Solution to “Problem: Adversarially Robust Classifier”

Wrik Bhadra
{wrik18027@iiitd.ac.in}

1. Introduction

This report contains partial solution to the research problem “Adversarially Robust Classifier”, namely it contains and discusses defense strategies to parts (i) and (ii) of question (1) – FGSM [1] and PGD [2] attacks to image classifiers trained on the CIFAR-10 dataset [3].
FGSM (Fast Gradient Sign Method) and PGD (Projected Gradient Descent) attacks are gradient-based techniques that aim to perturb an image so as to fool an image classifier to predict incorrectly, while parallely preserving visual integrity of the perturbed image. In other words, the modified images look identically similar to their originals. As per literature available with the research community and experiments performed as part of this exercise, these attack techniques have been shown to be highly effective against even state-of-the-art image classification deep neural networks.

2. Background

This section briefly touches upon the FGSM and PGD attacks and the defense mechanism adopted as the solution.

2.1 Attacks

The FGSM attack was introduced by Goodfellow et. al. in 2015 and works based on the following mechanism:

$$x_{adv} = x_{naive} + \epsilon * \mathrm{sign}(\nabla_{x\_naive} L(\theta, x, y)) \qquad (1)$$

The PGD attack was formulated by Madry et. al. in 2018 and follows the mechanism below for t iterations:

$$x_{adv} = \Pi(x_{naive} + \alpha * \nabla_{x\_naive} L(\theta, x, y)) \qquad (2)$$

For the equations (1) and (2) above,
x_{adv} => perturbed image created adversarially, x_{naive} => naïve/original image,
ϵ => perturbation, sign => sign operator, ∇ => gradient operator,
L => loss function with parameters θ, Π => projection operator, α => step size

2.2 Defense

Adversarial training has been implemented as a defense mechanism to produce robust image classifiers.

3. Experiments

This section begins with the experiemental setup followed by the methodology used and mini studies conducted for FGSM and PGD attacks. It then states further studies that may be conducted. All tests were performed using the test split of the CIFAR-10 dataset, as instructed in the problem.

3.1 Experimental setup

All experiments were conducted using Paperspace Gradient notebooks [4], which provide an online managed Jupyter Notebook instance with free GPUs. It is similar to Google Colab, Kaggle Notebooks and other such services. As part of available instances, pre-configured Ubuntu image with PyTorch v1.11 and TensorFlow v2.7 were available alongwith access to an Nvidia Quadro P5000 8GB GPU. The instance had an octa-core CPU and 30GB RAM.

3.2 FGSM studies

Two broad studies were conducted while exploring a solution for the FGSM attack – first, using a ResNet-18 model and secondly, using a simple CNN model to understand how the attack affects each of them.

3.2.1 ResNet-18

Experiments using this model were done with the PyTorch framework. As the training set contains less data points (5000 images per class), data augmentation was done followed by normalization of images (specific to the CIFAR-10 dataset). Please refer to the code in "./fgsm/pytorch/" for more details. For training, the SGD optimizer was used alongwith Cosine Annealing learning rate scheduler as an effective learning strategy. The loss function was multi-class Cross Entropy Loss.

Both untargeted and targeted attacks were performed and accuracy before and after adversarial training were recorded. For untargeted attacks, the perturbation ϵ was varied to study its effect on accuracy. For targeted attacks specifically, a mini study was performed by taking the highest probability in the output vector (other than the actual class) as the target class and the same with the lowest probability.

3.2.2 Simple CNN

Experiments using this model were done with the TensorFlow framework. Since the server was unavailable when trying to fetch the dataset from tensorflow.keras.datasets, the same was manually downloaded as a zipped file and extracted to the same location as the corresponding python script. As pre-processing steps, pixels of the images were scaled to [0,1] and the labels one-hot encoded. For training, the Adam optimizer was used with a learning rate 10^{-3} and the loss function was Categorical Cross Entropy Loss. In this case, the perturbation was kept constant at 0.01 and there was an equal mix of normal and perturbed images during adversarial training.

3.3 PGD studies

The ResNet-18 model was used to conduct experiments in this case as well. Considerations were same as for § 3.2.1 above. Additionally, the number of iterations were varied to study its effect on test accuracy.

3.4 Further studies

Several other studies were thought of but could not be performed owing to the paucity of time. They are listed below (no particular order):

- Vary α (step size) and ϵ (perturbation) for the PGD attacks and check test accuracy.
- Implementing attack code from toolboxes such as ART [5], Foolbox [6], etc.
- Exploring optimal learning rate during training from Leslie Smith's paper [7].
- Varying mix percentage of normal and adversarial images in § 3.2.2 above.

4. Results

This section reports the experimental findings of various studies specified in § 3.2 and § 3.3 in the previous page and draws inference from them. Plots of loss & accuracy, perturbed images and other results are included in the supplementary section to comply with the 4-page prime limit.

4.1 FGSM studies

4.1.1 ResNet-18 (PyTorch)

Network type	Test accuracy (%)	Perturbation ϵ	Test image type	Attack type	Target class
naïve	88.13	-	unperturbed	-	-
adversarially-trained	86.55	-	unperturbed	untargeted	
	85.79	4.0/255	perturbed		
	85.06	8.0/255			
	84.24	12.0/255			
naïve	30.40	8.0/255	perturbed	targeted	highest probability
	48.51				lowest probability
adversarially-trained	84.79				highest probability
	85.03				lowest probability

Table 1: Test accuracy measures by varying ϵ, image type, attack type and target class for FGSM attacks using the ResNet-18 (PyTorch) model

From Table 1, we observe that the test accuracy dips as we increase the perturbation. This makes sense as it becomes more difficult for the model to correctly label the perturbed examples, since for higher ϵ more perturbations in images are allowed.

For the targeted FGSM attack, we observe that the test accuracy of the adversarially-trained model in both the cases of target class is much higher than the naïve model since the former has become more robust and therefore has higher accuracy against adversarial examples.

4.1.2 Simple CNN (TensorFlow)

Network type	Train accuracy (%)	Test accuracy (%)	Perturbation ϵ	Test image type
naïve	88.59	67.67	-	unperturbed
		7.20	0.01	perturbed
adversarially-trained	63.94	69.45	-	unperturbed
		38.38	0.01	perturbed

Table 2: Accuracy measures by varying ϵ and image type for FGSM attacks using the Simple CNN (TensorFlow) model

From Table 2 in the previous page, we observe that even small values can cause the accuracy to drop massively on the naïve image classifier when testing perturbed (adversarial) images. However, once the model is adversarially-trained with an equal mix of original and perturbed images, the accuracy surpasses that of the baseline. This is probably because the model has seen more varied examples during adversarial training.
Another observation: despite adversarial training, the accuracy on perturbed images is low overall when compared to the ResNet-18 case. The primary reason behind this could be the network has overfit on the training data, which is not surprising as the CIFAR-10 dataset has few training examples. Another reason: since the network is too simple, it has not been able to learn the discriminating features too well.

4.2 PGD studies

Network type	Test accuracy (%)	Iterations t	Perturbation ϵ	Step size α	Test image type
naive	90.01	-	-	-	unperturbed
adversarially-trained	89.07	-	8.0/255	3.0/255	unperturbed
	87.92	3			perturbed
	86.41	7			
	84.94	12			

Table 3: Test accuracy measures by varying #iterations and image type for PGD attacks using the ResNet-18 (PyTorch) model

From Table 3 above, we observe that as the #iterations increase, the test accuracy decreases. This is because with the rise in iterations, the PGD attack algorithm produces stronger attacks making it harder for the model to predict perturbed image labels correctly. Also, if one refers to the code, one can see that the model was adversarially-trained with 3 iterations. Hence, the test accuracy for t=3 also closely resembles the same. But it is difficult for the model to defend against 2 or 4 times more PGD attack iterations.

5. Conclusion

As part of the solution to the research problem, the author has demonstrated the use of adversatial training as a robust general defense strategy against both FGSM and PGD attacks on image classifiers. Several experiments were performed by varying hyperparameters to verify this claim. For the FGSM attack specifically, experiments conducted show the generalization of the claim for both a sophisticated ResNet-18 and a simple CNN model. Several other studies were thought of, as listed in § 3.4, but could not be performed owing to paucity of time. Such studies are expected to reveal more relevant findings.

6. References

[1] I.J. Goodfellow, J. Shlens, C. Szegedy. Explaining and Harnessing Adversarial Examples. In *ICLR 2015*.

[2] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, A. Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks. In *ICLR 2018*.

[3] A. Krizhevsky, V. Nair, G. Hinton. CIFAR-10. University of Toronto, 2009. {http://www.cs.toronto.edu/~kriz/cifar.html}

[4] Paperspace Gradient Notebooks. {https://gradient.run/notebooks}

[5] Adversarial Robustness Toolbox. {https://adversarial-robustness-toolbox.readthedocs.io/en/latest/}

[6] Foolbox Native. {https://foolbox.readthedocs.io/en/stable/}

[7] L.N. Smith. Cyclical Learning Rates for Training Neural Networks. In *WACV 2017*.

[8] A. Rosebrock. Adversarial attacks with FGSM (Fast Gradient Sign Method). 2021.

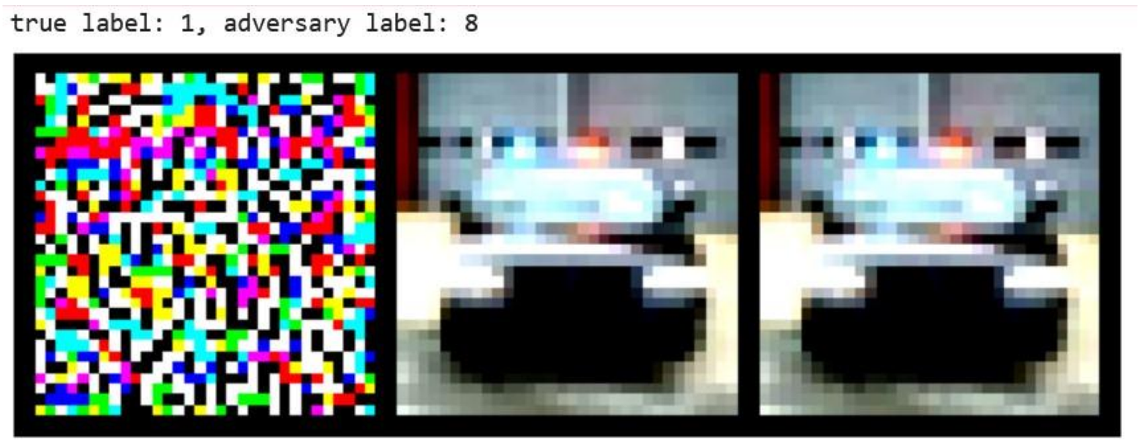
{pyimagesearch.com/2021/03/01/adversarial-attacks-with-fgsm-fast-gradient-sign-method/}

[9] O. Knagg. Know your enemy: How you can create and defend against adversarial attacks. 2019. {https://towardsdatascience.com/know-your-enemy-7f7c5038bdf3}

[10] Z. Kolter, A. Madry. Adversarial Robustness: Theory and Practice. In NeurIPS 2018. {https://adversarial-ml-tutorial.org/}

Supplementary

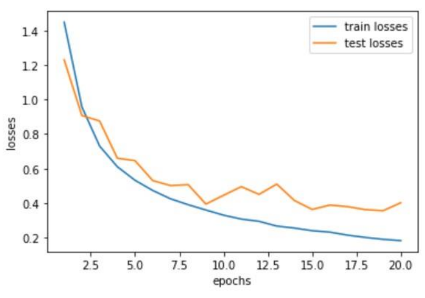
1. Perturbed images for FGSM attack using ResNet-18 (PyTorch) model



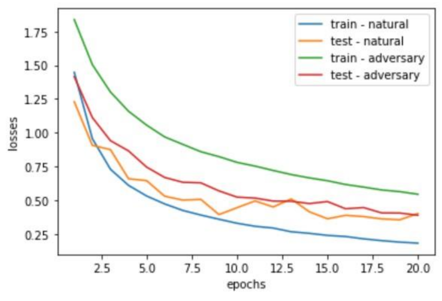
```
true label: 1, adversary label: 8
```

Figure 1: (from left to right) perturbation visualised, original image with true label as 1 (class "automobile" in CIFAR-10), perturbed image with incorrect label as 8 (class "ship")

2. Model losses against #epochs for FGSM attack using ResNet-18 (PyTorch) model



(a) naive model

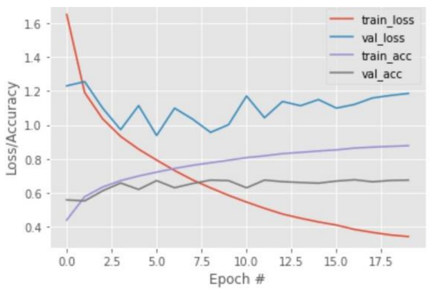


(b) naïve and adversarially-trained model

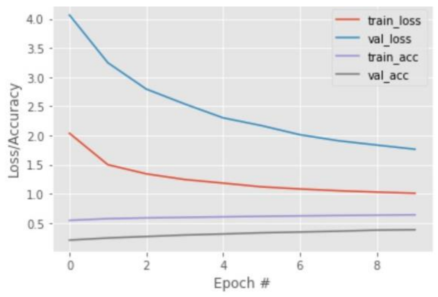
Figure 2: Model train and test loss plotted against #epochs

One can observe that the train loss of the adversarially-trained model is higher than train loss of the naïve model, which is intuitive since the adversarially-trained model is trained against adversarial examples, which makes it harder for the same to label the unperturbed inputs correctly and results in higher errors.
The loss of the naïve model on test data is higher than the train loss, since test data is unseen by the model, resulting in higher error in classification.
However, the loss of the adversarially-trained model on test data is lower than the corresponding train loss. This is probably because the test instances are not adversarial (in contrast to train data) and that the model has learned to extract important and useful features, thus performing better on test images.

3. Model losses against #epochs for FGSM attack using Simple CNN (TensorFlow) model



(a) train & val loss/acc before defense



(b) train & val loss/acc after defense

Figure 3: Model train and validation loss/accuracy plaotted against #epochs

One can observe that the validation metrics indicate a poor performance before applying defense while the values don't differ as much after applying mixed adversarial training as defense.

4. Model losses against #epochs for PGD attack using ResNet-18 (PyTorch) model

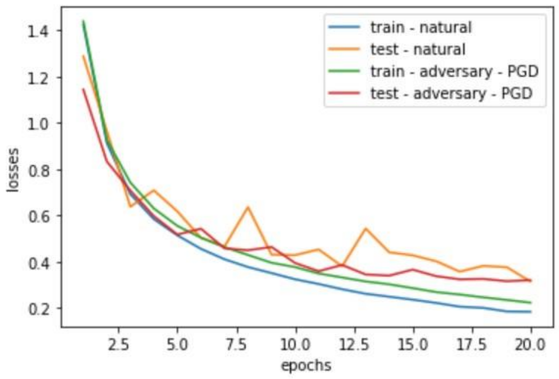


Figure 4: Model train and test loss plotted against #epochs

We observe that the test loss is comparatively lower than the train loss because the adversarially-trained model has probably learnt more differentiating features than its naïve counterpart.

- END -