Dependable Artificial Intelligence

Adversarial Robustness

TEAM MEMBERS:
Khushi Maheshwari [B21AI052]
Rutuja Janabandhu [B21AI017]

INTRODUCTION

Adversarial attacks pose a significant threat to the reliability and security of deep neural networks (DNNs) deployed in various real-world applications, ranging from image classification to autonomous driving systems. These attacks are specifically crafted perturbations added to input data with the intent of misleading the model's predictions while being imperceptible to human observers. In recent years, the vulnerability of DNNs to such attacks has garnered increasing attention from the research community, prompting the development of robust defense mechanisms.
In this assignment, we address the challenge of adversarial robustness by training a deep neural network using a combination of architectural design choices and training techniques.
We takea convolutional neural network (CNN) architecture. Our model, named NeuralNet, consists of two convolutional layers followed by two fully connected layers. We employ rectified linear unit (ReLU) activation functions to introduce non-linearity and max-pooling layers to downsample the feature maps, facilitating hierarchical feature extraction.
The objective of this assignment is to evaluate the robustness of the NeuralNet model against various adversarial attack methods, including Projected Gradient Descent (PGD), Fast Gradient Sign Method (FGSM), and Deepfool.

Dataset:

We leverage the MNIST dataset, a widely used benchmark dataset for digit classification tasks, to train and evaluate the model. The MNIST dataset comprises 28x28 grayscale images of handwritten digits (0-9), making it suitable for assessing the performance of our model in the context of image classification.

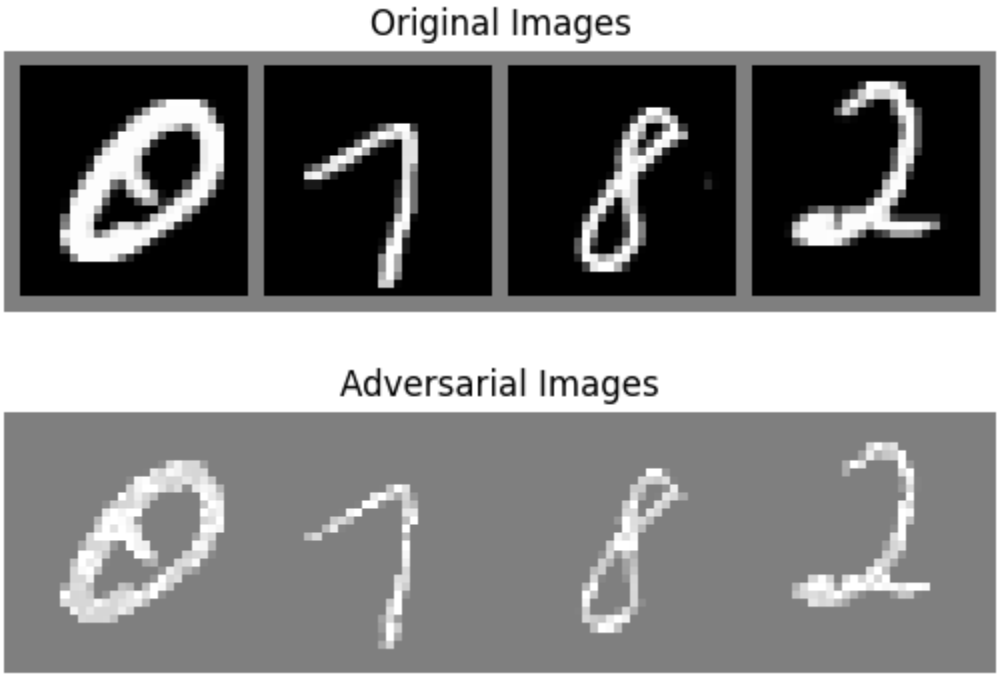
Projected Gradient Descent (PGD) Attack:

The PGD attack is a powerful iterative method used to generate adversarial examples. It works by iteratively perturbing the input data in the direction that maximizes the loss, subject to a maximum perturbation size ε. At each iteration, the perturbation is projected back onto the ε-ball centered around the original input to ensure it remains within the desired bounds. This projection step helps prevent the generated adversarial examples from deviating too far from the original data distribution, making them more effective in real-world scenarios.

We instantiate the PGD attack with the following parameters:

Maximum perturbation size (ε): 0.3, Step size (α): 2/255, Number of iterations: 40

These parameters are chosen to ensure the generation of strong adversarial examples while keeping the perturbations imperceptible.



Integration into Training Loop:

Within the training loop, we utilize the PGD attack to generate adversarial examples from the original input images. The generated adversarial examples are then used to compute the loss and update the model parameters via backpropagation. This process enables the model to learn from both clean and adversarial data, enhancing its robustness against adversarial attacks.

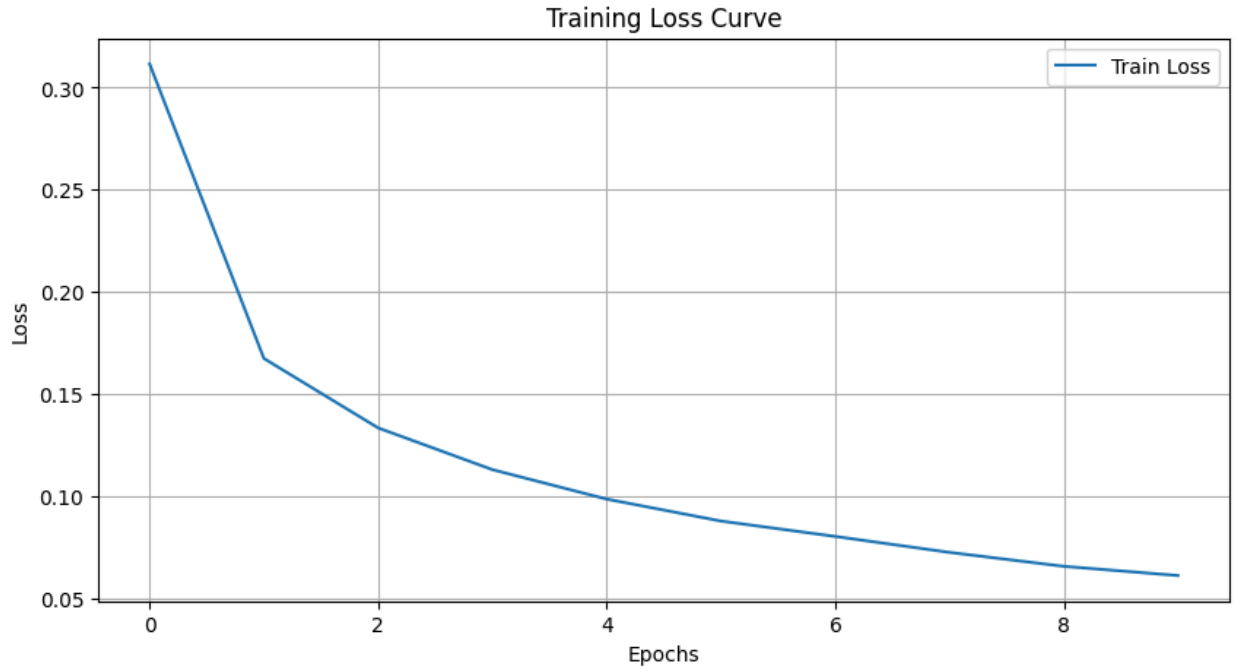
Evaluation of Model Robustness:

We evaluate the performance of our model against adversarial examples generated using the PGD attack. By feeding these adversarial examples into the model and

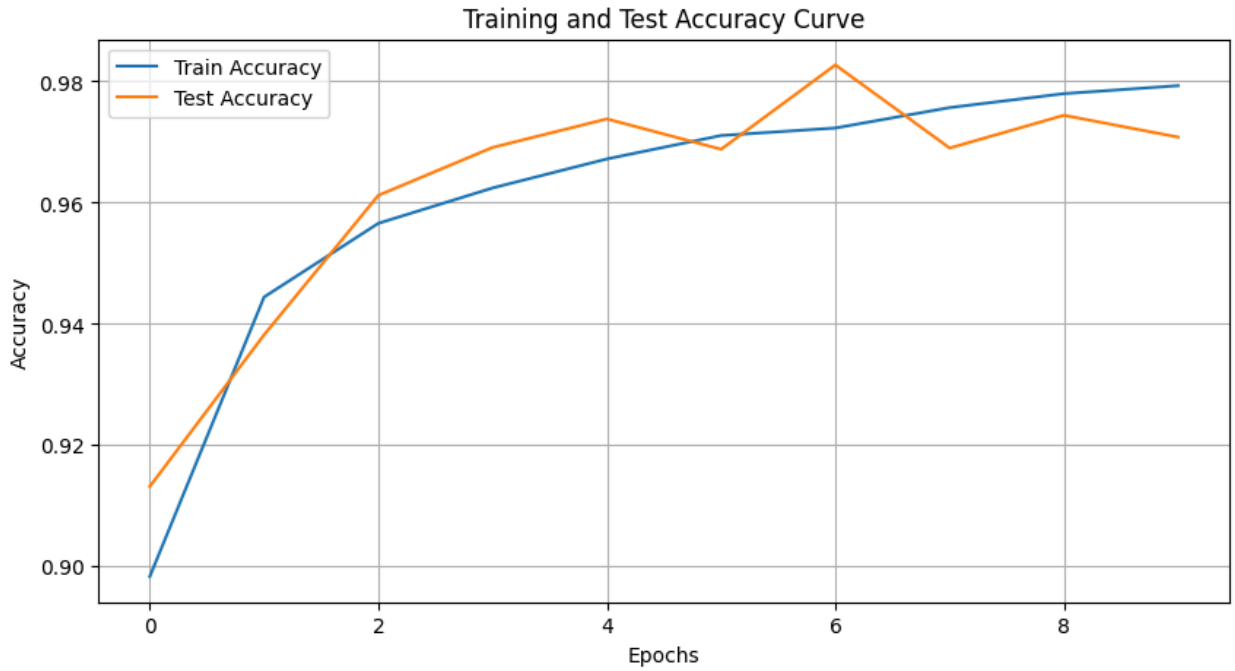
comparing the predicted labels with the ground truth labels, we assess the model's ability to maintain accuracy in the presence of perturbations.

Results and Analysis:

Training Loss Curve:



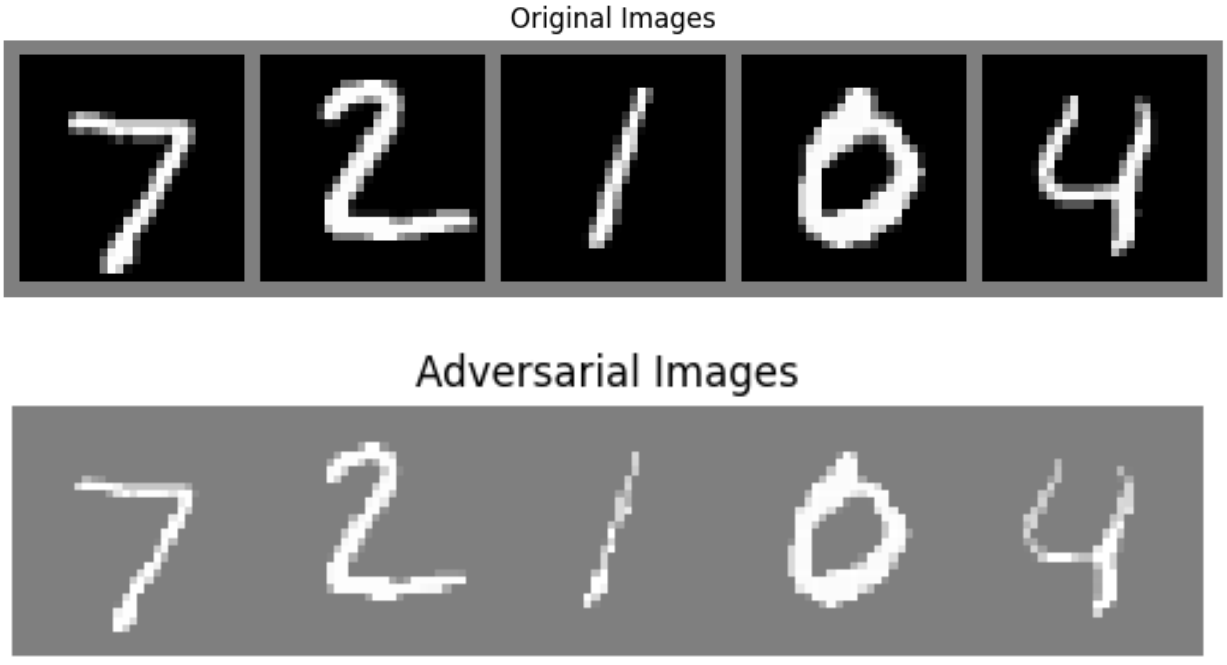
Training and Test Accuracy Curve:



Analysis:
The testing accuracy shows an increasing trend, indicating that the model's performance on unseen data (test set) is improving over epochs. The final testing accuracy is approximately 97.08%, which is slightly lower than the training accuracy but still quite high. This suggests that the model is generalizing well to unseen data, demonstrating its ability to make accurate predictions in real-world scenarios.
By analyzing the performance of the model on adversarial examples, we can assess its resilience to potential attacks and its ability to maintain accuracy in the presence of perturbations.

Fast Gradient Sign Method (FGSM) Attack:

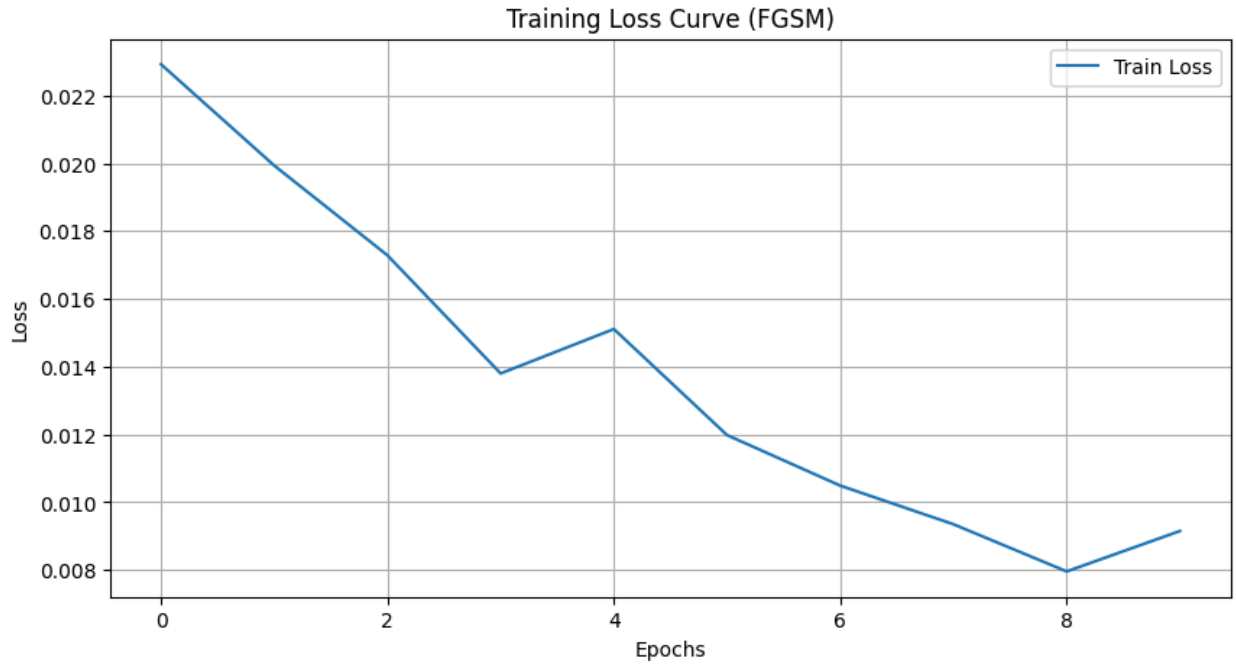
The FGSM attack is a fast and computationally efficient method for crafting adversarial examples by perturbing the input data in the direction of the gradient of the loss function with respect to the input. It generates perturbations that maximize the loss, leading to misclassification by the model.
We instantiate the FGSM attack with the following parameter:
Maximum perturbation size (ε): 0.3
This parameter determines the magnitude of perturbation applied to the input data to generate adversarial examples.



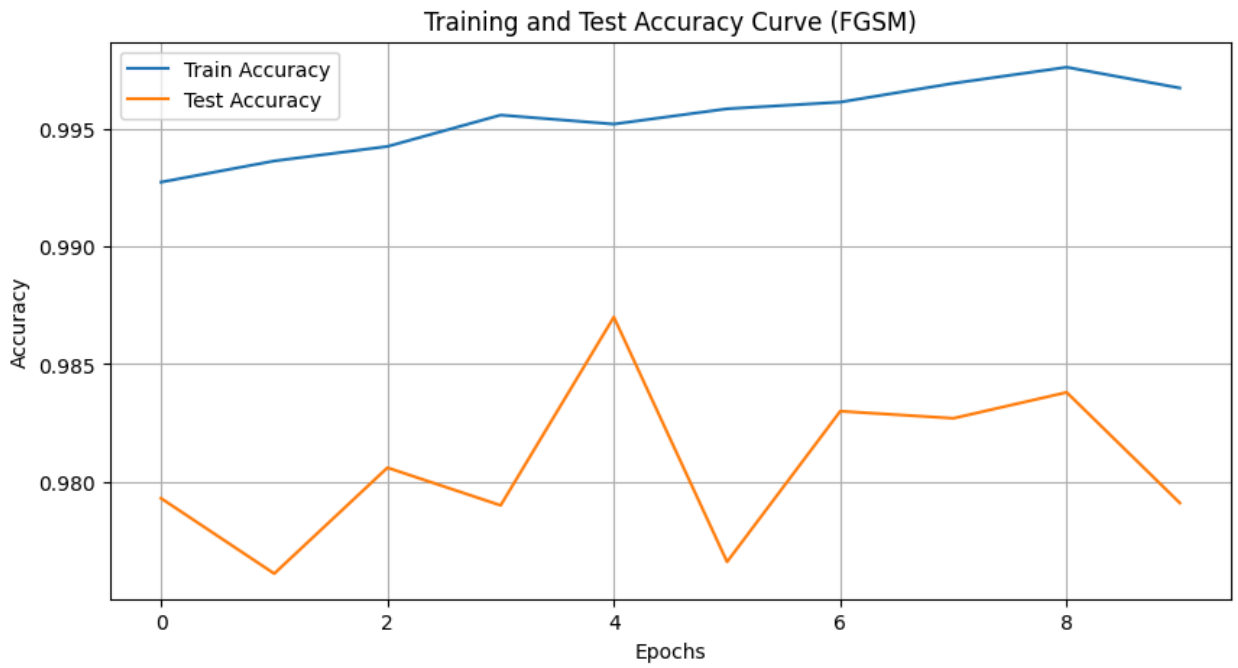
Integration into Training Loop:
Within the training loop, we utilize the FGSM attack to generate adversarial examples from the original input images. The generated adversarial examples are then used to compute the loss and update the model parameters via backpropagation.

Evaluation of Model Robustness:
We evaluate the performance of our model against adversarial examples generated using the FGSM attack. By analyzing the model's accuracy on adversarial data, we assess its robustness and resilience against adversarial perturbations.

Results and Analysis:
Training Loss Curve:



Training and Test Accuracy Curve:



Analysis:
The testing accuracy is consistently lower than the training accuracy, indicating that the model's performance on clean data is slightly better than on adversarial examples. By generating adversarial examples using the FGSM attack, we assess the model's robustness against perturbations. The observed testing accuracy suggests that the model maintains a high level of accuracy even in the presence of adversarial perturbations.
While the FGSM attack may not generate as strong adversarial examples as PGD, it offers a computationally efficient alternative for evaluating model robustness and resilience against adversarial perturbations.

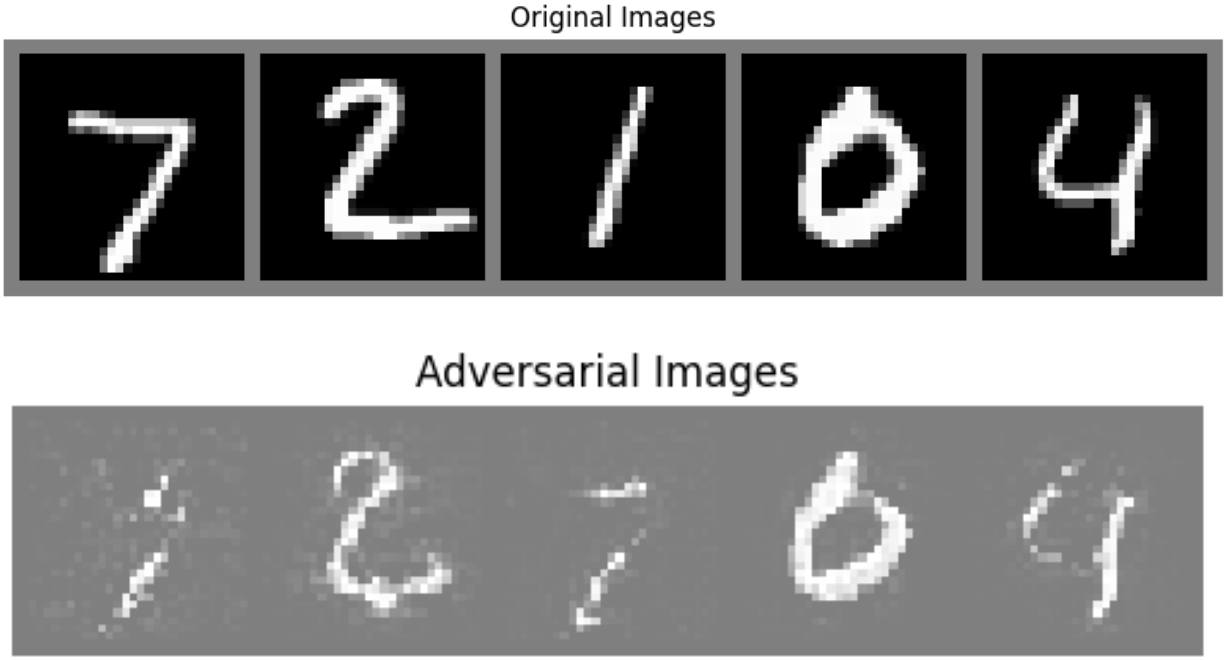
DeepFool Attack

The DeepFool attack is an iterative technique designed to generate adversarial examples by iteratively perturbing the input data in the direction of the decision boundary of a machine learning model. Unlike some other adversarial attack methods that rely on gradient information, DeepFool operates by computing the minimal perturbation required to cause a misclassification, thereby making it more efficient and effective in certain scenarios.

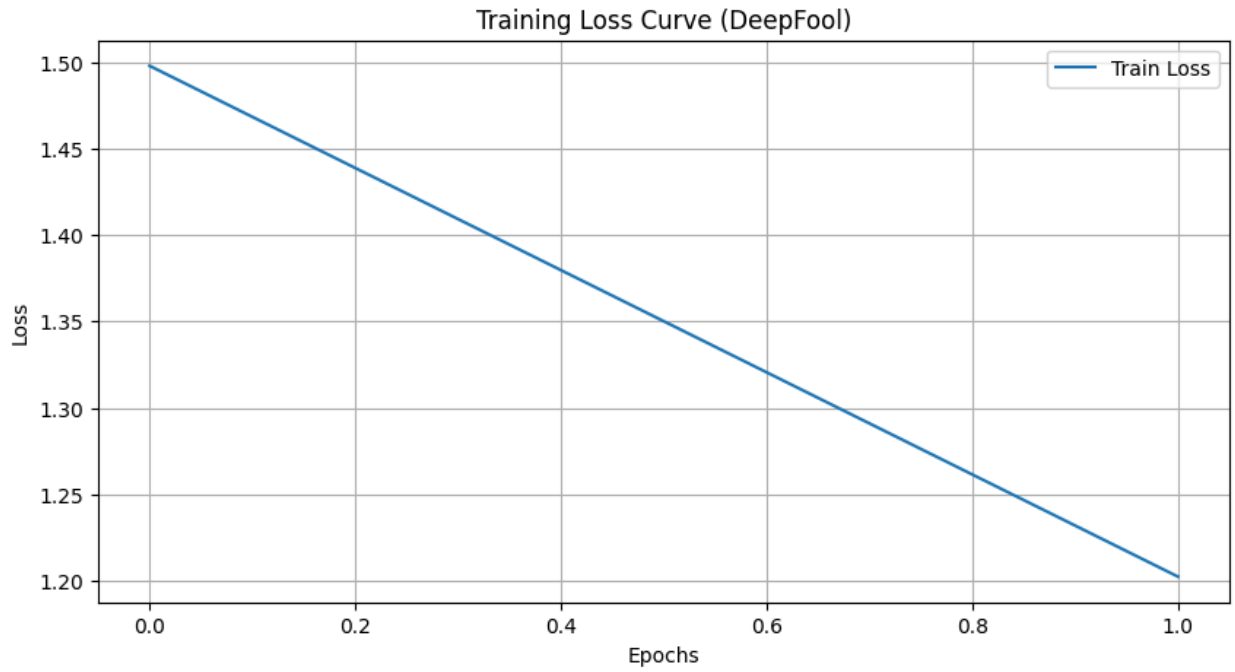
We initialize the DeepFool attack with the following parameters: Number of iterations: 10

Integration into Training Loop:
Within the training loop, we utilize the DeepFool attack to generate adversarial examples from the original input images. The generated adversarial examples are then used to compute the loss and update the model parameters via backpropagation.

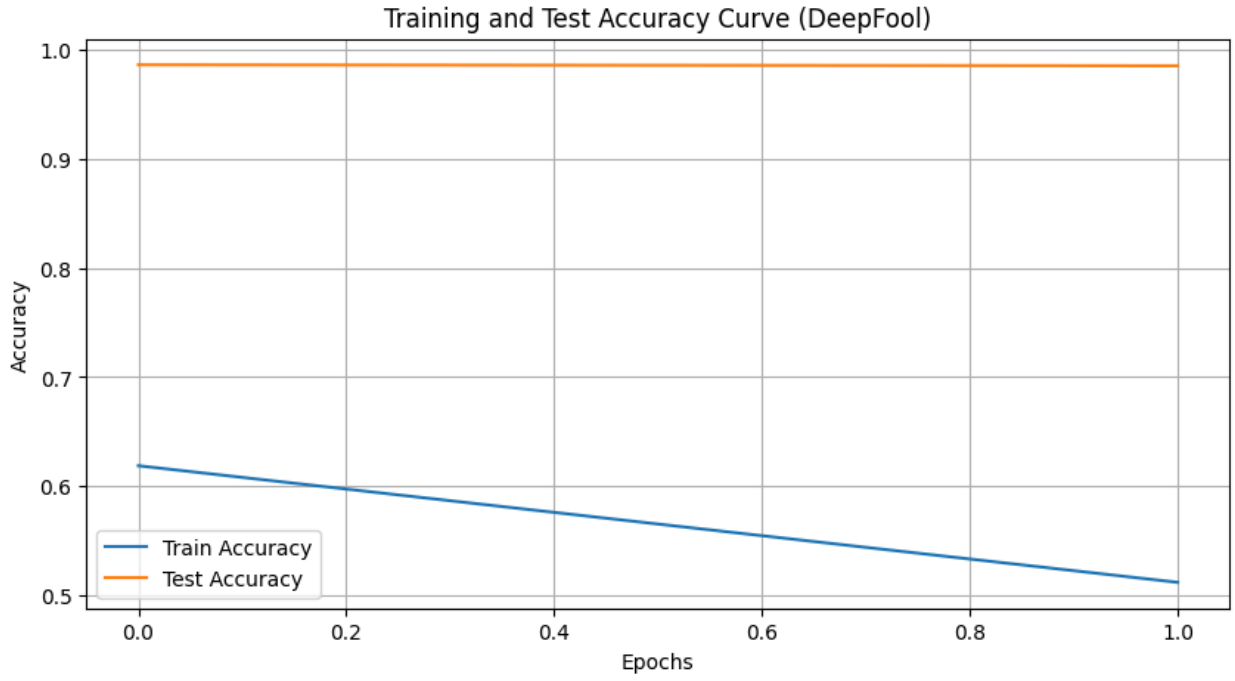
Evaluation of Model Robustness:
We evaluate the performance of our model against adversarial examples generated using the DeepFool attack. By analyzing the model's accuracy on adversarial data, we assess its robustness and resilience against adversarial perturbations.



Results and Analysis:
Training Loss Curve:



Training and Test Accuracy Curve:



Analysis:

The significant drop in training accuracy compared to testing accuracy suggests that the model may struggle to accurately classify adversarial examples generated by the DeepFool attack during training. However, the high testing accuracy indicates that the

model remains robust and maintains high accuracy on adversarial examples during evaluation.
Effectiveness of DeepFool: The relatively high testing accuracy suggests that the DeepFool attack may successfully generate adversarial examples that are challenging for the model to classify correctly during training. Despite this, the model's overall performance on the test set remains strong, indicating that it can generalize well to unseen data.