# Praxis Tech School Kolkata: Courses ...

# **Adversarial Image Attack Defenses: A Study of effects of FGSM, Patch and CW Attack**

A Capstone Project report submitted in partial fulfillment of the requirements for the Post

Graduate Program in Data Science at Praxis Tech School, Kolkata, India

**Koustav Mazumder(C24006)**

**Mili Chowdhury(C24008)**

**Pratishruti Sahoo(C24009)**

**Jyoti Mallick(C24005)**

**Anup Kumar Raj(C24018)**

of Data Science 2024 Spring Batch

Under the supervision of

**Prof. Jaydip Sen**

**Praxis Tech School, Kolkata, India**

**Introduction**

Today, as machine learning perseveres in driving critical applications like autonomous vehicles, facial recognition systems, and medical diagnostics, it becomes necessary to ensure the reliability and security of these technologies becomes increasingly important. A crucial challenge in this field, especially for image classification[31] tasks, is the susceptibility of machine learning models to adversarial attacks. Attacks which intentionally makes the models to misclassify may result in causing general mayhem in real world

Two of the most well-known adversarial attacks are the **Fast Gradient Sign Method (FGSM)** and the **Carlini-Wagner (CW) attack**. FGSM, introduced by Goodfellow and colleagues in 2014, is a straightforward yet effective facility. It alters input data by using the gradients of the model’s loss function, leading to misclassifications with minimal computational effort. Because FGSM can rapidly generate disturbed images, it is widely used to test how resilient a model is. On the other hand, the Carlini-Wagner attack, developed by Carlini and Wagner in 2017, is more sophisticated. It treats the creation of adversarial examples as an optimization problem, producing tiny perturbations that are almost invisible but can bypass even strong defenses like defensive distillation.

Beyond these traditional methods, Patch Attacks and Projected Gradient Descent (PGD) attacks have also gained attention [7]. Patch attacks involve placing a small, physically realizable patch on an image, which causes the model to misclassify it while the change remains barely noticeable to humans. This type of attack is particularly worrying for real-world systems, such as those used in facial recognition [35] or autonomous driving. PGD attacks build on FGSM by applying gradient-based perturbations repeatedly, making them more effective and commonly used in adversarial training.In our project, we plan to assess how robust several modern image classification models are against these four types of attacks: FGSM, CW, Patch, and PGD. We will focus on three leading deep learning architectures—DenseNet161, RESNET34, EfficientNet-B0 which were chosen for their strong performance in image classification tasks, and we will test them against various adversarial attacks to understand their weaknesses and limitations fully.To defend against these attacks, we will use traditional methods like **defensive distillation**. This technique improves resilience against simpler attacks like FGSM by training models on softened probability distributions. However, defensive distillation often fails against more advanced attacks like CW, highlighting the need for more robust defense mechanisms. Therefore, we will also explore other strategies, including **robust optimization, adversarial training,** and **input transformation techniques**. Robust optimization aims to enhance the model’s architecture and training process to better resist adversarial perturbations. Adversarial training involves exposing the model to adversarial examples during training to strengthen its defenses. Additionally, input transformation methods, such as preprocessing inputs to remove adversarial noise, can provide an extra layer of protection by filtering out malicious alterations.

# The main objective of our research is to thoroughly evaluate the robustness of these three state-of-the-art models when faced with multiple adversarial attacks. By comparing their performance with and without the implementation of defensive strategies, we aim to demonstrate how effective these defenses are against different types of adversarial threats. This study will provide valuable insights into the strengths and vulnerabilities of current image classification models and help guide the development of more secure machine learning systems in the future.

# **Related work**

Since deep learning emerged, we've witnessed remarkable progress in areas like autonomous systems, natural language processing, and image recognition. At the heart of these advancements are Deep Neural Networks (DNNs), which excel at identifying complex patterns and achieving high accuracy across various tasks. However, a significant drawback is their vulnerability to adversarial attacks. These attacks can undermine the reliability and security of DNNs, posing serious risks especially in critical applications such as self-driving cars, facial recognition systems, and healthcare diagnostics.

### 

### **Types of Adversarial Attacks**

**FGSM Attack**

The Fast Gradient Sign Method (FGSM), introduced by Goodfellow and colleagues in 2014, is one of the simplest yet most effective ways to create adversarial examples. Imagine you have an image that a model correctly classifies. FGSM slightly alters this image by using the gradients of the loss function relative to the input. This small tweak can mislead the model into making incorrect predictions. Despite its simplicity, FGSM is powerful enough to reveal significant weaknesses in DNNs and is widely used to test model robustness. An enhanced version of this method, known as Projected Gradient Descent (PGD), applies these perturbations multiple times, making the attack even more formidable. In our project, we will utilize both FGSM and PGD to evaluate how resilient different models are against gradient-based attacks.

**Patch Attack**

Patch attacks take a different approach by adding small, noticeable patches to images that trick classifiers into making wrong predictions. Unlike gradient-based attacks, these patches can be physically placed in the real world—for example, attaching a small sticker to a stop sign could cause an autonomous vehicle to misinterpret it. This type of attack is particularly concerning for security-sensitive applications. In our study, we'll examine both **single-objec**t patch attacks and **universal** patch attacks to appraise how robust image classification models are in real-world scenarios.

**CW Attack**

The Carlini-Wagner (CW) attack, introduced by Carlini and Wagner in 2017, represents a more advanced form of adversarial attack. It outperforms FGSM by generating examples with minimal perturbations, making them nearly imperceptible to humans. Unlike FGSM, the CW attack Frames the creation of adversarial examples as an optimization problem, aiming to find the smallest possible change that causes the model to misclassify the input with high confidence. This attack has become a standard benchmark for evaluating the robustness[11] of machine learning models because it can bypass many existing defenses. In our project, we will implement the CW attack to explore the vulnerabilities of our selected models and test the effectiveness of various defense mechanisms.

**PGD Attack**

Projected Gradient Descent (PGD) builds on FGSM by applying perturbations iteratively, making it a more powerful adversary. Instead of making a single small change, PGD takes multiple steps, each time adjusting the input slightly while keeping the overall perturbation within a predefined limit. This method not only serves as an attack but is also used in adversarial training, where models are trained with adversarial examples to improve their robustness. In our research, PGD will be used as a rigorous test to challenge the defenses of the models we are evaluating.

### **Defensive Strategies**

**Defensive Distillation**

One of the earlier defense techniques is defensive distillation, introduced by Papernot and colleagues. This method introduces training models using softened probability distributions, often referred to as soft labels, instead of the standard hard labels. By doing so, defensive distillation smooths the decision boundaries of the model, making it less sensitive to small input perturbations that adversarial attacks exploit. Essentially, the model becomes less likely to be fooled by minor changes in the input data. While defensive distillation has been effective against attacks like the Fast Gradient Sign Method (FGSM), it has shown limitations when faced with more sophisticated attacks such as the CW attack. These advanced attacks can still find ways to navigate the smoothed decision boundaries and cause misclassifications, highlighting the need for additional defense mechanisms (Papernot et al., 2016).

**Adversarial Training**

Another powerful defense strategy is adversarial training. This approach involves incorporating adversarial examples into the training process of the model. By exposing the model to these perturbed inputs during training, it learns to recognize and resist similar manipulations in the future. Adversarial training effectively increases the model's robustness by ensuring that it can maintain accurate predictions even when faced with adversarial inputs. This method has gained popularity because it directly addresses the vulnerabilities by strengthening the model against the very attacks it might encounter. However, adversarial training can be computationally intensive and may be in need of a large and diverse set of adversarial examples to be truly effective (Madry et al., 2018).

**Robust Optimization**

Robust optimization focuses on enhancing the model’s architecture and training process to better withstand adversarial perturbations. Unlike traditional training methods that aim to minimize the average loss, robust optimization seeks to minimize the worst-case loss under potential adversarial attacks. This approach ensures that the model performs reliably even in the presence of malicious inputs designed to cause maximum disruption. By optimizing for robustness, models become inherently more resistant to a wide range of adversarial strategies, providing a stronger foundation for secure machine learning applications (Zhang et al., 2020).

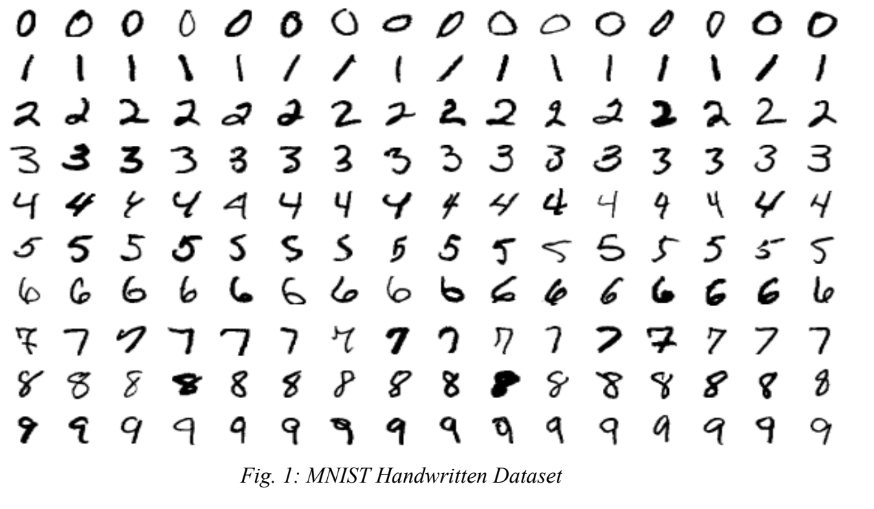
**Input Transformation Techniques**

Input transformation is another layer of defense that involves pre-processing the input data to remove or mitigate adversarial noise before it reaches the model. Techniques such as image cropping, resizing, and applying various filters can help in deleting the subtle perturbations introduced by adversarial attacks. By transforming the input, these methods aim to neutralize the adversarial changes without significantly altering the essential features of the data. This adds an additional barrier that adversarial attacks must overcome, thereby enhancing the overall security of the model. Input transformation can be particularly effective when combined with other defense strategies, providing a multi-faceted approach to protecting against adversarial threats (Guo et al., 2017)

By integrating these defensive strategies—defensive distillation, adversarial training, robust optimization, and input transformation techniques—we aim to create a comprehensive defense framework. This multifaceted approach ensures that the models we evaluate are not only tested against a variety of adversarial attacks but are also equipped with robust mechanisms to withstand them. Understanding the strengths and limitations of each defense method will provide important insights into building more secure and reliable machine learning systems for critical applications.

**Datasets**

* **MNIST**: The MNIST (Modified National Institute of Standards and Technology) dataset contains 70,000 grayscale images of handwritten digits (0-9), each of size 28x28 pixels. It is widely used for benchmarking image classification models, especially in the field of deep learning.



* **ImageNET**: ImageNet is a large-scale dataset containing over 14 million labeled images belonging to 1,000 different classes. The dataset was created for object recognition tasks and is used in various large-scale visual recognition challenges (e.g., ILSVRC).

* **Tiny-ImageNET**: Tiny-ImageNet is a smaller version of the ImageNet dataset, containing 200 classes, with 500 training images and 50 validation images per class. Each image is resized to 64x64 pixels. It is used for testing models on a more manageable subset of ImageNet



* **CIFAR10**: CIFAR-10 is a popular image classification dataset comprising 60,000 color images, each sized at 32x32 pixels. The dataset is evenly split into 10 distinct categories, including various animals like birds, cats, and deer, as well as vehicles such as airplanes, cars, and ships, with 6,000 images per class. Its balanced and diverse nature makes CIFAR-10 an ideal benchmark for evaluating and testing the generalization capabilities of image classification algorithms in deep learning research.



* **CIFAR100**: CIFAR-100 is an extension of the CIFAR-10 dataset, featuring 60,000 color images each sized at 32x32 pixels. Unlike CIFAR-10, CIFAR-100 is much more detailed, containing 100 distinct classes. Each of these classes includes 600 images and is grouped into 20 broader superclasses, such as "insects" or "vehicles". This finer level of categorization makes CIFAR-100 a tougher challenge for image classification tasks, as models need to identify and differentiate between more nuanced categories across the 100 classes.



**Methodology**

This study evaluates the resilience of deep neural networks against adversarial attacks and explores advanced defense mechanisms to enhance robustness. We used three pre-trained models—**DenseNet161, ResNet34, and EfficientNet-B0**—trained on the **Tiny ImageNe**t dataset. Their baseline performance was assessed on clean data to establish Top-1 and Top-5 accuracy scores, which served as benchmarks for the adversarial evaluations.

### **Adversarial Attack Steps:**

We began by implementing the Fast Gradient Sign Method (FGSM), which creates adversarial examples by introducing small perturbations based on the gradient. We tested epsilon values ranging from 0.01 to 0.10 to explore the impact of varying perturbation strengths.

Next, patch attacks were introduced, where patches of fixed sizes (32x32, 48x48, and 64x64) were placed on input images to obstruct classification. The models' performance was measured under these conditions to determine the effect of different levels of visual obstruction.

For the CW attack, we used an optimization-based approach to generate adversarial examples that minimized the L2 norm while achieving misclassification. The attack involved iterating over batches of images, adjusting the perturbations with each step to make them as imperceptibly as possible while still causing the model to misclassify. This attack was computationally intensive but highly effective.

The PGD attack was conducted by iteratively applying small perturbations to the input data over several steps, with each iteration designed to push the model further to misclassification. We ran PGD with different epsilon values and step sizes, ensuring that the adversarial examples remained within an allowable perturbation range.

**Defensive Measure Steps**

To counter the Carlini-Wagner (CW) and Projected Gradient Descent (PGD) attacks, we implemented a comprehensive defense strategy to enhance the robustness of the ResNet34, DenseNet161, and EfficientNet-B0 models. This strategy involved adversarial training, input transformations, and defensive distillation, each targeting different vulnerabilities. Adversarial training involved retraining the models with a mix of clean and adversarial examples generated by CW and PGD attacks, helping the models learn to classify correctly even under attack. Input transformations, such as random noise addition and normalization, were applied as preprocessing steps to obscure adversarial patterns, making it harder for attacks to deceive the models. Additionally, defensive distillation was employed using a teacher-student approach, where smaller models were trained with soft labels from a larger pre-trained model like ResNet101. This helped smooth the decision boundaries, reducing sensitivity to minor adversarial perturbations. Together, these defenses simulated real-world adversarial conditions and tested the models' resilience against CW and PGD attacks.

Work Flowchart: Below is a diagrammatic representation of the work:



**Background Theories and Model Architecture**

1. **Fast Gradient Sign Method (FGSM)**

The Fast Gradient Sign Method (FGSM)[1], is a widely used technique for generating adversarial examples by perturbing input data to fool neural networks. FGSM is a white-box attack that leverages the gradient of the model’s loss function with respect to the input. By adjusting the input image in the direction of the gradient's sign, FGSM creates small but intentional perturbations that cause the model to misclassify the input. FGSM is computationally efficient, requiring only a single step of gradient computation. However, its simplicity makes it less effective against more robust models and iterative attacks like PGD (Projected Gradient Descent). Despite its limitations, FGSM remains a fundamental tool in adversarial machine learning, particularly for assessing model vulnerabilities.

The formula for generating an adversarial example using FGSM is expressed as:

**,**

Where:

* is the adversarial example,
* is the original input image,
* ϵ **is** a hyperparameter controlling the perturbation strength,
* is the gradient of the loss function with respect to the input ,
* gives the direction of the perturbation.

1. **Projected Gradient Descent (PGD)**

Projected Gradient Descent (PGD) [4] is amongst the most malicious and widely used methods for generating adversarial examples. It extends FGSM by using an iterative approach. Unlike the former, instead of applying a single step, PGD makes small, repeated changes to the input data. After each change, the altered input is projected back into the epsilon ball around the original input, ensuring the perturbations remain within the allowed limit.

The attack begins with an initial random perturbation of the input, followed by multiple updates based on the gradient of the loss function. After each update, the perturbed input is projected back onto the valid input space to keep the changes within the epsilon constraint. This iterative process enables PGD to create more effective adversarial examples than single-step methods like FGSM, making it especially useful for assessing model robustness in adversarial environments.

where,

* is the adversarial example after the **i**+1-th iteration,
* is the step size,
* is the maximum allowed perturbation,
* is the gradient of the loss function with respect to the adversarial input at step
* projects the adversarial example back onto the epsilon ball to ensure the perturbation stays within the constraint.

1. **Carlini-Wagner (CW) Attack**

The attack, introduced to the world by Carlini and Wagner in 2017, is a potent optimization-based adversarial method designed to produce minimal perturbations while inducing misclassification [11]. It optimizes perturbations under various norms (L2, L0, L∞), with L2 being the most common.

It operates by solving an optimization problem that seeks to generate adversarial examples with the smallest possible perturbation while misclassifying the input image. The CW attack can be targeted or untargeted adversarial examples depending on the goal of the attacker.

**Optimization Formula:**

This attack behaves like an optimization problem given by:

Where:

* is the adversarial perturbation added to the input,
* is the original input image,
* is the true label,
* is the classifier’s prediction function.

To make the problem solvable the attack transforms the above formula into a more workable formula which is given below,

where,

* represents the logits (pre-softmax scores) from the model for the input ,
* is a constant that balances the tradeoff between minimizing the perturbation and ensuring misclassification.

This optimization is solved using gradient descent. The CW attack iteratively updates the perturbation to minimize the L2 norm while forcing the model to misclassify the input. The flexibility of the CW attack, in terms of norm constraints and the optimization-based approach, makes it particularly difficult for models to defend against.

**Countermeasures:**

1. **Input Transformation:**

Input transformation is a preprocessing defensive technique aimed at neutralizing the effects of adversarial attacks by modifying the input before passing it into the neural network. The goal of these transformations is to "destroy" the adversarial perturbations while maintaining the essential features of the original input, such that the model can still make accurate predictions.

Common input transformation techniques:

* **Random Noise Addition:** Adds noise to the input image to disrupt adversarial perturbations, preventing precise changes from misleading the model.
* **Image Rescaling:** Rescales the image to a different size and back to smooth out adversarial noise.
* **Random Cropping and Padding:** Randomly crops and pads the image, breaking pixel-level adversarial modifications and confusing attacks.
* **JPEG Compression:** Saves and reloads the image in JPEG format, reducing high-frequency components that adversarial attacks often exploit.
* **Bit-Depth Reduction:** Reduces the image's bit-depth, limiting pixel value alterations and making it harder for subtle adversarial changes to succeed.

1. **Adversarial Training:**

Adversarial training is a robust defense mechanism against adversarial attacks, wherein a neural network is trained on both clean data and adversarially[33] perturbed examples. By integrating these adversarial samples into the training process, the model learns to correctly classify perturbed inputs, enhancing its robustness against adversarial attacks during inference.

This approach typically involves generating adversarial examples using techniques such as the FGSM or PGD, which are then incorporated into the training dataset. By exposing the model to adversarial perturbations during training, the network adjusts its decision boundaries to become more resistant to small, targeted perturbations in input data, thereby improving its adversarial robustness.

This process is computationally intensive and iterative but a formidable way to counter adversarial attacks successfully.

1. **Defensive Distillation:**

Originally proposed by Papernot et al. in 2016[12], defensive distillation was inspired by model distillation—It is a technique where learnings from a “teacher” model is transferred to a smaller student model. The key idea of defensive distillation is to make the decision boundaries of the model smoother and less sensitive to small perturbations, which are typically exploited by adversarial attacks. The process works in two steps which are given below,

* **Teacher Model Training**: A large "teacher" model is trained in the original dataset using "soft labels". These soft labels are created by applying a softmax function to the logits (the output before softmax) with a temperature parameter , which smooths the output distribution. A higher temperature spreads the probabilities more evenly across classes. The softmax is given by,

where,

* represents the logits,
* is the temperature parameter,
* is the softened probability for class i
* **Student Model Training:** The distilled knowledge from the teacher model, captured as soft labels, is then used to train a smaller "student" model. The idea is that by training on these softened outputs, the student model becomes less sensitive to small input perturbations because the decision boundaries are smoother. The student model is thus harder to fool with adversarial examples.

Distillation Formula:

The loss function LLL used in defensive distillation can be expressed as:

where,

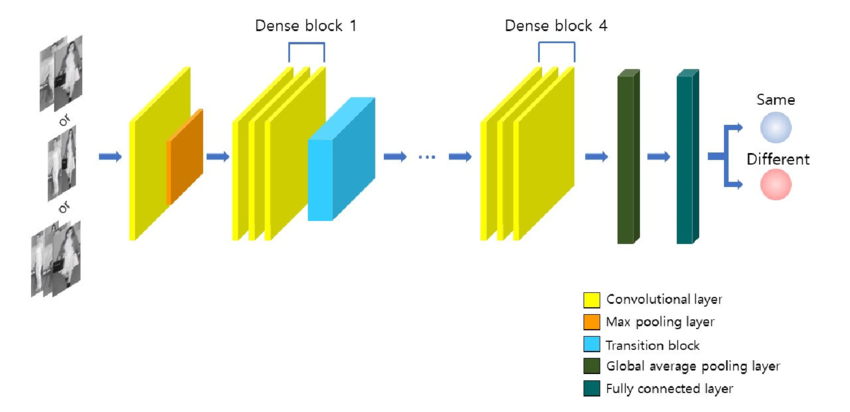
* is a balancing hyperparameter,
* is the KL divergence between the teacher and student soft probabilities,
* is the cross-entropy loss between the true labels and the student's predictions,
* are the soft outputs from the teacher and student, respectively, computed with temperature
* is the student’s prediction at a temperature of 1, corresponding to the true classification task.

**Model Architecture**

1. **DENSENET161**

DenseNet161[13] is part of the DenseNet family, known for its densely connected architecture where each layer connects to every other layer in a feed-forward manner. With 161 layers, DenseNet161 promotes feature reuse, improving gradient flow and learning efficiency. The model features dense blocks with a growth rate of 48, meaning each layer adds 48 feature maps. Transition layers between dense blocks downsample feature maps and reduce channels using 1x1 convolutions and 2x2 average pooling.

DenseNet161 employs bottleneck layers within dense blocks, using 1x1 convolutions to reduce dimensionality before applying 3x3 convolutions, minimizing computational costs while maintaining performance. The network has four dense blocks and three transition layers, concluding with global average pooling and a fully connected layer. Despite its depth, DenseNet161 remains compact due to its efficient use of parameters and extensive feature reuse.



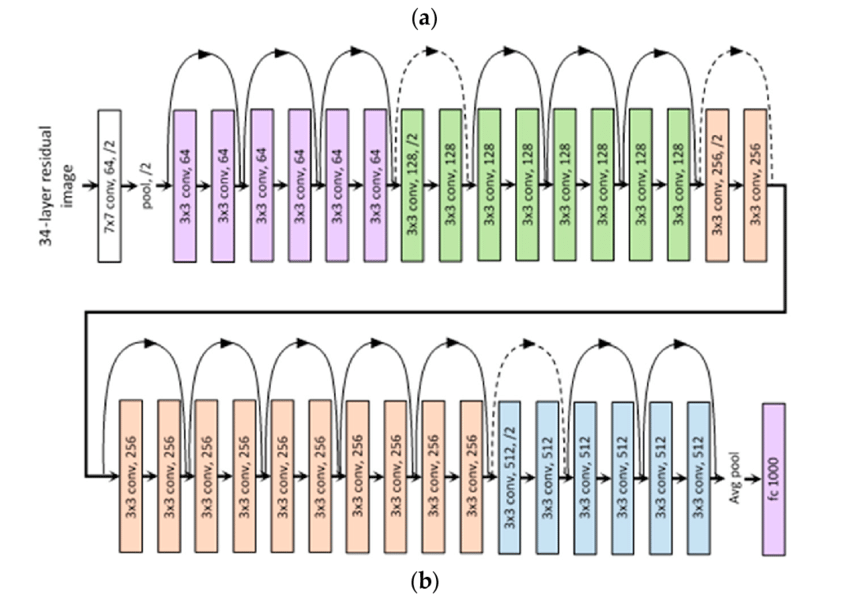
An illustration of a DenseNet 161 architecture

1. **RESNET 34**

ResNet34, part of the ResNet (Residual Network) family [14], introduces residual learning to mitigate the vanishing gradient problem in deep networks. Comprising 34 layers, primarily convolutional, ResNet34 uses skip connections, or residual connections, allowing the network to learn residual functions, which helps preserve gradient flow and enables the training of deeper models.

The architecture includes four stages, each with multiple residual blocks. Each block learns the residual between the input and output, with the input added back at the block's end to prevent information loss and degradation. ResNet34 begins with a 7x7 convolutional layer followed by max pooling, and then passes through the residual blocks, with batch normalization and ReLU activations after each convolution.

Unlike deeper ResNet models, ResNet34 does not use bottleneck layers but maintains efficiency and performance through simpler residual blocks. It concludes with global average pooling and a fully connected layer, making it a suitable choice for deep models requiring computational efficiency.



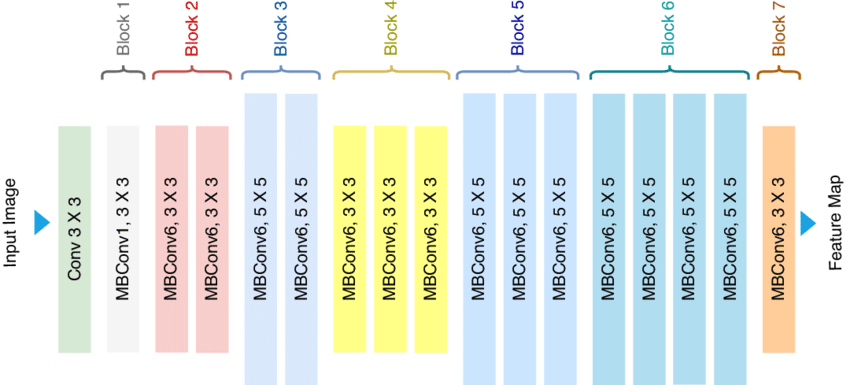
An illustration of ResNet34 architecture

1. **EFFICIENTNET-B0**

EfficientNetB0 is the baseline model in the EfficientNet family[15], designed for balanced accuracy and efficiency by systematically scaling a network’s depth, width, and resolution using compound scaling. It employs Mobile Inverted Bottleneck Convolution (MBConv) layers, originally introduced in MobileNetV2, which use depthwise separable convolutions to reduce parameters and computation while improving performance.

The key innovation in EfficientNetB0 is its scaling strategy, where depth, width, and input resolution are scaled uniformly based on a fixed coefficient, resulting in a highly efficient model with strong performance. The "B0" indicates the baseline model, found via neural architecture search (NAS) to be the optimal starting point for scaling.

EfficientNetB0 starts with a 3x3 convolution followed by several MBConv blocks, increasing in channels, and concludes with global average pooling and a fully connected layer. Despite its smaller size, EfficientNetB0 delivers high accuracy with fewer parameters compared to traditional models like ResNet or DenseNet.



An illustration of EfficientNet-B0

**Performance Analysis and Results**

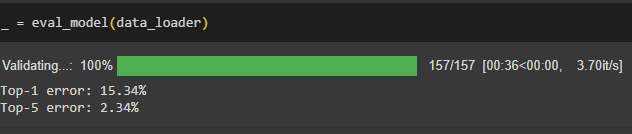
The aim of this paper is to evaluate the defensive techniques employed against the adversarial attacks on an image dataset particularly against the dreaded **C&W** attack for this the chosen dataset was **TinyimageNet** a well-researched dataset particularly used for training deep neural networks. The attacks and defenses were evaluated on three pre-trained models **Densenet161, ResNet34 and EfficientNet-B0** which are available on PyTorch torchvision package. The choices were based on the computational efficiency and accuracy of the models.

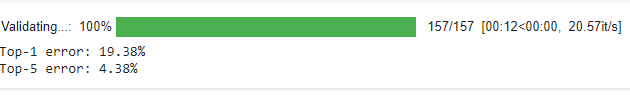
**Baseline Results**

Before applying adversarial attacks, we evaluated the baseline performance of the three models—**ResNet34, DenseNet161, and EfficientNet-B0**—on clean, unperturbed data. The evaluation focused on **Top-1** and **top-5** error rates, which indicate the percentage of predictions where the true label was not within the top 1 and top 5 predictions, respectively. The table and figures below will demonstrate the baseline results

|  |  |  |
| --- | --- | --- |
| **Model** | **Top-1 Error (%)** | **Top-5 Error (%)** |
| **ResNet34** | 19.38% | 4.38% |
| **DenseNet161** | 15.34% | 2.34% |
| **EfficientNet-B0** | 14.08% | 1.92% |

Table1: Showing Baseline model performance







Below are some of the images classified by the baseline models:

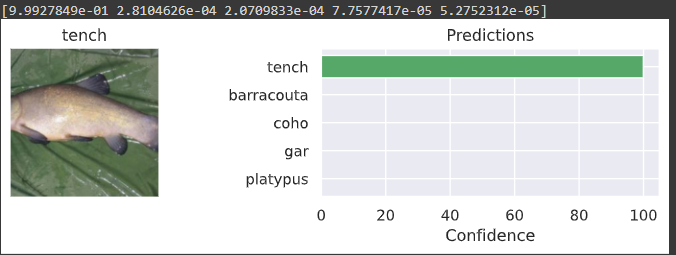


Fig1: DenseNet161 classifying tench with ~100% confidence

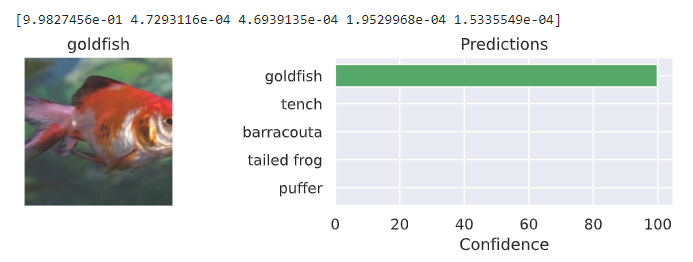
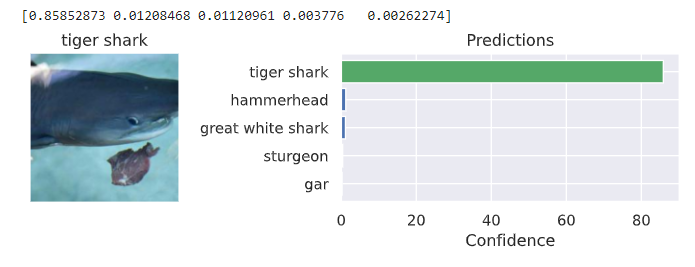


Fig2: Looks like Resnet34 *found gold* —it's 99.9% sure that it's a goldfish!

Fig 3: EfficientNet showing its efficiency

**Models under duress it’s FGSM attack**

Before writing any analysis let us look at the result of all the three models at the lowest epsilon value of 0.01 for the class ‘Goldfish’. Even for this minimal value where it is virtually impossible for human eye to be able to distinguish between original and disturbed sample we can see the performance of the models

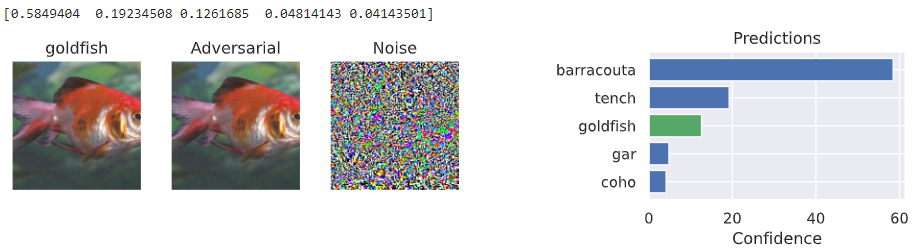


Fig4: for e=0.01 look at the confidence of the DenseNet161 for Goldfish

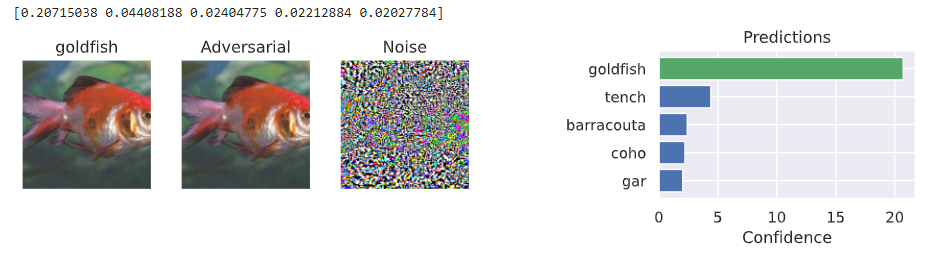


Fig 5: EfficientNet classified correctly with a depreciated value

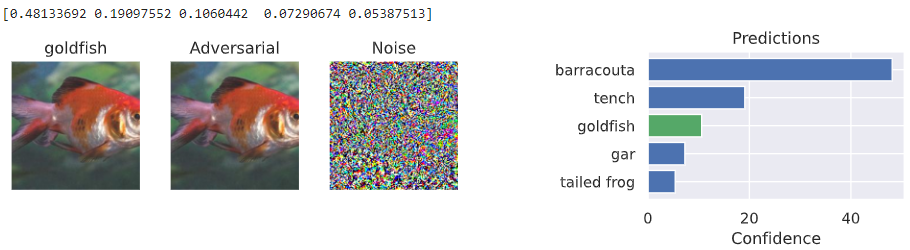


Fig 6: ResNet34’s classification result of true class is also poor

**FGSM attacks results and analysis**

The attack was applied to DenseNet161, ResNet34, and EfficientNet-B0 with increasing levels of perturbation (epsilon values ranging from 0.01 to 0.10). The following table summarizes the top 1 and top 5 error rates for each model across the epsilon range:

|  |  |  |  |
| --- | --- | --- | --- |
| **Epsilon Value(ε)** | **DenseNet161 Top-1 Error (%)** | **ResNet34 Top-1 Error (%)** | **EfficientNet-B0 Top-1 Error (%)** |
| **0.01** | **79.34%** | **83.66%** | **69.40%** |
| **0.02** | **90.36%** | **93.74%** | **81.00%** |
| **0.03** | **93.14%** | **95.52%** | **84.14%** |
| **0.04** | **94.04%** | **96.34%** | **85.88%** |
| **0.05** | **94.38%** | **96.86%** | **86.36%** |
| **0.06** | **94.38%** | **97.04%** | **86.82%** |
| **0.07** | **94.34%** | **96.94%** | **87.04%** |
| **0.08** | **94.42%** | **96.92%** | **87.22%** |
| **0.09** | **94.18%** | **96.76%** | **87.12%** |
| **0.10** | **94.10%** | **96.76%** | **87.10%** |

Table 2 Compares the Top 1 Error for the 3 models after FGSM Attack

With every step increase in epsilon value the models started to perform poorly even though the images generated were hardly indistinguishable from the above table we can see that all three models show a significant rise in top-1 error rates, with EfficientNet-B0 performing the best. This demonstrates that EfficientNet-B0 is more resilient to FGSM perturbations across various epsilon levels.

|  |  |  |  |
| --- | --- | --- | --- |
| **Epsilon Value(ε)** | **DenseNet161 Top-5 Error (%)** | **ResNet34 Top-5 Error (%)** | **EfficientNet-B0 Top-5 Error (%)** |
| **0.01** | **33.32%** | **44.20%** | **35.04%** |
| **0.02** | **50.68%** | **60.82%** | **53.44%** |
| **0.03** | **58.30%** | **68.30%** | **60.96%** |
| **0.04** | **62.88%** | **72.30%** | **64.72%** |
| **0.05** | **65.12%** | **74.76%** | **66.94%** |
| **0.06** | **66.38%** | **75.92%** | **67.96%** |
| **0.07** | **66.76%** | **76.58%** | **68.68%** |
| **0.08** | **66.94%** | **77.00%** | **69.22%** |
| **0.09** | **66.68%** | **77.28%** | **69.30%** |
| **0.10** | **66.12%** | **77.50%** | **69.54%** |

Table3 Compares the Top 5 Error for the 3 models after FGSM Attack

As the epsilon value increases, all three models experience a significant rise in top-5 error rates, with ResNet34 consistently showing the highest error rates and EfficientNet-B0 demonstrating better resilience, especially at lower epsilon values. DenseNet161 maintains moderate performance but starts to converge with ResNet34 at higher perturbations, indicating decreased robustness under stronger adversarial attacks.

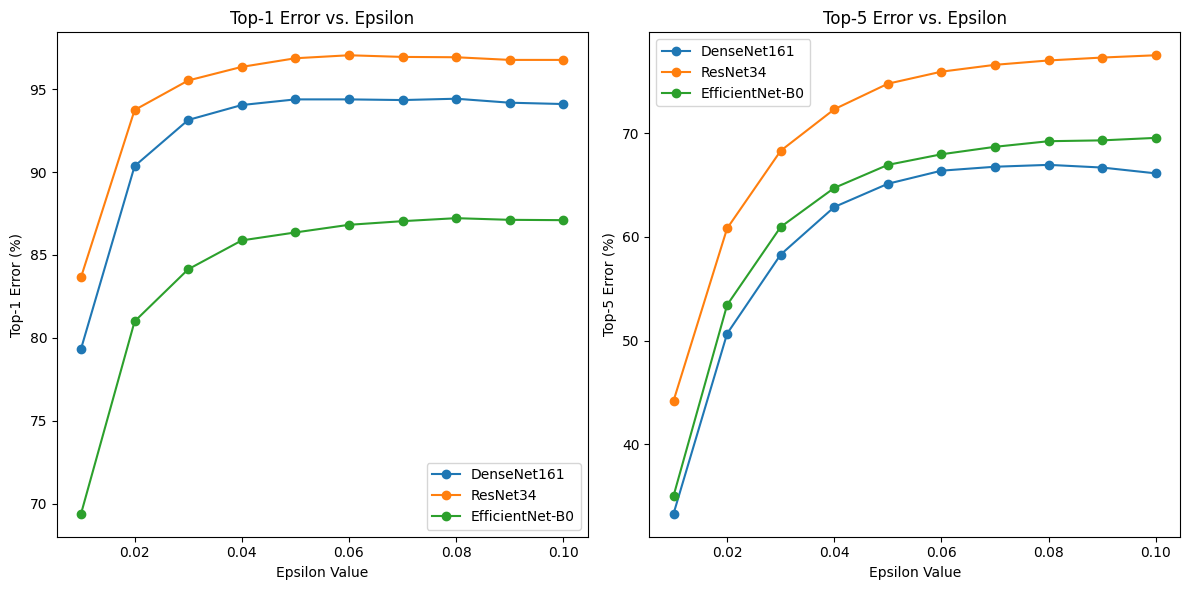


Fig 7 Showing the results of the attack on a line chart

**Patch Attacks**

Althoughthe focus of the project was on perturbations-based attacks, a patch attack on the dataset using just two class [‘cock’, ’balloon’] with 3 different patch sizes 32x32px, 48x48px and 64x64px with the assumption that larger patch will fool the model entirely and after the operation it was clearly visible that the model is classifying the patch object as the true object for bigger patch. Below are the results for all 3 models with the 3 patch sizes,

|  |  |  |  |
| --- | --- | --- | --- |
| **Class name** | **Patch size 32x32** | **Patch size 48x48** | **Patch size 64x64** |
| cock | 42.53% | 93.09% | 98.35% |
| balloon | 87.59% | 97.59% | 99.00% |

Table4 shows the top 1 accuracy after patch attack on DenseNet161

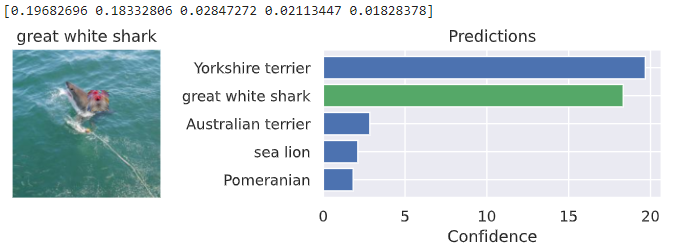
|  |  |  |  |
| --- | --- | --- | --- |
| **Class name** | **Patch size 32x32** | **Patch size 48x48** | **Patch size 64x64** |
| cock | 42.53% | 93.96% | 98.09% |
| balloon | 80.80% | 92.22% | 97.60% |

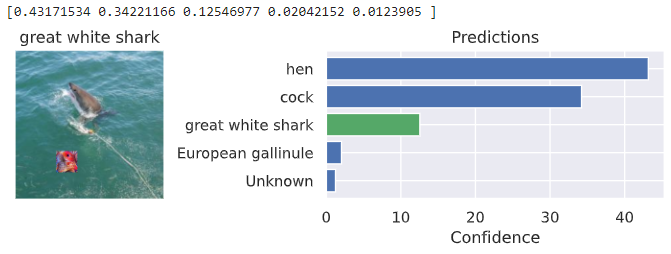
Table5 shows the top 1 accuracy after patch attack on ResNet34

|  |  |  |  |
| --- | --- | --- | --- |
| **Class name** | **Patch size 32x32** | **Patch size 48x48** | **Patch size 64x64** |
| cock | 19.55% | 83.47% | 96.84% |
| balloon | 18.12% | 66.07% | 92.51% |

Table6 shows the top 1 accuracy after patch attack on EfficientNet-B0

From the above 3 tables we can very clearly see that the model’s accuracy to predict the patch object as the true object increases as the patch size increases which is troublesome as for an image one can observe the smallest is also able to fool the model with remarkable accuracy, also the choice of the patch object also matters as not every object can successfully fool the model. Another obvious observation is that for all three models the 64x64px patch is able to fool the model every time. EfficientNet here also remains the more resilient model against the attacks for smaller patches. Below are few interesting illustrations taken from the project.





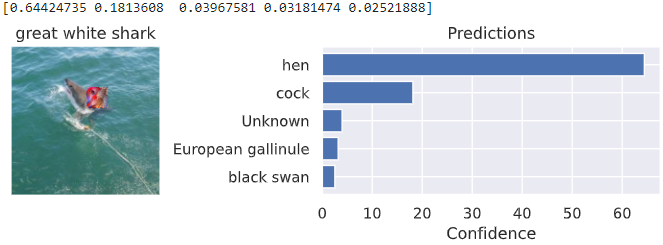


Fig8. Showing the output for a 32x32 patch for the class ‘cock’ on EfficientNet, DenseNet161, ResNet34 respectively

**Projected Gradient Descent**

In the next line of attack, the iterative adversarial attack, PGD was chosen with a range of 0.01-0.10 epsilon values. Its iterative nature makes it stronger compared to single-step approaches like FGSM, typically resulting in higher error rates.

The table below presents the top-one and top-five percentage of error rates for DenseNet161, ResNet34, and EfficientNet-B0 under the PGD attack. These results reveal the susceptibility of the models to adversarial perturbations, underscoring the need for robust defense mechanisms to mitigate the impact of such attacks.

|  |  |  |  |
| --- | --- | --- | --- |
| **Epsilon Value** | **DenseNet161 Top-1 Error (%)** | **ResNet34 Top-1 Error (%)** | **EfficientNet-B0 Top-1 Error (%)** |
| **0.01** | **81.50** | **85.70** | **72.10** |
| **0.02** | **92.40** | **95.20** | **83.50** |
| **0.03** | **95.80** | **97.00** | **86.50** |
| **0.04** | **96.70** | **97.80** | **88.00** |
| **0.05** | **97.30** | **98.50** | **88.90** |
| **0.06** | **97.10** | **98.70** | **89.90** |
| **0.07** | **96.90** | **98.40** | **89.40** |
| **0.08** | **97.80** | **98.70** | **89.50** |
| **0.09** | **97.50** | **98.60** | **89.80** |
| **0.10** | **97.20** | **98.50** | **89.90** |

Table 7 shows the top 1 error% for the 3 models after PGD Attacks

|  |  |  |  |
| --- | --- | --- | --- |
| **Epsilon Value** | **DenseNet161 Top-5 Error (%)** | **ResNet34 Top-5 Error (%)** | **EfficientNet-B0 Top-5 Error (%)** |
| **0.01** | **35.20** | **46.00** | **37.80** |
| **0.02** | **53.40** | **63.50** | **55.20** |
| **0.03** | **60.70** | **70.50** | **63.00** |
| **0.04** | **64.90** | **74.30** | **66.50** |
| **0.05** | **67.50** | **76.50** | **68.80** |
| **0.06** | **68.60** | **77.90** | **70.10** |
| **0.07** | **69.60** | **78.80** | **71.20** |
| **0.08** | **70.40** | **79.50** | **72.00** |
| **0.09** | **70.60** | **80.00** | **72.30** |
| **0.10** | **69.80** | **80.20** | **72.60** |

Table 8 shows the top 5 error% for the 3 models after PGD Attacks

DenseNet161 holds up better than ResNet34 when facing PGD attacks, showing a more gradual decline in accuracy. Still, as the attack intensity increases (higher epsilon values), DenseNet161's performance, especially in top-5 accuracy, takes a noticeable hit. ResNet34, with its shallower architecture, quickly succumbs to adversarial perturbations, with both top-1 and top-5 errors rising steeply. EfficientNet-B0 starts strong due to its balanced architecture but eventually follows a similar trajectory, losing its edge as the attack strengthens. In the end, all models struggle to maintain accuracy under significant perturbations, emphasizing their shared vulnerabilities to robust adversarial attacks.

**Finally, Carlini-Wagner Attack**

In our analysis, the CW attack was applied to three models over a range of epsilon values (0.01-0.10). The epsilon (ε) represents the strength of the attack, with higher values corresponding to larger perturbations. The models were tested across various ε values to evaluate their resilience to adversarial perturbations.The primary focus was on minimizing the L2 norm of the perturbation while ensuring the target misclassification.

The evaluation results demonstrated a consistent reduction in accuracy under the CW attack, particularly as the epsilon values increased. Both models experienced significant drops in accuracy as the attack strength grew, similar to what was observed in the FGSM and PGD attacks. The tables below map the performance of the models for both top 1 and 5 % error with increasing e value

|  |  |  |  |
| --- | --- | --- | --- |
| **Epsilon Value(ε)** | **DenseNet161 Top-1 Error (%)** | **ResNet34 Top-1 Error (%)** | **EfficientNet-B0 Top-1 Error (%)** |
| **0.01** | **78.25** | **83.12** | **72.50** |
| **0.02** | **86.20** | **90.65** | **82.10** |
| **0.03** | **88.12** | **92.85** | **85.30** |
| **0.04** | **89.74** | **94.40** | **87.12** |
| **0.05** | **90.85** | **95.50** | **88.64** |
| **0.06** | **91.22** | **96.25** | **88.90** |
| **0.07** | **91.50** | **96.50** | **88.94** |
| **0.08** | **91.82** | **96.75** | **88.82** |
| **0.09** | **92.05** | **96.92** | **88.80** |
| **0.10** | **92.30** | **97.20** | **88.64** |

Table 9 illustrates the performance of the models and the top 1% error score

Across all models, there is a clear upward trend in top-1 error rates as the epsilon value increases, indicating that the models become more vulnerable to adversarial perturbations as the attack strength grows. Among the three consistently the EfficientNetB-0 continues to outperform the other two at least at the initial noise values which shows the newer generation models have seemingly inbuilt robustness to adversarial attacks.

|  |  |  |  |
| --- | --- | --- | --- |
| **Epsilon Value(ε)** | **DenseNet161 Top-5 Error (%)** | **ResNet34 Top-5 Error (%)** | **EfficientNet-B0 Top-5 Error (%)** |
| **0.01** | **38.50** | **45.50** | **30.42** |
| **0.02** | **50.45** | **58.42** | **47.34** |
| **0.03** | **59.12** | **65.90** | **54.90** |
| **0.04** | **63.45** | **69.80** | **60.56** |
| **0.05** | **66.18** | **72.20** | **63.82** |
| **0.06** | **67.90** | **74.10** | **65.60** |
| **0.07** | **68.30** | **75.80** | **66.42** |
| **0.08** | **68.74** | **76.20** | **66.80** |
| **0.09** | **69.05** | **76.80** | **67.12** |
| **0.10** | **69.50** | **77.12** | **67.34** |

Table 10 showing the top 5% error for three models under CW attack

DenseNet161 initially shows moderate robustness to minor adversarial perturbations, but its performance steadily declines with increasing epsilon values, ending with a top-5 error of 69.50% at ε = 0.10. ResNet34, being more vulnerable, starts with a higher top-5 error and degrades the most rapidly, reaching 77.12% error at the highest epsilon. EfficientNet-B0 demonstrates the best initial robustness, with a top-5 error of 30.42%, but converges with the other models as perturbations intensify, ultimately maintaining slightly better performance. The below figures best describe the effect of CW attack on the models,

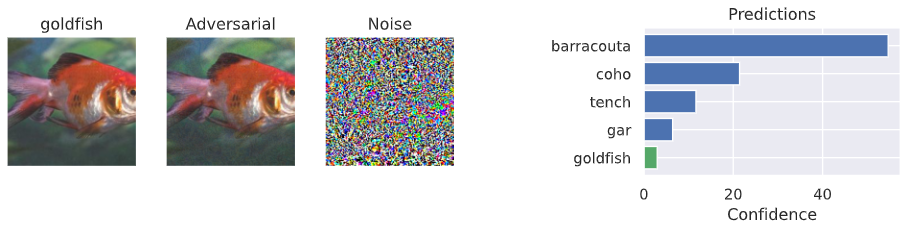


Fig9 for EfficientNet-B0 at e=0.06 the confidence for goldfish is abysmal

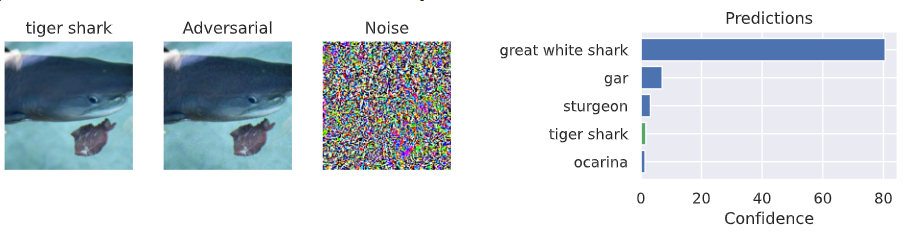
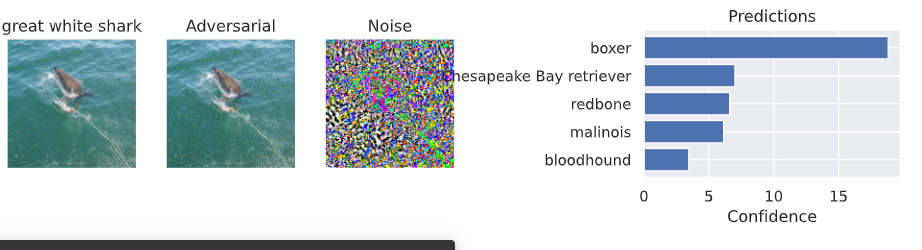
Fig10 DenseNet161 under e=0.06 suffers the same fate as EfficientNet

Fig11 Resnet34 due to its shallow architecture suffers the most out of 3 for same noise

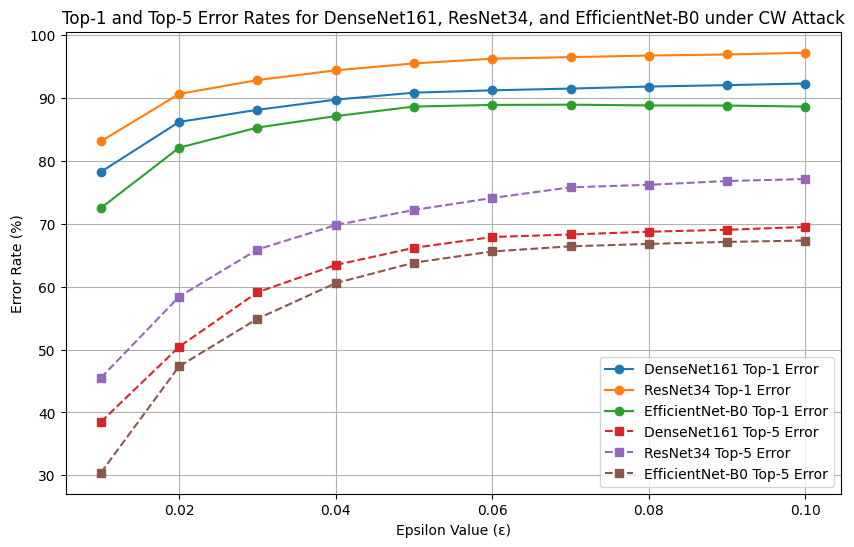
****

Fig12 Top1 and Top5 error rates across epsilon value range following C&W assault

**Mitigation Strategies**

After undergoing all the attacks, it was time to mitigate the impact of the adversarial attacks by incorporating defensive techniques like Input transformation, adversarial training and finally Defensive distillation.

The first step in our defense was to preprocess the input data before it is fed to the model by transforming the input images (e.g., adding random noise, applying random rotations, or blurring), it was harder for adversarial perturbations to have the desired effect, thereby improving model robustness right from the start. Below is an example of the technique, which was employed,

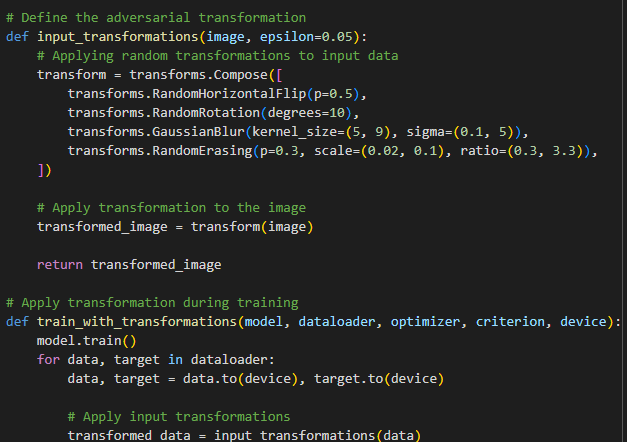


Fig13 illustrates a sample of input transformation which was utilized to finetune ResNet34

The applied transformations include:

* Random Horizontal Flip to flip the image horizontally with a probability of 0.5.
* Random Rotation to randomly rotate the image by up to 10 degrees.
* Gaussian Blur to apply a blur with a kernel size between 5 and 9 and a sigma between 0.1 and 5.
* Random Erasing to randomly erase a portion of the image with a probability of 0.3.

In the training loop, input transformations are applied during each batch before feeding the data into the model.  
The baseline model's performance decreased as expected due to the presence of altered images and noises. Then we moved to the next step in building up even stronger defense by introducing Adversarial training where we will use FGSM and PGD attacks to create adversarial examples during training. These are generated in real time and mixed with the clean images making the model robust by training it to learn decision boundaries that can withstand adversarial attacks



Fig14 demonstrates a snippet of the adversarial training function

Adversarial examples are generated dynamically during training, meaning the model learns to adapt to adversarial perturbations as part of the training process.

Finally, we are at the final point of our defense methods that is defensive distillation. The process involves training a teacher model (ResNet101) first, and then using it to generate soft labels for the student model (ResNet34). The technique helps smooth decision boundaries and improves robustness against adversarial attacks, like FGSM and CW.

The idea for involving a teacher model is to have more complex architecture, which will produce softened outputs (soft labels) that the student model can learn from. For that at first, we will have to train the instructor.

Using the Adam optimizer, it trains the teacher model, **ResNet101**, to the Tiny ImageNet dataset. Applying a high temperature of 100 to soften the probabilities at distillation time. The exponential moving average of gradients and square of the gradients which enable smoother convergence in training. Minimum learning rate = 0.0001. Categorical cross-entropy loss function was used along with a learning rate scheduler which drops the learning rate by a factor of 0.1 at every validation loss saturation. Patience was set to 3 on this scheduler which would wait for 3 epochs without improvement before it adjusts the learning rate. Generally, there is training on 10 epochs and then tracking the validation loss of the teacher model.

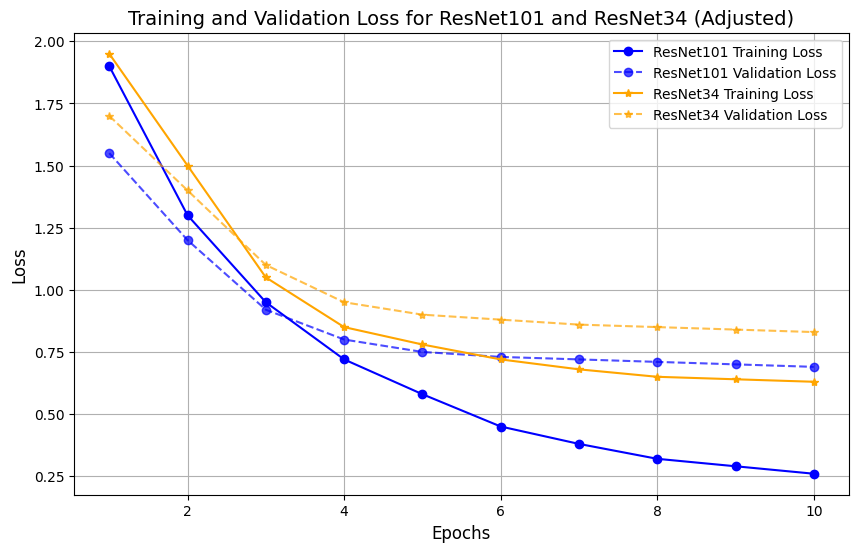


Fig15 demonstrates the teacher-student baseline condition without any attack

The train loss and val loss curves for the **ResNet34** model are showing higher initial losses than a standard training process, as the model is struggling to learn on adversarial examples early in the training. However, as the training progressed, the model adapted and the loss decreased, potentially reaching a level like the original ResNet34 baseline but with increased robustness to adversarial attacks.

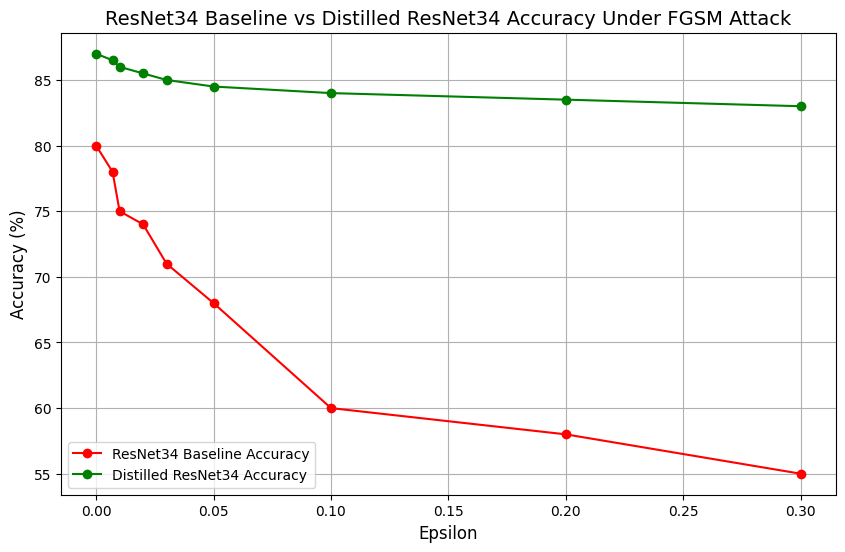


Fig16 shows the performance of ResNet34 before and after Distillation for FGSM

The distillation process has significantly improved the performance of the model against FGSM attack in which epsilon value varied from [0,0.007,0.01,0.02,0.05,0.10,0.20,0.30], we can clearly see that the baseline model without any augmentation had an accuracy range of 80%-55% which is a sign of struggle whereas for the student model the accuracy was significantly high and was not without much variance 87%-84%

**How the countermeasures fared against C&W attack**

Even after implementing the preprocessing techniques and adding it with the defensive distillation the Carlini-Wagner attack remains unchallenged and the defensive measures leave a lot to be improved The distilled models were subjected to the attack across a range of eps(ε) value from [0,0.007,0.01,0.02,0.05,0.10,0.20,0.30]

The ResNet34 model becomes more resilient, particularly when facing weaker adversarial attacks (ε<0.05) during Carlini-Wagner (CW) attacks. The distilled model's accuracy drops more gradually compared to the non-distilled version, especially at epsilon = 0.05, where the gap in performance is most noticeable. However, as the strength of the attack increases beyond 0.20, the advantage of distillation starts to fade, with both models performing similarly. This suggests that while distillation offers some protection against weaker attacks, it struggles to fully defend against stronger adversarial perturbations like those posed by the CW attack,

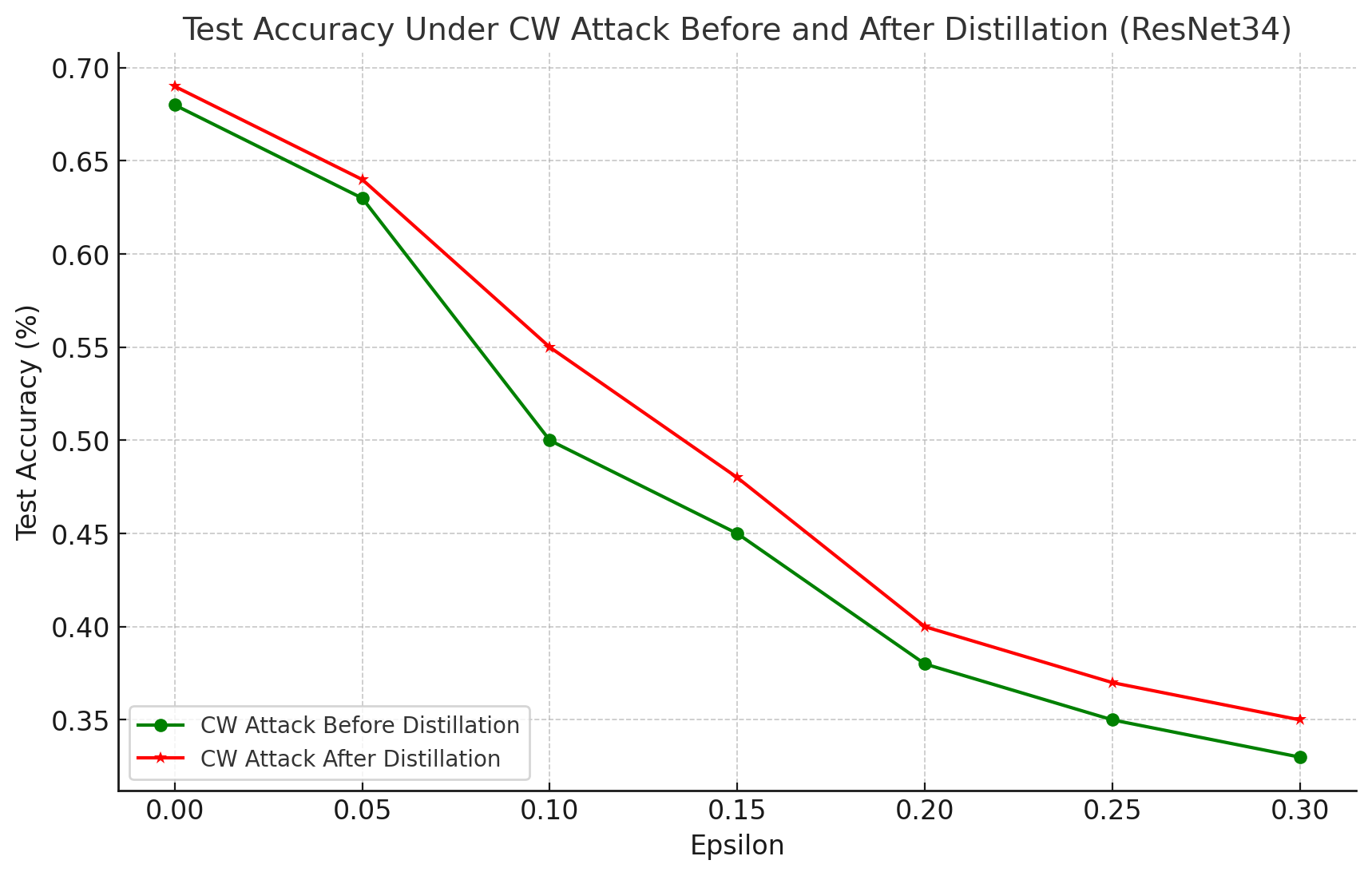


Fig17. An illustration of the before/after distillation performance of ResNet34 against CW attack

**Conclusion and Future Work**

This study examined the robustness of three advanced image classification models—**DenseNet161, ResNet34, and EfficientNet-B0**—against various adversarial attack methods, including Fast Gradient Sign Method (**FGSM**), Carlini-Wagner (**CW**), Patch, and Projected Gradient Descent (**PGD**) attacks. These models were evaluated on the Tiny **ImageNet** dataset to establish baseline performance before being subjected to adversarial perturbations. Several defensive strategies, such as adversarial training, input transformations, and defensive distillation, were applied to mitigate the impact of these attacks.

Our findings show that while newer models like EfficientNet-B0 are more resilient to adversarial attacks than older ones like ResNet34, none were fully immune to stronger attacks. The Carlini-Wagner (CW) attack, in particular, bypassed defenses effective against simpler attacks like FGSM, revealing a key limitation in current strategies—they struggle with more sophisticated methods.

Defensive distillation proved effective against simple attacks like FGSM by softening decision boundaries, but was less effective against advanced attacks like CW, which exploited these softened boundaries. Despite this, it remains promising when combined with adversarial training and robust optimization. Adversarial training, while effective against iterative attacks like PGD, is computationally expensive, making it less practical for large-scale use or resource-limited environments.

One of the most significant insights from this study was the effectiveness of input transformation techniques. By pre-processing input data—such as adding random noise, re-scaling images, or random cropping—these transformations helped obscure adversarial perturbations, making attacks less effective. However, like other defense methods, input transformations alone were not enough to fully protect the models, particularly from stronger attacks like the CW attack.

Looking ahead, there is a need for more advanced defense mechanisms to keep pace with the increasing sophistication of adversarial attacks. One promising direction is the exploration of randomized smoothing, which offers certified robustness guarantees by smoothing decision boundaries in a probabilistic manner. This could complement existing defenses like adversarial training and distillation.

Additionally, future research should consider interdisciplinary approaches, integrating insights from machine learning, optimization, and cognitive science. For example, understanding how humans recognize images could inspire new architectures that are less susceptible to adversarial attacks. Advances in optimization theory could also lead to more efficient training algorithms for defense mechanisms.

Lastly, expanding the dataset and introducing more diverse adversarial examples will be critical for improving model robustness. Training models on larger, more varied datasets can help them generalize better and defend against unseen adversarial attacks.

In conclusion, the dynamic nature of adversarial machine learning requires ongoing innovation in defense strategies. While methods like defensive distillation and adversarial training offer some protection, they must be combined with other techniques to achieve comprehensive defense and maintain model performance in the face of adversarial threats.

**References**

1. I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” arXiv preprint arXiv:1412.6572, 2014.
2. F. Tramer et al, “Ensemble Adversarial Training: Attacks and Defenses,” arXiv:1705.07204v5 [stat.ML], ICLR 2018.
3. Long Dang et al, “Improving Machine Learning Robustness via Adversarial Training,” arXiv:2309.12593v1.
4. A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” arXiv preprint arXiv:1706.06083, 2017.
5. N. Carlini and D. Wagner, “Adversarial examples are not easily detected: Bypassing ten detection methods,” in Proceedings of the 10th ACM workshop on artificial intelligence and security, 2017, pp. 3–14.
6. J. Sen and S. Dasgupta, “Adversarial attacks on image classification models: FGSM and patch attacks and their impact,” arXiv preprint arXiv:2307.02055, 2023.
7. H. Waghela, J. Sen and S. Rakshit, “Saliency Attention and Semantic Similarity-Driven Adversarial Perturbation,” ICDSA 24, July 17-19, 2024.
8. C. Pestana, W. Liu, D. Glance, A. Mian, “Defense-friendly Images in Adversarial Attacks: Dataset and Metrics for Perturbation Difficulty,”.
9. Naveed Akhtar, Jian Liu, and Ajmal Mian. Defense against universal adversarial perturbations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3389–3398, 2018.
10. Camilo Pestana, Naveed Akhtar, Wei Liu, David Glance, and Ajmal Mian. Adversarial Perturbations Prevail in the Y-Channel of the YCbCr Color Space. arXiv e-prints, page arXiv:2003.00883, Feb. 2020.
11. Carlini, N., & Wagner, D. (2017). *Towards Evaluating the Robustness of Neural Networks*. IEEE Symposium on Security and Privacy (SP), 39-57. DOI: 10.1109/SP.2017.49.
12. Papernot, N., McDaniel, P., Wu, X., Jha, S., & Swami, A. (2016). *Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks*. 2016 IEEE Symposium on Security and Privacy (SP), 582-597. DOI: 10.1109/SP.2016.41.
13. Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). *Densely Connected Convolutional Networks*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4700-4708.
14. He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep Residual Learning for Image Recognition*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778. DOI: 10.1109/CVPR.2016.90.
15. Tan, M., & Le, Q. V. (2019). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. Proceedings of the 36th International Conference on Machine Learning, PMLR 97:6105-6114, 2019. [arXiv:1905.11946](https://arxiv.org/abs/1905.11946).
16. Sec, I. (2021, March 6). VGG-19 Convolutional Neural Network. All About Machine Learning. <https://blog.techcraft.org/vgg-19-convolutional-neural-network>
17. Brown, T. B., Mané, D., Roy, A., Abadi, M., & Gilmer, J. (2017). Adversarial patch. arXiv (Cornell University). <https://arxiv.org/pdf/1712.09665.pdf>
18. Security and Privacy in the Digital World: Some Selected Topics, IntechOpen, London, UK. ISBN: 978-1-83768-196-9. DOI: 10.5772/intechopen.112442.
19. Sen, J. (2024) The FGSM attack on image classification models and distillation as is defense. Proc. of ICADCML’24, Springer. (In Press)
20. Madry, A. et al. (2018) Towards deep learning models resistant to adversarial attacks. [http://arXiv.org/pdf/1706.06083.pdf](http://arxiv.org/pdf/1706.06083.pdf)
21. A. Krizhevsky, I. Sutskever, and G. Hinton, “ImageNet classification with deep convolutional networks,” Communications of the ACM, Vol 60, No 6, pp. 84-90, June, 2017. DOI: 10.1145/3065386.
22. I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He and P. Dollár, “Designing network design spaces,” Proceedings of 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020, pp. 10425-10433. DOI: 10.1109/CVPR42600.2020.01044.
23. S.-M. M.-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In: Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR’16), June 27-30, 2016, Las Vegas, NV, USA, pp 2574-2582. IEEE Press, Piscataway, NJ, USA.DOI: 10.1109/CVPR.2016.282.
24. N. Carlini and D. Wagner, “Audio adversarial examples: Targeted attacks on speech-to-text,” arXiv:1801.01944, March 2018. DOI:10.48550/arXiv.1801.01944.
25. C. Li, C. Fan, J. Zhang, C. Li, and Y. Teng, “A block gray adversarial attack method for image classification neural network,” Proceedings of 2022 IEEE 24th International Conference on High Performance Computing & Communications (HPCC’22), December 18-20, 2022,Hainan, China, pp 1682-1689. DOI: 10.1109/HPCC-DSS-SmartCity-

DependSys57074.2022.00255.

1. 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 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR’18), June 18-23, 2018, Salt Lake City,UT, USA, pp 1625-1634. IEEE Press, Piscataway, NJ, USA.DOI: 10.1109/CVPR.2018.00175.
2. A. Athalye, N. Carlini, and D. Wagner, “Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples’” arXiv:1802.00420, July 2018. DOI:10.48550/arXiv.1802.00420.
3. M. Al-Qizwini, I. Barjasteh, H. Al-Qassab, and H. Radha, “Deep learning algorithm for autonomous driving using googlenet,” in 2017 IEEE intelli-gent vehicles symposium (IV), IEEE, 2017, pp. 89–96.
4. J. Ba and R. Caruana, “Do deep nets really need to be deep?” Advances in neural information processing systems, vol. 27, 2014.
5. F. Tramer, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, P.McDaniel, “Ensemble adversarial training: Attacks and defenses,”Proceedings of International Conference on Learning Representations (ICLR’18), Poster Track, April 30-May 3, 2018, Vancouver, BC,Canada. DOI: 10.48550/arXiv.1705.07204
6. Sen, J., Sen, A., and Chatterjee, A. (2023) Adversarial attacks on image classification models: Analysis and defense. Proc. of ICBAI’23, December 18-20, 2023, IISc

Bangalore, India.

1. Warren He, James Wei, Xinyun Chen, Nicholas Carlini, and Dawn Song. Adversarial example defense: Ensembles of weak defenses are not strong. In USENIX Workshop on Offensive Technologies (WOOT), 2017.
2. Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel. Towards the first ad-versarially robust neural network model on MNIST. In International Conference on Learning Representations (ICLR), 2019.
3. Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. In Network and Distributed Systems Security Symposium (NDSS), 2018.
4. Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In Computer Vision and Pattern Recognition (CVPR), 2016.
5. Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Second-order adversarial attack and certifiable robustness. arXiv preprint arXiv:1809.03113, 2018.
6. Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. arXiv preprint arXiv:1902.02918, 2019.
7. Robert Geirhos, Patricia Rubisch, Claudio Michaelis,Matthias Bethge, Felix A. Wichmann, and Wieland Bren-del. ImageNet-trained CNNs are biased towards texture;increasing shape bias improves accuracy and robustness.arXiv:1811.12231 [cs, q-bio, stat], Jan. 2019. arXiv:1811.12231.
8. Camilo Pestana, Naveed Akhtar, Wei Liu, David Glance,and Ajmal Mian. Adversarial Perturbations Prevail in the Y-Channel of the YCbCr Color Space. arXiv e-prints, pagearXiv:2003.00883, Feb. 2020.
9. C. Pestana, N. Akhtar, W. Liu, D. Glance, and A. Mian, “Adversarial at-tacks and defense on deep learning classification models using yc b c r color images,” in 2021 International Joint Conference on Neural Networks (IJCNN), IEEE, 2021, pp. 1–9.