

### Question1: Adversarial Attacks, Defense and Mitigation

**Dataset Used:** CIFAR10

1. Normalize the Dataset in the range of [0,1] by dividing images by 255.
2. Create one hot encoding with number of classes = 10

**Dataset Split:**

**Split the dataset into:**

1. **Train set:** 50,000 samples
2. **Validation set:** 10,000 samples
3. **Test set:** 10,000 samples

**Algorithm:** VGG16

**(a)**

**A. Test Accuracy:** 72.46 %

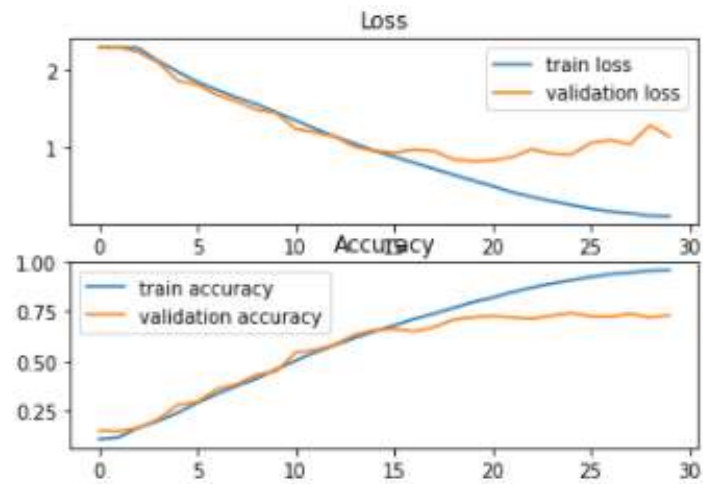
**B. Confusion Matrix:**

|           |   | Actual |     |     |     |     |     |     |     |     |     |
|-----------|---|--------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Predicted |   | 0      | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   |
|           | 0 | 758    | 4   | 55  | 18  | 36  | 3   | 4   | 15  | 76  | 31  |
|           | 1 | 21     | 713 | 3   | 9   | 4   | 0   | 7   | 5   | 76  | 162 |
|           | 2 | 59     | 0   | 636 | 80  | 104 | 34  | 41  | 21  | 18  | 7   |
|           | 3 | 18     | 1   | 97  | 614 | 70  | 83  | 45  | 40  | 18  | 14  |
|           | 4 | 19     | 3   | 62  | 59  | 741 | 22  | 32  | 53  | 3   | 6   |
|           | 5 | 9      | 0   | 66  | 258 | 73  | 497 | 15  | 67  | 8   | 7   |
|           | 6 | 3      | 4   | 35  | 116 | 44  | 7   | 769 | 2   | 16  | 4   |
|           | 7 | 8      | 0   | 34  | 46  | 79  | 35  | 0   | 785 | 4   | 9   |
|           | 8 | 50     | 1   | 12  | 19  | 5   | 0   | 2   | 2   | 881 | 28  |
|           | 9 | 40     | 21  | 6   | 22  | 4   | 2   | 5   | 15  | 33  | 852 |

### C. Class-wise Accuracy:



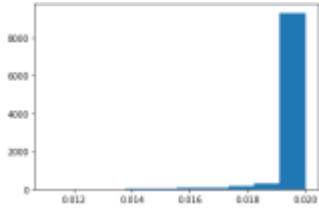


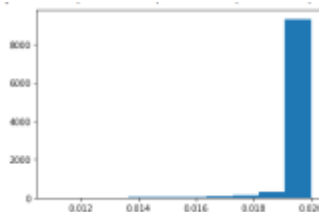


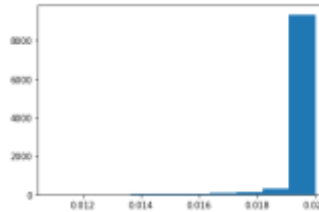
| Class    | Airplane<br>(0) | Automobile<br>(1) | Bird<br>(2) | Cat<br>(3) | Deer<br>(4) | Dog<br>(5) | Frog<br>(6) | Horse<br>(7) | Ship<br>(8) | Truck<br>(9) |
|----------|-----------------|-------------------|-------------|------------|-------------|------------|-------------|--------------|-------------|--------------|
| Accuracy | 76.95%          | 95.44%            | 63.22%      | 49.47%     | 63.87%      | 72.76%     | 83.58%      | 78.11%       | 77.75%      | 75.07%       |

### D. Loss Vs Epoch and Accuracy Vs Epoch Graphs:



(b)

### I. Untargeted Attacks on *Test Set*

| Attack  | Original Image  | Attacked Image  | Mean SSIM  | Test Accuracy (Initially) | Test Accuracy (Attack) | Histogram for magnitude of Perturbation   |
|---|---|---|------------|---------------------------|------------------------|---|
| Fast Gradient Sign Method (FGSM)<br>$\epsilon = 0.02$ |    |    | 0.96459904 | 72.46 %                   | 21.08 %                |    |
| Projected Gradient Descent (PGD)<br>$\epsilon = 0.02$ |   |   | 0.96418404 | 72.46 %                   | 21.33%                 |   |
| Basic Iterative Method (BIM)<br>$\epsilon = 0.02$     |  |  | 0.96418378 | 72.46 %                   | 21.33%                 |  |

## Observations and Analysis:

### A. Fast Gradient Sign Method (FGSM):

$$\eta = \epsilon \text{sign}(\nabla_x J(\theta, x, y))$$

Here, as we can see that  $\eta$  is the perturbation that is added to the original Image to attack it. Here I took  $\epsilon = 0.02$ . From the above table we can visualize very little amount seen-able perturbation in image. This leads to fool the VGG16 model trained on CIFAR10. As we can visualize that the similarity of both perturbed and non-perturbed images looks similar and thus they have the similarity index given by mean Structural Similarity Index (SSIM) is **0.96459904**. This tells us the amount of similarity between 2 images. The amount of perturbation added is the gradients of loss and multiplied by some constant factor  $\epsilon$  which helps in telling the amount of perturbation to add to image so that similarity in image also doesn't lose much and Model also get fooled. Due to BIM attack the accuracy dropped to **21.08%** because of more miss-classification happened by the model. This FGSM attack deals with linearizing the loss function and computing the perturbation which maximizes the loss function subject to L-infinity norm constraint. Thus it is the fastest Adversarial Attack technique.

### B. Basic Iterative Method (BIM)

$$X_0^{adv} = X, \quad X_{N+1}^{adv} = \text{Clip}_{X, \epsilon} \left\{ X_N^{adv} + \alpha \text{sign}(\nabla_X J(X_N^{adv}, y_{true})) \right\}$$



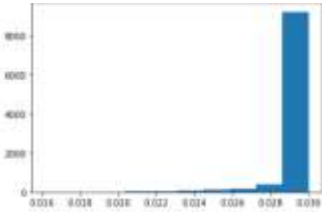


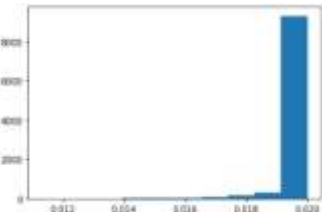


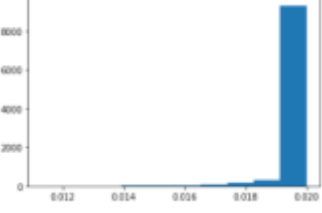
This method is also known as Iterative FGSM. BIM is introduced as an improvement to the FGSM attack. In BIM it is running similarly to FGSM but with small step size and clipping the pixel values after each step to ensure that they are in range of epsilon-neighborhood of non-perturbed image. Here, also I took  $\epsilon = 0.02$ . Here we get the mean SSIM value of **0.96418378**. Due to BIM attack the accuracy dropped to **21.33%** because of more miss-classification happened by the model.

### C. Projected Gradient Descent (PGD)

PGD attack is most of the same attack as BIM. However, the only change in PGD is that it initialized the example to a random point based on L-infinity norm and do random restarts whereas BIM initializes to original point only. Thus also, from above table we can see that the accuracy dropped due to PGD and BIM are same i.e., **21.33%** and mean Structural Similarity score (SSIM) is **0.96418404**

**Note: Therefore, observed that higher the mean SSIM and more the fooling of network better the attacked performed.**

## II. Targeted Attacks on *Test Set*

| Attack   | Original Image  | Attacked Image  | Mean SSIM  | Test Accuracy (Initially) | Test Accuracy (Attack) | Histogram for magnitude of Perturbation   |
|--|---|---|------------|---------------------------|------------------------|---|
| <b>Fast Gradient Sign Method (FGSM)</b><br><br>$\epsilon = 0.02$ |    |    | 0.93059524 | 72.46 %                   | 21.34 %                |    |
| <b>Projected Gradient Descent (PGD)</b><br><br>$\epsilon = 0.02$ |   |   | 0.96430591 | 72.46 %                   | 55.23%                 |   |
| <b>Basic Iterative Method (BIM)</b><br><br>$\epsilon = 0.02$     |  |  | 0.96430592 | 72.46 %                   | 55.23%                 |  |

### ***(g) Results after Adversarial Training***

**Split the dataset into:**

1. **Train set:** 1,28,000 samples
2. **Validation set:** 32,000 samples
3. **Test set:** 40,000 samples

**Algorithm: VGG16**

**A. Test Accuracy: 98.54 %**

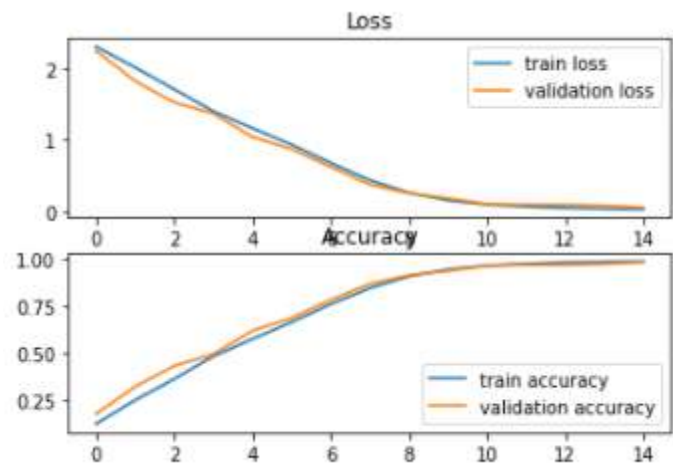
**B. Confusion Matrix:**

|           |   | Actual |      |      |      |      |      |      |      |      |      |
|-----------|---|--------|------|------|------|------|------|------|------|------|------|
| Predicted |   | 0      | 1    | 2    | 3    | 4    | 5    | 6    | 7    | 8    | 9    |
|           | 0 | 3064   | 0    | 13   | 1    | 0    | 0    | 0    | 3    | 26   | 2    |
|           | 1 | 2      | 3213 | 0    | 0    | 1    | 0    | 0    | 0    | 2    | 3    |
|           | 2 | 8      | 0    | 3174 | 4    | 8    | 1    | 0    | 0    | 3    | 0    |
|           | 3 | 5      | 0    | 13   | 3065 | 15   | 14   | 14   | 11   | 40   | 6    |
|           | 4 | 6      | 0    | 18   | 4    | 3205 | 3    | 8    | 30   | 0    | 0    |
|           | 5 | 0      | 2    | 19   | 31   | 11   | 3112 | 1    | 18   | 4    | 2    |
|           | 6 | 0      | 0    | 7    | 12   | 19   | 3    | 3177 | 1    | 1    | 4    |
|           | 7 | 7      | 0    | 1    | 11   | 4    | 5    | 1    | 3201 | 0    | 2    |
|           | 8 | 2      | 0    | 0    | 1    | 0    | 0    | 0    | 0    | 3205 | 0    |
|           | 9 | 7      | 2    | 0    | 2    | 0    | 0    | 0    | 16   | 6    | 3118 |

**C. Class-wise Accuracy:**

| Class    | Airplane<br>(0) | Automobile<br>(1) | Bird<br>(2) | Cat<br>(3) | Deer<br>(4) | Dog<br>(5) | Frog<br>(6) | Horse<br>(7) | Ship<br>(8) | Truck<br>(9) |
|----------|-----------------|-------------------|-------------|------------|-------------|------------|-------------|--------------|-------------|--------------|
| Accuracy | 98.80%          | 99.87%            | 97.81%      | 97.89%     | 98.22       | 99.17%     | 97.59%      | 97.59%       | 97.50%      | 99.34%       |

D. Loss Vs Epoch and Accuracy Vs Epoch Graphs:



I. Untargeted Attacks on *Test Set*

| Attack  | Original Image | Attacked Image | Mean SSIM  | Test Accuracy (Initially) | Test Accuracy (Attack) | Histogram for magnitude of Perturbation |
|---|----------------|----------------|------------|---------------------------|------------------------|---|
| Fast Gradient Sign Method (FGSM)<br><br>$\epsilon = 0.02$ |                |                | 0.97048231 | 98.54 %                   | 26.92 %                |   |
| Projected Gradient Descent (PGD)<br><br>$\epsilon = 0.02$ |                |                | 0.97027007 | 98.54 %                   | 25.43%                 |   |

|  |   |   |            |         |        |   |
|--|---|---|------------|---------|--------|---|
| <b>Basic Iterative Method (BIM)</b><br><br>$\epsilon = 0.02$ |  |  | 0.97027016 | 98.54 % | 25.43% |  |
|--|---|---|------------|---------|--------|---|

(d)

|             | <b>Compression rate : 40<br/>(Accuracy)</b> | <b>Compression rate : 70<br/>(Accuracy)</b> |
|-------------|---|---|
| <b>FGSM</b> | 26.12 %                                     | 25.92%                                      |
| <b>BIM</b>  | 28.95%                                      | 28.18%                                      |
| <b>PGD</b>  | 28.51%                                      | 28.32%                                      |

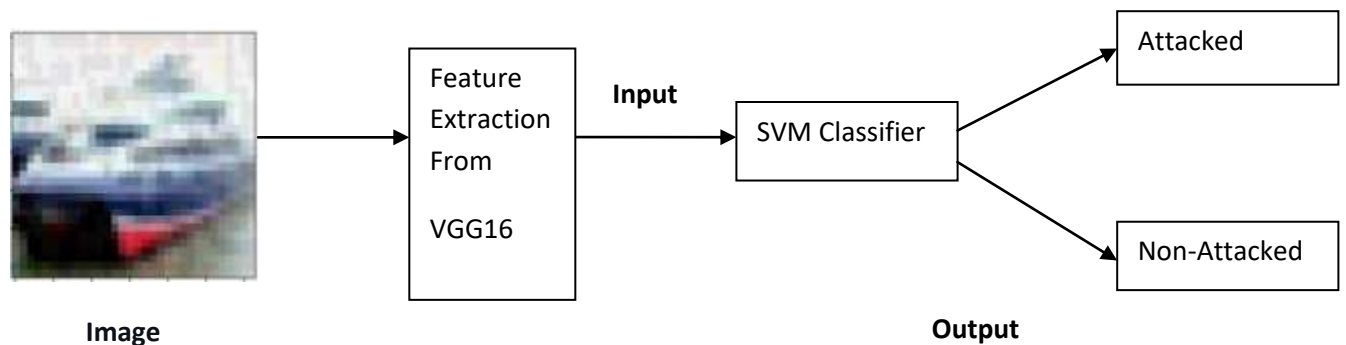
**(e) Results and Analysis:**

- On Observing and Analyzing the above tables we can analyze that the accuracy obtained is less on Perturbed images, especially for iterative attacks. I also observed that the perturbations are more visible in that case. There is a trade-off between visibility of perturbations and success of the attacks.
- Moreover, I noticed that after JPEG compression, I was able to slightly improve the performance of attacked images, leading to the observation that JPEG compression could be one way to mitigate the attacks.



## Question2: Adversarial Attack Detection

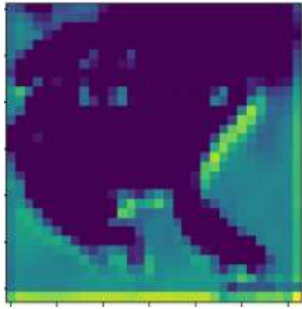
### *(a) Using SVM classifier for Attack Detection*



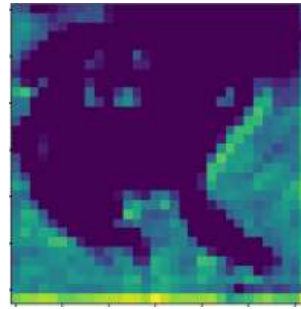
### **Steps for Attack Detection:**

1. Extract features from model trained earlier on Cifar10.
2. Give labels 0 to Attacked Image and 1 to Non-Attacked image.
3. Put these extracted features into SVM classifier with 2 class classification technique.
4. After training this Model we got Test Accuracy = 75.00%

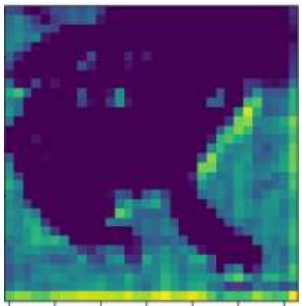
### ***(b) Network Activations***



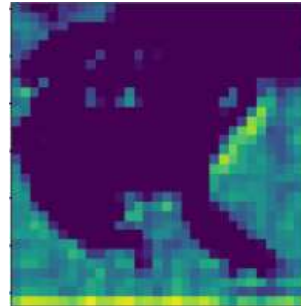
(i) Original Image's Activation



(ii) FGSM attacked Image's Activation



(iii) PGD attacked Image's Activation



(ii) BIM attacked Image's Activation

#### **Inference from Network Activations:**

- From above images we can visualize the attacked and non-attacked activation maps of images.
- We can visualize that after attack there are some visual changes in the activation maps of Neural Network model that we have used.
- As compared to original image in (i) it seems like normal activation map with no disruption whereas in images(ii),(iii),(iv) there are disruptions which can make the model predict it as wrong class label and misclassify it.
- However, these are the images of Activation maps of Model after Adversarial training, thus now the model is so robust that it can classify the frog images with these activation maps correctly without miss-classifying. Thus Adversarial Training helps in working over to protect images from being getting miss-classified even though there are some disruptions too.

### Question3: ADVERSARIAL ATTACKS IN NLP

**(a)**

- 1.** To deal with the problems of NLP to make the sentences in the way they can be put as input to the model. So, for that first we will tokenize the sentences. Then follow further strategies:
- 2.** Now we will Extract the features using bag of words on the basis of n-grams, max number of features and stop words. Similarly we will do for TF-idf technique.
- 3.** Then train the Linear SVM for these extracted features by above mentioned 2 features techniques.
- 4.** Testing Accuracy:
  - i. On BOW un-stemmed features: **60.41%**
  - ii. On tf-idf un-stemmed features: **60.78%**

***(c) Applied TEXTFOOLER attack which fools the learnt svm model***

***1. Performance on BOW features***

Accuracy dropped down to 0.0% because attack success rate is 100% and the amount of words attacked is 6.08%

***2. Performance on tf-idf features***

Accuracy dropped down to 0.0% because attack success rate is 100% and the amount of words attacked is 6.08%

***(d) Performance on Sentences after TEXTFOOLER attack which changed the sentiments on classification of attacked sentences.***

***1.***

**Original Sentence:** perhaps no picture ever made has more literally showed that the road to hell is pave with **good** intentions. -> **Positive sentiment**

**Attacked Sentence:** perhaps no picture ever made has more literally showed that the road to hell is pave with **descent** intentions. -> **Negative sentiment**

***2.***

**Original Sentence:** if you sometimes like to go to the **movies** to have **fun**, wasabi is a good place to start. -> **Positive sentiment**

**Attacked Sentence:** if you sometimes like to go to the **movie** to have **amuse**, wasabi is a good place to start. -> **Negative sentiment**

**References:**

[1]. <https://adversarial-robustness-toolbox.readthedocs.io/en/latest/>

[2]. [https://textattack.readthedocs.io/en/latest/2notebook/Example\\_1\\_sklearn.html](https://textattack.readthedocs.io/en/latest/2notebook/Example_1_sklearn.html)

## **Working Process of Code:**

### **A. For Ques1 and Ques2**

1. P21CS007\_colab\_main\_file.ipynb
2. P21CS007\_Compression.ipynb
3. P21CS007\_Attacks\_After\_adversarial\_training.ipynb

### **B. For Ques3**

1. P21CS007\_Textfooler.ipynb