Trustworthy AI (Monsoon 2024)

Assignment 1 - Report

Saumya Mishra

[0.5cm] February 23, 2025

Contents

1 Introduction

In this project, we were given a model, a dataset, and were asked two implement two different kinds of adversarial attacks on the classifier. The model was a pre-trained SmallAlexNet model, the dataset was the CIFAR-10 dataset, and we were asked to implement a poison frog attack and a deep fool attack. I will get into the details about both shortly, but the Deep Fool ended up being a tougher algorithm to implement than the Poison Frog.

2 Model Exploration

2.1 The Dataset: CIFAR - 10

The dataset we were provided was split into 5 batches, and a 6th test dataset, which had to be unpickled to be used. Here is what a subset of the dataset looked like:

Figure 1: Samples from the various classes

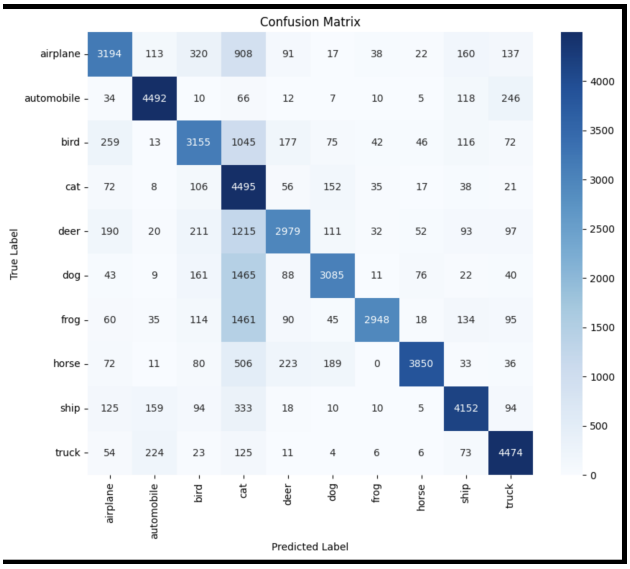

Figure 2: Confusion Matrix for Training set

2.2 Model Performance

The first task we were asked to do was to look at the various performance metrics of the model. Here is the confusion matrix of the model on both the dataset it was trained on, and the provided test dataset.

The model had an accuracy of 73 percent on the training data, and 63 on the test data. An interesting thing to note about the matrix is that there seems to be some bias towards classifiying things as cat: this points to cat being used as an 'other' category.

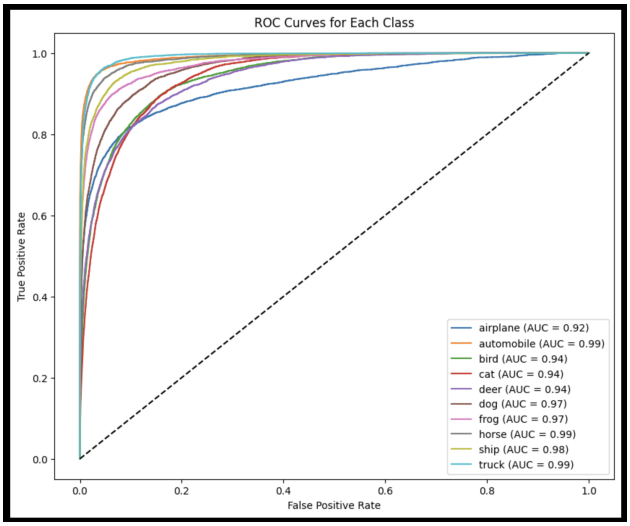

Figure 3: ROC full Dataset

2.3 Comparison with Untrained Model

The untrained model is classifying everything as truck. This is interesting, as unlike a classifier that guesses wildly randomly, the classifier classifies everything as one thing, and eventually spreads its predictions out to the other classes too. The "other" class also changed every time I ran the intialization, so it is possible that the pretrained model having a bias towards "cat" is simply an artefact of the model initialising cat as the other class.

3 Poison Frog

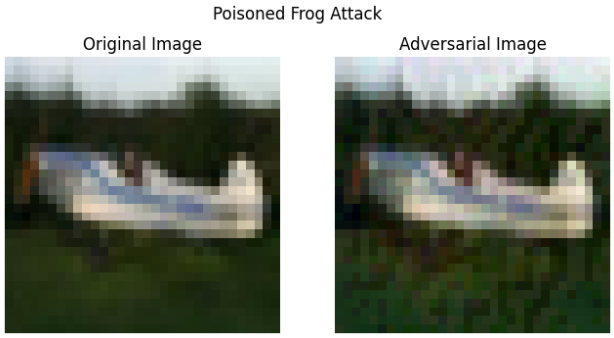

Figure 4: Plane getting misclassified as Ship

3.1 Implementation

The implementation was fairly simple, although it took some tinkering to get the output looking coherent. In my case, I made had image of a plane get misclassified as a ship. The final version of the code ended up looking with classification loss and perceptual loss being treated differently, and then getting added together to form the final loss function.

3.2 Challenges

There was a point at which I was struggling to produce coherent looking perturbations: the images were saturated and incoherent looking. Splitting the loss function into the perceptual similarity and categorical similarity seemed to help, as did changing from the Adam optimiser to standard SGD with momentum.

4 Deep Fool

This was a challenge to implement, as the first few rounds of the program that I wrote seemed to be producing no perturbation at all. After looking at implementations online, I started from scratch and got it working.

4.1 Implementation

This program implements the DeepFool attack, an adversarial attack designed to find the smallest perturbation needed to misclassify an image when given a pretrained classifier. The model is set to evaluation mode, and the true label of the input image is determined. The algorithm then iteratively computes minimal perturbations by analyzing the gradients of the logits for the true class and the top competing classes.
At each step, it determines the optimal direction to shift the classification decision boundary and updates the image accordingly. The perturbation is applied in small steps, ensuring the image remains within valid pixel values. The process continues until the model misclassifies the image or the maximum number of iterations is reached. Finally, the function returns the adversarial image and the total perturbation added. Unlike simpler methods such as FGSM, DeepFool computes an optimal perturbation direction, instead of than applying a fixed step in the gradient direction, making it more effective at minimally altering the image while successfully fooling the classifier.

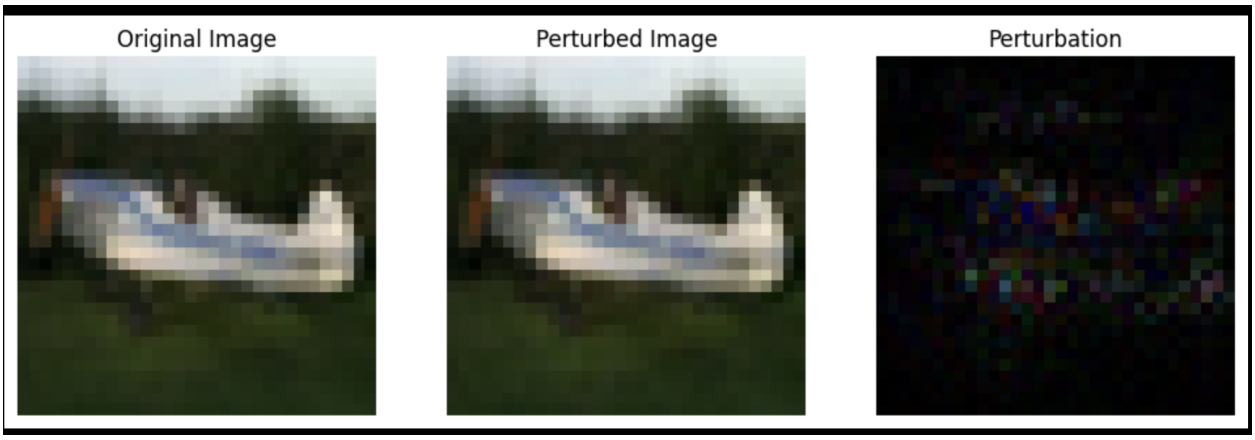

Figure 5: Deep Fool Results