.

[68] Z. Chen, M. Tang, J. Li and others, "Inversion attacks against CNN models based on timing attack," *Security and Communication Networks,* vol. 2022, 2022.

[69] J. H. Metzen, T. Genewein, V. Fischer and B. Bischoff, "On detecting adversarial perturbations," *arXiv preprint arXiv:1702.04267,* 2017.

[70] F. Carrara, F. Falchi, R. Caldelli, G. Amato, R. Fumarola and R. Becarelli, "Detecting Adversarial Example Attacks to Deep Neural Networks," in *Proceedings of the 15th International Workshop on Content-Based Multimedia Indexing*, Florence, Italy, 2017.

[71] M. Yin, S. Li, Z. Cai, C. Song, M. S. Asif, A. K. Roy-Chowdhury and S. V. Krishnamurthy, "Exploiting multi-object relationships for detecting adversarial attacks in complex scenes," in *proceedings of the IEEE/CVF international conference on computer vision*, 2021.

[72] P. Harder, F.-J. Pfreundt, M. Keuper and J. Keuper, "SpectralDefense: Detecting Adversarial Attacks on CNNs in the Fourier Domain," in *2021 International Joint Conference on Neural Networks (IJCNN)*, Shenzhen, China, 2021.

[73] G. Cohen, G. Sapiro and R. Giryes, "Detecting adversarial samples using influence functions and nearest neighbors," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020.

[74] K. D. Gupta, D. Dasgupta and Z. Akhtar, "Determining sequence of image processing technique (ipt) to detect adversarial attacks," *SN Computer Science,* vol. 2, pp. 1-20, 2021.

[75] S. Tian, G. Yang and Y. Cai, "Detecting adversarial examples through image transformation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[76] Y. Deng, X. Zheng, T. Zhang, C. Chen, G. Lou and M. Kim, "An Analysis of Adversarial Attacks and Defenses on Autonomous Driving Models," in *2020 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, Austin, TX, USA, 2020.

[77] Google, "Google Colaboratory," [Online]. Available: https://colab.research.google.com/.

[78] G. Rodola, "psutil," [Online]. Available: https://github.com/giampaolo/psutil.

[79] A. Yeboah-Ofori, U. M. Ismail, T. Swidurski and F. Opoku-Boateng, "Cyber Threat Ontology and Adversarial Machine Learning Attacks: Analysis and Prediction

Perturbance," in *2021 International Conference on Computing, Computational Modelling and Applications (ICCMA)*, Brest, France, 2021.

[80] V. Mavroeidis and S. Bromander, "Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence," in *2017 European Intelligence and Security Informatics Conference (EISIC)*, Athens, Greece, 2017.

[81] X. Hu, D. Cheng, J. Chen, X. Jin and B. Wu, "Multiontology Construction and Application of Threat Model Based on Adversarial Attack and Defense Under ISO/IEC 27032," *IEEE Access,* vol. 10, pp. 117955-117972, 2022.

[82] D. P. Pereira, C. Hirata and S. Nadjm-Tehrani, "A STAMP-based ontology approach to support safety and security analyses," *Journal of Information Security and Applications,* vol. 47, pp. 302--319, 2019.

[83] S. Kotyan and D. V. Vargas, "Adversarial Robustness Assessment: Why both L0 and L8 Attacks Are Necessary," *arXiv preprint arXiv:1906.06026,* 2019.