One Pixel Attack for Fooling Deep Neural Networks

Surya Theja Gunti
7002979

Nitish Juttu
7002402

Nobel Jacob Varghese
7002401

Abstract

Deep learning algorithms are essential and popular techniques in terms of achieving state-of-the-art applications for cybersecurity, ranging from intrusion detection, prevention systems, network traffic analysis, behaviour analysis, detection of malware and social engineering detection. But recent research has revealed that the deep neural network techniques are prone to adversarial attacks which can result in misinterpretation of input images or data and this needs to handled in a systematic manner. The authors of the original paper have proposed the idea of one pixel attack for fooling deep neural networks. Our objective is to validate the concept of one-pixel adversarial perturbations based on differential evolution (DE), which is a black box attack that can fool more types of networks due to the inherent features of DE. We explore the 1, 3, and 5 pixel attacks on a standard neural network architecture across multiple datasets to check the efficacy of attack.

1. Introduction

Deep Neural Networks(DNN's) have played a major role in improving image classification accuracies. However it has been proven that these networks can be easily fooled or mislead such that they produce incorrect outputs or misclassify the input.

There are many approaches to attack deep neural networks like Fast Gradient Sign Method (FGSM),PGD, DeepFool, JSMA and Adversarial Patch. Adversarial samples can be generated with various machine learning techniques which creates slightly modified input images that can effectively cause the DNN's to misclassify. The attacks on Deep neural network are classified into white box and black box attacks. In a white box attack the attacker is able to access all the model parameters while in the latter the attacker only has access to the output produced by the model.

Prior works on this direction have utilized techniques which explore excessive modifications to the input image, but these changes are visible to human eyes. However this approach proposes a method of effective attack by perturbing only one pixel with differential evolution, we propose a black-box DNN attack in a scenario where the only information available is the probability labels[1].

We utilize Differential Evolution algorithm as an optimization problem to generate the adversarial samples. It is a stochastic and population-based technique for solving nonlinear optimization problems. Differential Evolution(DE) is a method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. It does so by, optimizing a problem by maintaining a population of candidate solutions and creates new candidate solutions by combining existing ones according to the formula, and keeps which ever candidate solution has the best score or fitness on the optimization problem at hand[2].

2. Background

The vulnerabilities of Deep Neural Networks to black-box attacks can be investigated in good detailed as specified in the paper[3]. Adversaries can easily craft adversarial examples even without any internal knowledge of the target network. The first attack is based on a simple idea of adding perturbation to a randomly selected single pixel and improve effectiveness of the attack by constructing smaller set of pixels to perform perturbation by applying greedy local-search technique and without emphasising on the usage of gradient information, accurately leading to misclassification by neural networks[3]. Another paper proposes a technique to train a DNN from knowledge learned about the existing network to make the existing network robust to attacks. It introduce a defensive mechanism called defensive distillation to reduce the effectiveness of adversarial samples on DNNs and analytically investigates the generalizability and robustness properties by using distillation approach for the training of DNNs and the study shows that defensive distillation reduces effectiveness of sample creation from 95 percent to less than 0.5 percent on a studied deep neural networks[4] Furthermore, the resulting network have low generalisation error and are less prone to adversarial attacks.Keeping in mind about not utilising classical optimizing methods such as gradient descent and quasi-newton methods which requires the optimising problem to be differentiable and rather use other novel techniques such as evolutionary based algorithms (Differential evolution),we tried to incorporate and implement the idea as proposed by the base paper[1] and implement attack on various kinds of architectures and datasets.

3. Methodology

We now look into the problem description and explain about the datasets, methodology, experimental setup and step by step analysis of our work.

3.1. Datasets

The 3 different datasets in which we conducted our experiments are **Flowers Recognition dataset [5]**, **Fashion MNISNT[6]** and **Chessman image data-set [7]**.

Flower Recognition dataset consist of images of 4242 flowers which are dived into 5 classes- Chamomile, Tulip, Rose, Sunflower, Dandelion. Each class consists of approximately 500 images with 320x240 pixel resolution.

Fashion MNIST consist of approximately 70,000 greyscale images with 60,000 train images and 10,000 test images, each of size 28x28 classified into 10 labels(T-shirt, Trouser etc).

Chessman image dataset was initially used to check how well the idea of differential-evolution could be applied, as the dataset is relatively small. The conclusion's from the observations of Chessman image data-set was taken into consideration while scaling for the other larger datasets.

The Chess and Flower datasets are downloaded from Kaggle and FashionMNIST from Pytorch Datasets. The Chess dataset images are resized to **64x64** and the images of Flower dataset are resized to **32x32** due to computational limitations. We also made sure that the test accuracies of the models trained are good enough even after downscaling the images.

3.2. Experimental Setup

ResNet-18, a CNN architecture which is 18 layers deep was chosen as the DNN model for classification of input images and implementing the proposed attack scenario. We chose **ResNet** because it introduces identity shortcut connections that can skip one or more layers and the gradients flow directly through the skip connections backwards from later layers to initial filters, thereby helping to avoid vanishing gradient problem.

Cross-entropy loss and **Adam** optimizer where set as model parameters for training the network because cross entropy loss function works well for classification task and adaptive moment optimizer is efficient when working with large problem involving lot of data and parameters.

We encode perturbation in an array consisting of candidate solution which is evolved by differential evolution optimising method. The candidate solution consists fixed number of perturbations and each perturbation is tuple consisting 5 coordinates namely x, y and RGB values of perturbation. Initial set population is 50. Table 1 refers to the parameters used in implementing the algorithm. For each iteration, new children will be generated based on the formula mentioned below. Parents and Children continuously compete against each other and whichever is better stays in the population for next iterations. We use probabilities of class labels to determine if parent is better or the child.

Population Size	50
Max Iterations	100
Mutation	0.5
Recombination	0.7

Table 1. Parameters for Differential Evolution

$$x_i(g+1) = x_{r1}(g) + F(x_{r2}(g) - x_{r3}(g)),$$

where r_1, r_2, r_3 are random numbers, x_i is candidate solution element, g refers to index of generation and F refers to parameter scale value set to 0.5 in the experiment.

The different experimental settings chosen to conduct the attack and arrive at the results are **1-pixel, 3-pixel and 5-pixel attack**. All the **3 datasets(Chess, Flower, and Fashion MNIST)** were checked for each all the 3 different settings. Hence we have results for about 9 different experiments, which can provide conclusive evidence about the efficiency of the proposed method.

All the attacks are **non targeted** which implies the attack on an image is stopped if the network predicts a wrong label. Targeted attacks are also possible where the adversarial image is generated to target a specific class. Targeted attacks need much more computation effort (which is directly proportional to the number of classes) compared to Non Targeted attacks. This is the reason, we could only try non targeted attacks.

3.3. Defending the Attack

For Defending the model against these attacks, we retrained the model with the generated adversarial images. But we could not identify any noticeable difference in attack success rates. We could identify two possible reasons. 1) Since we are retraining the model with adversarial images with just few pixel changes, we might need a lot more adversarial images for the model to identify the subtle differences. 2) As the Differential Evolution algorithm initializes population randomly, the adversarial images generated would highly depend on initial population and the generated images would be different each time we run the algorithm for the same set of original images. As a result there is no common pattern in how the attack is performed and adversarial images are generated. These could be the reasons why finding a defense technique was difficult. Re-

Dataset	Accuracy	Network	Pixels	Success %
Flowers	0.86	ResNet18	1 Pixel	0.43
			3 Pixel	0.63
			5 Pixel	0.72
MNIST	0.9	ResNet18	1 Pixel	0.35
			3 Pixel	0.56
			5 Pixel	0.63
Chess	0.82	ResNet18	1 Pixel	0.33
			3 Pixel	0.43
			5 Pixel	0.49

Table 2. Attack success rates on various datasets

cently a paper on “Do not get fooled: Defense against the one-pixel attack to protect IoT-enabled Deep Learning systems”[8] was published in November 2021. The authors proposed a technique called Accelerated Proximal Gradient approach to tackle One Pixel Attacks. Implementing such a technique was not feasible for us due to time constraints.

4. Evaluation and Results

The Figures describe the 1, 3 and 5 pixel attacks on various datasets. The Differential Evolution algorithm is able to fool the network with just a single pixel with a decent attack success rate on all the three datasets. The success rate improves by a lot when 3 and 5 pixel attacks are used. From adversarial images generated from FashionMNIST, we can observe that Model with 100% confidence can be fooled with just a single pixel. Similar results can also be observed on the other two datasets also. From the Table, we can see models trained on all the datasets are able to achieve a test accuracy of at least 80%. 1 pixel attack had a success rate of at least 30% on all the datasets with 43% being the highest on the Flowers dataset. Chess dataset had the least success rate in terms of 3 Pixel attack with only 43% followed by FashionMNIST and Flowers with 56% and 63% respectively. For a 5 Pixel attack, Flowers dataset had the highest success rate of 73% followed by Fashion and Chess datasets with 63% and 49% respectively. Overall Flowers dataset was easier to attack and Chess dataset is difficult to attack compared to the other datasets. The Differential Evolution algorithm seems to be working irrespective of datasets and network architectures. The only drawback noticed was the computation times the algorithm takes especially when the image resolution is higher(256x256). This is the reason we had to resize the images to a smaller resolution. This increase in computation time can be justified as the image size increases, the algorithm needs more iterations to find pixels that can fool the network. Finally we can conclude that Single Pixel Attack using Differential Evolution algorithm is a very successful technique to fool networks irrespective of datasets.

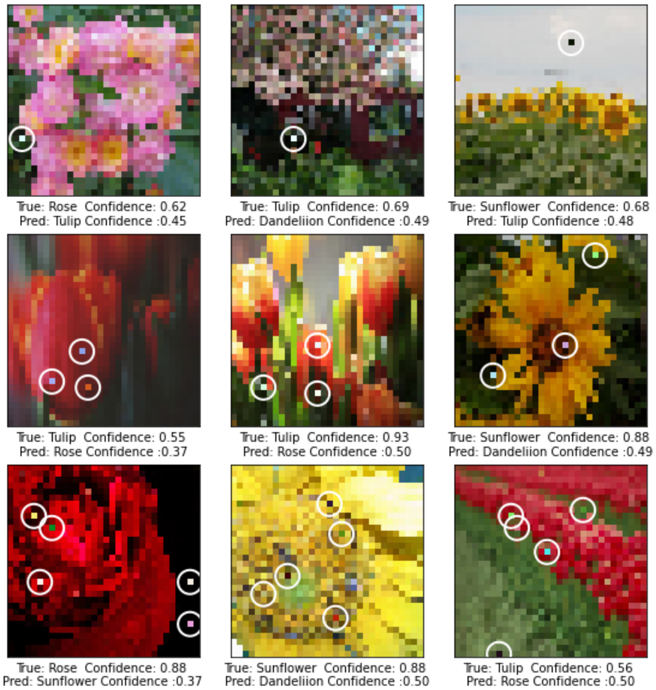
True: Rose Confidence: 0.62
Pred: Tulip Confidence :0.45
True: Tulip Confidence: 0.69
Pred: Dandeliion Confidence :0.49
True: Sunflower Confidence: 0.68
Pred: Tulip Confidence :0.48
True: Tulip Confidence: 0.55
Pred: Rose Confidence :0.37
True: Tulip Confidence: 0.93
Pred: Rose Confidence :0.50
True: Sunflower Confidence: 0.88
Pred: Dandeliion Confidence :0.49
True: Rose Confidence: 0.88
Pred: Sunflower Confidence :0.37
True: Sunflower Confidence: 0.88
Pred: Dandeliion Confidence :0.50
True: Tulip Confidence: 0.56
Pred: Rose Confidence :0.50

Figure 1. 1, 3, 5 Pixel Attacks on Flowers dataset

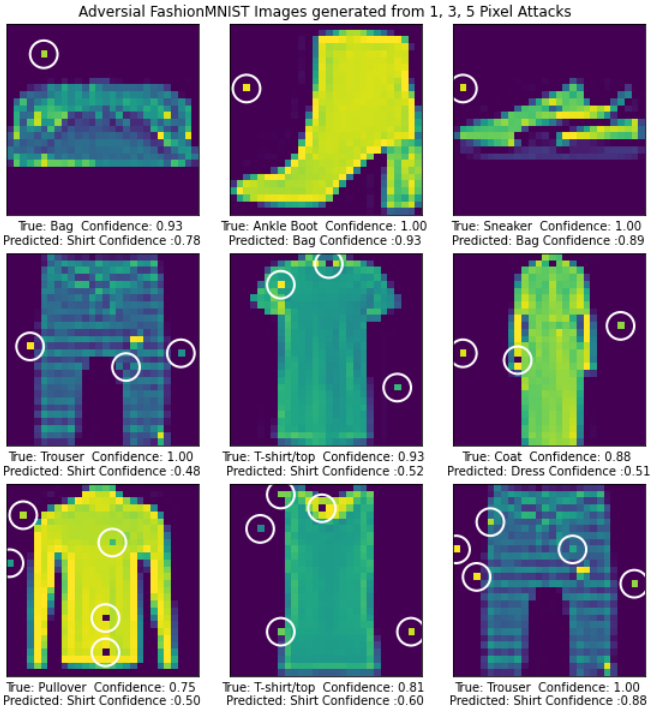
Adversial FashionMNIST Images generated from 1, 3, 5 Pixel Attacks
True: Bag Confidence: 0.93
Predicted: Shirt Confidence :0.78
True: Ankle Boot Confidence: 1.00
Predicted: Bag Confidence :0.93
True: Sneaker Confidence: 1.00
Predicted: Bag Confidence :0.89
True: Trouser Confidence: 1.00
Predicted: Shirt Confidence :0.48
True: T-shirt/top Confidence: 0.93
Predicted: Shirt Confidence :0.52
True: Coat Confidence: 0.88
Predicted: Dress Confidence :0.51
True: Pullover Confidence: 0.75
Predicted: Shirt Confidence :0.50
True: T-shirt/top Confidence: 0.81
Predicted: Shirt Confidence :0.60
True: Trouser Confidence: 1.00
Predicted: Shirt Confidence :0.88

Figure 2. 1, 3, 5 Pixel Attacks on FashionMNIST dataset

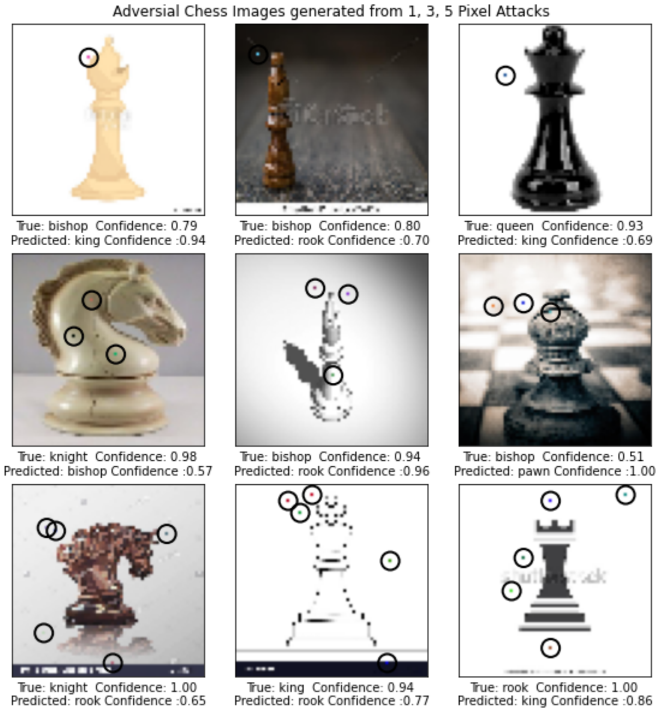
Adversial Chess Images generated from 1, 3, 5 Pixel Attacks
True: bishop Confidence: 0.79
Predicted: king Confidence :0.94
True: bishop Confidence: 0.80
Predicted: rook Confidence :0.70
True: queen Confidence: 0.93
Predicted: king Confidence :0.69
True: knight Confidence: 0.98
Predicted: bishop Confidence :0.57
True: bishop Confidence: 0.94
Predicted: rook Confidence :0.96
True: bishop Confidence: 0.51
Predicted: pawn Confidence :1.00
True: knight Confidence: 1.00
Predicted: rook Confidence :0.65
True: king Confidence: 0.94
Predicted: rook Confidence :0.77
True: rook Confidence: 1.00
Predicted: king Confidence :0.86

Figure 3. 1, 3, 5 Pixel Attacks on Chess dataset

References

[1] Jiawei Su , Danilo Vasconcellos Vargas and Kouichi Sakurai. One Pixel Attack for Fooling Deep Neural Networks

[2]https://medium.datadriveninvestor.com/why-you-should-be-using-differential-evolution-for-your-optimization-problems-b3b2ed622c4a

[3] N. Narodytska and S. Kasiviswanathan. Simple black-box adversarial attacks on deep neural networks. In 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp.1310–1318. IEEE, 2017

[4] N. Papernot, P. McDaniel, X. Wu, S. Jha, A. Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In Proceedings of IEEE Symposium on Security and Privacy (SP), pp.1701– 1708.

[5]https://www.kaggle.com/alxmamaev/flowers-recognition

[6] https://github.com/zalandoresearch/fashion-mnist

[7] https://www.kaggle.com/niteshfre/chessman-image-dataset

[8] Muhammad Akbar Husnooa and Adnan Anwar. Do Not Get Fooled: Defense Against the One-Pixel Attack to Protect IoT-Enabled Deep Learning Systems.