HAL open science

Pixel Based Adversarial Attacks on Convolutional Neural Network Models

Kavitha Srinivasan, Priyadarshini Jello Raveendran, Varun Suresh, Nithya Rathna Anna Sundaram

To cite this version:

Kavitha Srinivasan, Priyadarshini Jello Raveendran, Varun Suresh, Nithya Rathna Anna Sundaram. Pixel Based Adversarial Attacks on Convolutional Neural Network Models. 4th International Conference on Computational Intelligence in Data Science (ICCIDS), Mar 2021, Chennai, India. pp.141-155, 10.1007/978-3-030-92600-7_14 . hal-03772929

HAL Id: hal-03772929

https://inria.hal.science/hal-03772929

Submitted on 8 Sep 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

This document is the original author manuscript of a paper submitted to an IFIP conference proceedings or other IFIP publication by Springer Nature. As such, there may be some differences in the official published version of the paper. Such differences, if any, are usually due to reformatting during preparation for publication or minor corrections made by the author(s) during final proofreading of the publication manuscript.

Pixel based Adversarial Attacks on Convolutional Neural Network Models

Kavitha S[1][0000-0003-3439-2383], Priyadarshini J R[2], Varun Suresh[3], Nithya Rathna A[4]
Department of Computer Science and Engineering,
Sri Sivasubramaniya Nadar College of Engineering, Kalavakkam – 603 110, INDIA.
[1]kavithas@ssn.edu.in, [2, 3, 4]{priyadarshini15074, varun15122, nithyarathna15067}@cse.ssn.edu.in
*Corresponding Author: kavithas@ssn.edu.in

Abstract. Deep Neural Networks (DNN) has found their applications in the real time, for example, facial recognition for security in ATMs and self-driving cars. A major security threat to DNN is through adversarial attacks. An adversarial sample is an image that has been changed in such a way that it is imperceptible to human eye but causes the image to be misclassified by a Convolutional Neural Networks (CNN). The objective of this research work is to devise pixel based algorithms for adversarial attacks on images. For validating the algorithms, untargeted attack is performed on MNIST and CIFAR-10 dataset using techniques such as edge detection, Gradient weighted Class Activation Mapping (GRAD-CAM) and noise addition whereas targeted attack is performed on MNIST dataset using Saliency maps. These adversarial images thus generated are then passed to a CNN model and the misclassification results are analyzed. From the analysis, it has been inferred that it is easier to fool CNNs using untargeted attacks than the targeted attacks. Also, grayscale images (MNIST) are preferred to generate robust adversarial examples compared to colored images (CIFAR-10.
Keywords: Deep Neural Networks, Adversarial attacks, Convolutional Neural Network Models, Gradient weighted Class Activation Mapping, Edge detection, Noise addition, Saliency maps.

1 Introduction

Deep Learning is finding its use in many applications nowadays. Unsolved problems of the Machine Learning and Artificial Intelligence are being solved by Deep Learning techniques. As a result, it is currently being used in a variety of real-world application such as human face recognition, image analysis and self-driving cars. These applications lead to large volume of information and computations. Face recognition tool or software, efficient enough to be used in ATMs and for unlocking phones, have been developed. Some of the medical systems developed for analysis and disease detection performs better than human experts of that specific field. Self driving cars are a distinguishable application where human drivers are no longer required.

The latest research by Google Brain has shown that any machine learning classifier could be modified to result a wrong outcome. The attackers can showcase their skills by changing the result as per their requirement. This scenario affected the real time systems in banks, ATMs, facial recognition on laptops and self-driving cars, which are developed by artificial intelligence and many of them are decisive for a secured life.

For example, if an attacker aims to create adversarial attack on direction signs, then the self-driving car may interpret in a wrong way and can take unwanted actions. This might result in a major road accident causing severe damage. An adversarial attack involves slightly changing an image in such a way that the modifications are indistinguishable to the human eyes. The changed image is called as adversarial image, which results a misclassification when submitted to the classifier. The two types of adversarial attacks are targeted attack and untargeted attack. In a targeted attack, the attacker dissimulates to get the image classified as a specific target class, which is different from the class of the original image. The objective of untargeted attacks is to make the model to return a wrong prediction as outcome using adversarial image.

This research work is proposed to create untargeted attacks using techniques such as edge detection, GRAD-CAM and noise application and targeted attacks using Saliency maps. Edge detection involves detecting the edge pixels using Canny library and altering their intensity values. Grad-CAM takes the gradients from the final convolutional layer to generate a map that helps determine the essential regions in an image. Saliency maps are generated by computing the gradient of output image with respect to input image. These algorithms are an evidence to understand how the output value changes with respect to a small modification in input image pixels. These adversarial images thus generated are then passed to a CNN model and the results are compared and analyzed.

2 Related Work

Adversarial samples are inputs to machine learning models that an attacker has deliberately designed a slight perturbation to cause the model to misclassify an image. The fascinating properties of neural networks perturbations are applied to an image that caused the network to misclassify it [Szegedy, Christian et al., 2014]. These perturbations were found by maximizing the network's prediction error. Attacks and defenses for deep learning model are defined as adversarial examples to be invisible to human eyes that could easily fool deep neural networks is elaborated in [Yuan, Xiaoyong et al., 2019]. The paper summarized recent discoveries on adversarial examples, methods for generating adversarial examples and also proposed categorization of those methods.

One Pixel attack [Su, Jiawei et al., 2019] is carried out by changing only one pixel with differential evolution in a scenario where the only information available is the probability labels. The attack was tested on pre-trained models on CIFAR-10 and Alexnet mode trained on Imagenet. Zeroth Order Optimization [Chen, Pin-Yu et al., 2017] based attacks directly estimate the gradients of the targeted DNN for generating adversarial examples.

The one-step gradient-based approach namely FGSM, finds an adversarial example by maximizing the loss function [Goodfellow, Ian J. et al., 2015]. Iterative methods run FGSM multiple times with a small step size α and named as I-FGSM [Kurakin, Alexey et al., 2017]. MI-FGSM is a method for speeding up gradient descent algorithms by accumulating a velocity vector in the direction of loss function across iterations [Dong, Yinpeng et al., 2018]. These iterative based attacks were trained on Imagenet and tested on Inception network.

Carlini and Wagner attacks are adapted to three distance metrics L_0, L_2 and L_∞ [Carlini, Nicholas and David A. Wagner., 2017]. The target is to find δ that minimizes D(x, x+δ), where x is the given image. That is, to find some small change δ that can be made to an image x that will change its classification, however the result would still be a valid image. In Table 1, the summary of existing approaches is discussed.

In Adversarial Attacks and Defenses: A Survey by [Chakraborty, A., Alam M. et al., 2018], explained about different threat models, existing attack algorithms and its countermeasures. The paper compares the efficiency and limitations of the various attacks and countermeasures that were selected for examination.

In Practical Adversarial Attack against Object Detector by [Zhao, Y.S., Zhu, H. et al., 2018], discussed two different attack algorithms against object detectors in realistic situations such as autonomous driving cars are provided. YOLO V3 was the object detector upon which the attacks were tested on by changing various factors such as distance, illumination, angles etc.

The research papers mentioned have created an algorithm that makes subtle changes in the images causing neural networks to misclassify the adversarial images. This establishes the fact that neural networks are vulnerable to adversarial changes. The goal of this project is to create similar yet simpler attack algorithms using pixel modification techniques. For instance, by causing changes to the background pixels without affecting the pixels that affect the shape of the number, misclassifications were caused. This shows that CNNs do not view images as humans do and are vulnerable to adversarial attacks. Attack algorithms were hence created, applied on images from MNIST and CIFAR-10 datasets.

TABLE 1. SUMMARY OF EXISTING ATTACK METHODS

Method	Type	Dataset	Model Used
One Pixel Attack	Untargeted	CIFAR-10 ImageNet	VGG AlexNet
Zeroth Order Optimisation	Targeted Untargeted	MNIST CIFAR-10 ImageNet	C&W Framework C&W Framework Inception v3
Fast Gradient Sign Method (FGSM)	Targeted	MNIST ImageNet	Softmax Classifier Inception v3
Iterative-FGSM	Targeted	ImageNet	Inception v3
Momentum Iterative – FGSM	Targeted	ImageNet	Inception v3
C&W Attack	Targeted	MNIST CIFAR-10 ImageNet	C&W Framework C&W Framework AlexNet

3 Proposed System

The proposed system aims to create untargeted and targeted attacks using techniques such as edge detection, GRAD-CAM, noise addition and saliency maps. The proposed algorithms (adversarial image creation) for two datasets are given in Figure 1.

Input: 28x28 black and white MNIST
32x32 colored CIFAR-10 image
Output: Attacked image

1. *Untargeted Attacks on MNIST*
   a. Modification of background pixels o
   b. Modification of edge pixels
   c. Modification of edge and background pixels
   d. Gaussian noise addition and modification of edge pixels
2. *Untargeted Attacks on CIFAR-10*
   a. Modification of edge pixels
   b. Heatmap generation and modification of edge pixels
3. *Targeted Attacks on MNIST*
   a. Altering effective pixels with fixed values
   b. Altering effective pixels using AES
   c. Altering maximum occurring pixels

Figure 1. Workflow of proposed system

3.1 Untargeted Attacks

An untargeted attack aims to affect the image such that the model misclassifies the image. The untargeted attack is performed on MNIST and CIFAR-10 dataset using various techniques as mentioned in Figure 1.

The methods used to perform untargeted attacks are explained below:

Canny edge detection algorithm: The canny edge detector is a multistage edge detection algorithm. It has 4 stages such as preprocessing, calculating gradients, non-maximum suppression and thresholding with hysteresis. The two key parameters of the algorithm are - an upper threshold and a lower threshold. The upper threshold is used to mark edges that are definitely edges. The lower threshold is to find faint pixels that are actually a part of an edge.

Heat map: A heat map is a visual representation of data where the individual values contained in a matrix are represented as colors. It helps in decision making by highlighting areas of greater attention. Areas of high activity are represented using bright colors while the areas of low activity are represented using darker colors.

AES algorithm: The Advanced Encryption Standard (AES) is a symmetric block cipher. It is the most popular and widely used encryption algorithm because it is six times faster and has a smaller key size than Data Encryption Standard (DES). It contains four phases such as add round key, substitute bytes, shift rows and mix columns to perform encryption.

3.1.1 MNIST

In the MNIST dataset, there are two sets of pixels. One set contributes to the background while the other contributes to the shape of the number. The simplest change that could be caused to one such image is altering the intensity of the background pixels.

After the attack caused misclassification, the next step was to verify there would be a similar effect upon altering the pixels that affect the shape of the number. Thus, the pixels contributing the edge of the numbers (along with their surrounding pixels) were detected and their intensity was altered using AES algorithm. As the misclassification did not match up to the background change, the two methods were combined.

Modification of background pixels: The background color of the image is modified by adding a constant value.

Algorithm 1*:* Attack algorithm for modifying background pixels

Input: 2D black and white image of handwritten digits from 0 to 9 of dimensions 28x28 (MNIST)

Output: Attacked image

```
function BACKGROUND PIXEL MODIFICATION(Image)
    Identify pixels with intensity zero
    for all pixels i in identified pixels do
        Add a constant value to the intensity
    end for
    return Image
end function
```

Edge detection: Edge pixels are identified using Canny library and the intensity of these pixels are modified using AES algorithm.

Algorithm 2**:** Attack algorithm for modifying edge pixels

Input: 2D black and white image of handwritten digits from 0 to 9 of dimensions 28x28 (MNIST)

Output: Attacked image

```
function EDGE PIXEL MODIFICATION(Image)
    Identify the edge pixels using Canny library
    Add the pixels to the left, right, top and bottom of identified pixel to
    the list
    for all pixels i in identified pixels do
        Apply AES algorithm on the intensity value
        Perform mod 256 on the obtained value
        Set this value as the new intensity of the pixel i
    end for
    return Image
end function
```

Combination of edge detection and background modification: Intensity of edge pixels and background pixels are modified using AES algorithm in which two different nonces are set for edge pixels and background pixels.

Algorithm 3: Attack algorithm for modifying edge and background pixels

Input: 2D black and white image of handwritten digits from 0 to 9 of dimensions 28x28 (MNIST)

Output: Attacked image

```
function EDGE & BACKGROUND MODIFICATION(Image)
    Identify pixels in Image with intensity zero
    Identify edge pixels in Image using Canny Algorithm
    for all pixels i in identified pixels do
        If pixel i is an edge pixel set nonce as '!#!9'
        If pixel i is a background pixel set nonce as '!#!#'
        Apply AES algorithm on the intensity value
        Perform mod 256
        Set this value as the new intensity of the pixel i
    end for
    return Image
end function
```

Addition of noise and edge detection: Gaussian noise is applied to the image and then AES algorithm is applied to modify the intensity of those edge pixels.

***Algorithm 4*:** Attack algorithm for addition of gaussian noise and modifying edge pixels

Input: 2D black and white image of handwritten digits from 0 to 9 of dimensions 28x28 (MNIST)

Output: Attacked image

```
function NOISE ADDITION(Image)
    Add Gaussian noise to the image using random noise function
    Determine the edge pixels using Canny library
    Add the pixels to the bottom-left, top-left, bottom-right and top-
    right of identified pixel to the list
    for all pixels i in identified pixels do
        Apply AES algorithm on the intensity value with nonce
    '!#!%'
        Perform mod 256 on the obtained value
        Set this value as the new intensity of the pixel i
    end for
    return Image
end function
```

3.1.2 CIFAR-10

The images from CIFAR-10 have three channels, red-blue-green. As colored images do not have differentiated set of pixels, it did not make sense to apply background modification on this dataset. Similarly noise application could cause random spots of color and hence was not tested on CIFAR-10.

While edge pixels contribute to the shape of the number in MNIST, they contribute to the object or some object in the background in CIFAR-10. For instance, an image of a horse in front of a mountain causes the edge of the mountain to be de-

tected as well. Thus, the effect of the attack is less prominent in CIFAR-10. The pixel intensities have to be altered for each channel. Finally, we determined the important regions in an image by using Gradient-weighted Class Activation Mapping (Grad-CAM). Based on the heatmap generated from the image, the pixels contributing to the important regions were modified.

Edge detection: Edge pixels are identified using Canny library and the intensity of these pixels are modified using AES algorithm.

***Algorithm 5*:** Attack algorithm for modifying edge pixels

Input: 2D color image of dimensions 32x32 (CIFAR-10)
Output: Attacked image

```
function EDGE PIXEL MODIFICATION(Image)
    Identify edge pixels in Image using Canny Algorithm
    for all pixels i in identified pixels do
        Set nonce to 'wxyy' for the blue channel
        Set nonce to 'wxyz' for the green channel
        Set nonce to 'wxyw' for the red channel
        Apply AES algorithm on the channel intensities and perform mod 256
        Set this as the new intensity of pixel i
    end for
    return Image
end function
```

Combination of heat map generation and edge detection: Identifying significant pixels using a heat map and modifying them using AES algorithm. If heatmap is not generated, edge pixels are detected and AES is applied to those images.

***Algorithm 6*:** Attack algorithm for modifying significant pixels using a heatmap and modifying edge pixels for others

Input: 2D color image of dimensions 32x32 (CIFAR-10)
Output: Attacked image

```
1:  function HEATMAP(Image)
2:      Generate heatmap of the given image
3:      If heatmap is generated, goto step 10
4:      If heatmap is not generated, determine edge pixels using Canny library
5:      for all pixels i in identified pixels do
6:          Apply AES algorithm on the intensity value with nonce 'z123'
7:          Set this value as the new intensity of the pixel i
8:      end for
9:      goto step 15
10:     Identify the pixels whose channel with maximum intensity is red
11:     for all pixels i in identified pixels do
12:         Apply AES algorithm on the intensity value with nonce 'z123'
13:         Set this value as the new intensity of the pixel i
14:     end for
15:     return Image
```

```
end function
```

3.2 Targeted Attacks

In a targeted attack, the attacker aims to get the image classified as a specific target class that is different from the class of the original image. The attack has been applied on a subset of the MNIST dataset. The model is trained on the images of numbers 0, 1, 2 and 7. The attacks applied on the images are Altering effective pixels using fixed values and Altering pixels with maximum occurrence.

The untargeted attacks mentioned in the previous section caused changes to pixel intensities and caused misclassification. This section aims at causing targeted misclassification. Thus, it is important to determine the pixels considered important for a given class. To determine significant pixels for a given class, a saliency map is generated.

The first attack was to reduce the intensity of the pixels that contribute to the current class while increasing the intensity of the pixels contributing to the target class. Though this caused effective misclassification, the changes were visible to a human eye. To prevent this, the pixels were grouped based on their saliency values. The set with the highest occurrence was alone modified. Though the number of misclassification was reduced, the changes caused to the image were less prominent.

3.2.1 Altering effective pixels using fixed values

The intensity of effective pixels determined from saliency maps are changed using fixed values, causing the image to be misclassified as belonging to the class chosen by the attacker.

Algorithm 7: Attack algorithm for altering effective pixels with fixed values

Input: 2D black and white image of a handwritten number of 0, 1, 2 or 7 of dimensions 28x28 (MNIST), Target class

Output: Attacked Image

```
function ALTERING FIXED(Image,Target)
    Change the activation function of the Models last from softmax to linear
    Pick an image T from Target class
    Create a saliency map for both Image and T
    Change the intensity to 0 in Image, for positions in Image with the saliency greater than or equal to 0.1
    Change the intensity to 255 in the Image, for positions in T with the saliency greater than or equal to 0.2
    return Image
end function
```

3.2.2 Altering pixels with maximum occurrence

The intensity of maximum occurring pixels determined from saliency maps are changed. This causes the image to be misclassified as belonging to the class chosen by the attacker.

***Algorithm 8*:** Attack algorithm for altering maximum occurring values

Input: 2D black and white image of a handwritten number of 0, 1, 2 or 7 of dimensions 28x28 (MNIST), Target class

Output: Attacked Image

```
function ALTERING MAXIMUM(Image,Target)
        Change the activation function of the Models last from softmax to linear
        Pick an image T from Target class
        Create a saliency map for both Image and T
        Round of values in the saliency maps to a single decimal place
        Find maximum occurring element M in the saliency map of Image
        Change the intensity to 0 in Image, for positions in Image with the saliency M
        Find maximum occurring element M1 in the saliency map of T
        Change the intensity to 175 in the Image, for positions in T with the saliency M1
        return Image
end function
```

4 Results and Performance Analysis

Different layers and activation functions are used in CNN model training for MNIST and CIFAR-10 dataset. CNN for MNIST dataset has five set of convolutional layers includes the filter sizes as 64, 128, 256, 512 and 512 whereas, for CIFAR-10 dataset, it has two layers 32, 64 respectively. The images are convolved with the filters at each convolutional layer, followed by fully connected layers and a softmax layer. Rectified Linear Unit, ReLU is the activation function used at the convolutional layers [9]. The techniques applied on MNIST dataset to perform untargeted attacks are: (i) modification of background pixels, (ii) edge detection, (iii) combination of edge detection and background modification and, (iv) addition of noise and edge detection. Similarly for CIFAR-10 dataset, edge detection and combination of heat map generation with edge detection were applied. The size and accuracies obtained from both datasets are mentioned in Table 2.

TABLE 2. SUMMARY OF ACCURACY OBTAINED FOR THE CNN MODEL TRAINED OVER VARIOUS DATASETS

Dataset	**Training Set**		**Validation Set**	
	Size	**Accuracy (%)**	**Size**	**Accuracy (%)**
MNIST	55000	99.82	5000	99.30
CIFAR-10	45000	97.23	5000	79.32

In Figure 2, untargeted misclassifications caused by the techniques applied on MNIST dataset are shown. It shows the original image with its class against the adversarial image with the predicted class. The various untargeted attacks described

were applied on the test set of MNIST and CIFAR-10 datasets, where the test sets contain 10000 images each. In Table 3, the summary of various techniques applied and its percentage of misclassification caused is tabulated. In addition, the performance of untargeted attacks is graphically represented in Figure 3.

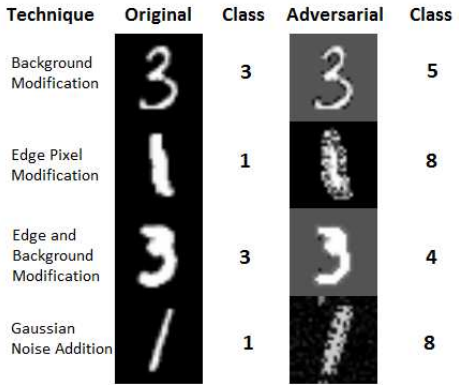

Figure 2. Untargeted attacks – MNIST dataset

TABLE 3. SUMMARY OF UNTARGETED ATTACKS

Name of the technique	Dataset used	Misclassification (%)
Background Modification	MNIST	68
Edge Detection	MNIST	37
Combination of Edge Detection and Background Modification	MNIST	89
Noise Addition and Edge Detection	MNIST	32
Edge Detection	CIFAR-10	47
Combination of Heatmap and Edge Detection	CIFAR-10	43

In case of MNIST dataset, highest numbers of misclassifications were caused by the combination of Edge Detection and Background Modification. Though, Background Modification provides a good amount of misclassification, Edge Detection does not match it. This could be due to the difference in the number of pixels altered. There is also an added amount of uncertainty due to AES algorithm. Noise addition caused the least amount of misclassification proving that DNNs are not vulnerable to random changes in an image.

The results on CIFAR-10 shows that it is harder to generate robust adversarial images when compared to MNIST because of the three channels involved in colored images. Edge Detection fares better when compared to combination of Heatmap and Edge Detection.

Targeted attacks have been applied on a subset of the MNIST dataset. The model is trained on the images of numbers 0, 1, 2 and 7. The division of the dataset is shown in Table 4. Various techniques used are altering effective pixels using fixed values, altering effective pixels using AES and altering pixels with maximum occurrence. Figures 4 and 5, shows the targeted misclassifications caused on the subset of

MNIST dataset, that is images with labels 0, 1, 2 and 7. For maximum occurring pixels, the images could not be modified to be misclassified as 1. The images belonging to class 3 (number 7) are not misclassified as class 0 or class 2.

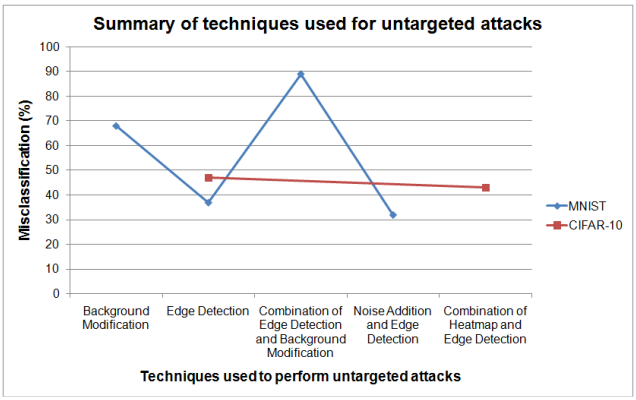

Figure 3. Techniques used Vs Misclassification (%) - Untargeted attacks

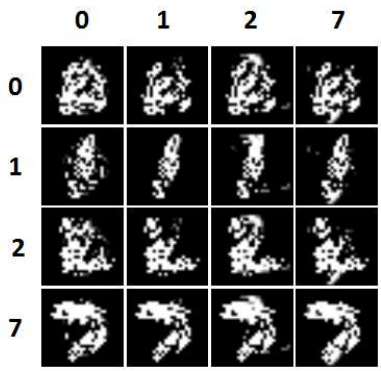

Figure 4. Altering pixels with fixed values on MNIST dataset

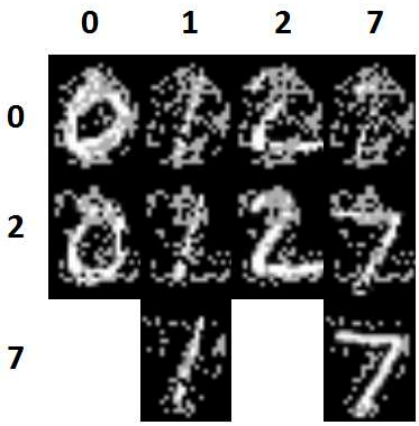

Figure 5. Altering pixels with maximum occurrence on MNIST dataset

In Table 5, summary of the targeted techniques applied and its percentage of misclassification caused are tabulated. Altering pixels with fixed values fares better than altering pixels with maximum occurrences in terms of misclassification because larger number of pixels are being modified. But the latter fares better in terms of visibility. The performance of the targeted attacks is shown in Figure 6.

TABLE 4. DIVISION OF IMAGES IN DATASET

Class	Number of images in training set	Number of images in testing set
0	5923	980
1	6742	1135
2	5958	1032
7	6265	1028

TABLE 5. SUMMARY OF TARGETED ATTACKS

Target	Misclassifications (%)	
	Altering pixels with fixed values	Altering pixels with maximum occurrences
0	49	32
1	77	0
2	96	71
7	71	5

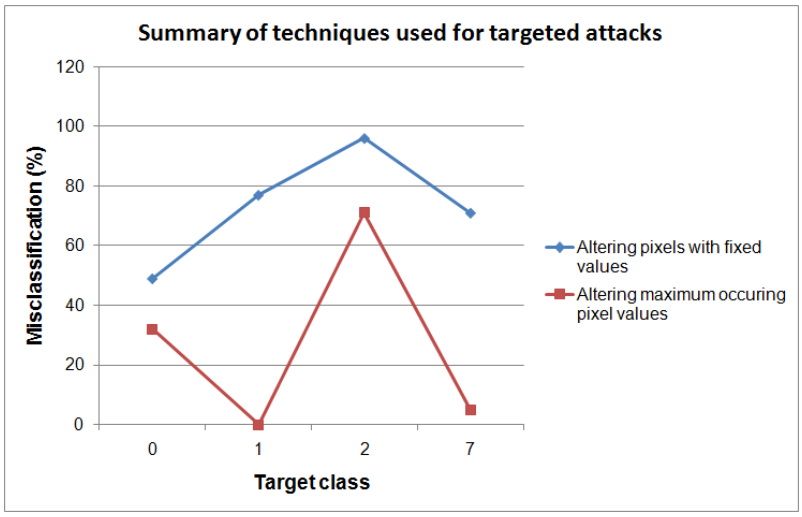

Figure 6. Techniques used Vs Misclassification (%) - Targeted attacks

Various untargeted and targeted attacks were applied on datasets such as MNIST and CIFAR-10. On comparing the two datasets, MNIST dataset has black and white images which uses single channel making it relatively easier to cause misclassification by making subtle changes when compared to CIFAR-10 having colored images with three channels. Since the untargeted attacks on CIFAR-10 did not yield results matching MNIST, targeted attacks were restricted to a subset of MNIST.

Untargeted attacks required the model to misclassify the image whereas targeted attacks required the model to misclassify the image to a particular class. The most successful untargeted attacks not only caused great amount of misclassifications but also made less prominent changes to the image. Comparatively, the targeted attacks required more visible changes in order to misclassify to the target class. This research work is an attempt, to prove that Convolutional Neural Networks are vulnerable to adversarial changes being made in images of different datasets.

5 Conclusion

Adversarial attacks are a form of cyber attacks on machine learning models where the attacker inputs intentionally modified examples in such a way that the changes are undetectable to the human eye but causes the model to make a mistake during classification or prediction. There are two types of adversarial attacks, targeted and untargeted. While an untargeted attack aims to misguide the Convolutional Neural Network (CNN), a targeted attack aims to misguide the CNN to a specific class.

In this research work, different attacks based on pixels were created in order to gain an understanding on adversarial attacks. The pixels present in the background and/or on the edges where modified by adding a specific value or by encrypting them using AES algorithm. Other methods that were implemented includes, heatmap generation using Gradient-weighted Class Activation Mapping (GRAD-CAM) and noise addition with edge pixel modification. Targeted attacks involved generating saliency maps to alter pixels having a particular value or to alter pixels with the maximum occurrences. Finally, the effectiveness and shortcomings of each technique were inferred and analyzed.

The existence of adversarial attacks limits the areas in which deep learning can be applied, especially for security-critical tasks. This is mainly because of their good transferability, i.e., the adversarial attacks crafted for one model remains effective for others. As a future addition, defense mechanisms to prevent adversarial attacks can be implemented.

6 Acknowledgements

The authors sincerely thank Dept. of CSE, Sri Sivasubramaniya Nadar College of Engineering for the permission given to utilize the High Performance Computing Laboratory and its GPU server for successful execution of this project.

Conflict of interest

The authors confirm that there is no conflict of interest for this publication.

References

1. Carlini N and Wagner D. Towards Evaluating the Robustness of Neural Networks. IEEE Symposium on Security and Privacy (SP), pp. 39-57. (2017)
2. Chakraborty A, Alam M, Dey V, Chattopadhyay A and Mukhopadhyay D. Adversarial Attacks and Defences: A Survey Machine Learning. <arXiv:1810.00069v1>. (2018)

3. Chen PY, Zhang H, Sharma Y, Yi J and Hsieh CJ. ZOO: Zeroth Order Optimization based Black-box Attacks to Deep Neural, Networks without Training Substitute Models. ACM Workshop on Artificial Intelligence and Security(AISec@CCS). pp. 15-26.(2017)
4. Dong Y, Liao F, Pang T, Su H, Zhu J, Hu X, and Li J. Boosting Adversarial Attacks with Momentum. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9185-9193.(2018)
5. Goodfellow IJ, Shlens J, and Szegedy C. Explaining and Harnessing Adversarial Examples. International Conference on Learning Representations. pp. 1-11.(2015)
6. Kurakin A, Goodfellow I, and Bengio S. Adversarial Machine Learning at Scale. International Conference on Learning Representations. pp. 1-17.(2015)
7. Su J, Vargas DV and Sakurai K. One Pixel Attack for Fooling Deep Neural Networks. IEEE Transactions on Evolutionary Computation. pp. 1-12. <doi: 10.1109/TEVC.2019.2890858>. (2019)
8. Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow I, and Fergus R. Intriguing Properties of Neural Networks. International Conference on Learning Representations. pp. 1-10.(2013)
9. Yuan X, He P, Zhu Q and Li X. Adversarial Examples: Attacks and Defenses for Deep Learning. IEEE Transactions on Neural Networks and Learning Systems, pp. 1-20. doi: 10.1109/TNNLS.2018.2886017 (2019)
10. Zhao YS, Zhu H, Shen Q, Liang R, Chen K and Zhang S. Practical Adversarial Attack against Object Detector. USENIX Security Symposium. <arXiv:1812.10217v3>. (2018)
11. CIFAR–10, https://www.cs.toronto.edu/~kriz/cifar.html , March 2020
12. MNIST, http://yann.lecun.com/exdb/mnist/ , March 2020
13. VGG-16 trained on CIFAR-10, https://github.com/mjiansun/cifar10-vgg16 , April 2020
14. Ilyas A, Santurkar S, Tsipras D, Engstrom L, Tran B, Madry A. Adversarial Examples Are Not Bugs, They Are Features. Computer Vision and Pattern Recognition. 2019. <arXiv:1905.02175v4>
15. Vargas AV and Su J. Understanding the One-Pixel Attack: Propagation Maps and Locality Analysis. 2019. <arxiv: 1902.02947v1>