Using Torchattacks to Improve the Robustness of Models with Adversarial Training

William S. Matos Díaz

School of Cybersecurity, Old Dominion University

Abstract

Adversarial training has proven to be one of the most successful ways to defend models against adversarial examples. This process consists of training a model with an adversarial example to improve the robustness of the model. In this experiment, Torchattacks, a Pytorch library made for importing adversarial examples more easily, was used to determine which attack was the strongest. Later on, the strongest attack was used to train the model and make it more robust against adversarial examples. The datasets used to perform the experiments were MNIST and CIFAR-10. Both datasets were put to the test using PGD, FGSM, and R+FGSM attacks. The results that will be discussed will prove that the PGD attack makes the MNIST, and CIFAR-10 model more robust. The results at the end show that this technique for defending models needs to be improved with more research.

Introduction

Basic Concepts

- Machine learning algorithms, also called models, imitate the way humans learn by using the patterns of an image to predict a specific input.
- Pytorch is a framework that can be used to train a model.
- When a model is trained, it goes from different iterations called epochs.
- Adversarial examples are carefully crafted attacks that fool a model by adding vectorial noise that the human eye cannot detect.
- Torchattacks is a Pytorch library used to import adversarial examples.
- *Adversarial training* is a technique used to train a model using an adversarial example to make the model more robust.
- Datasets are a collection of images that a model can learn and predict after training.
- Batch size is the number of images used to train the model.

Objectives

- Find strong adversarial examples from TorchAttacks and see which one is best suited for adversarial training.
- Test the strongest attack against the MNIST and CIFAR-10 datasets.
- Use adversarial training to make the models more robust.

Example:

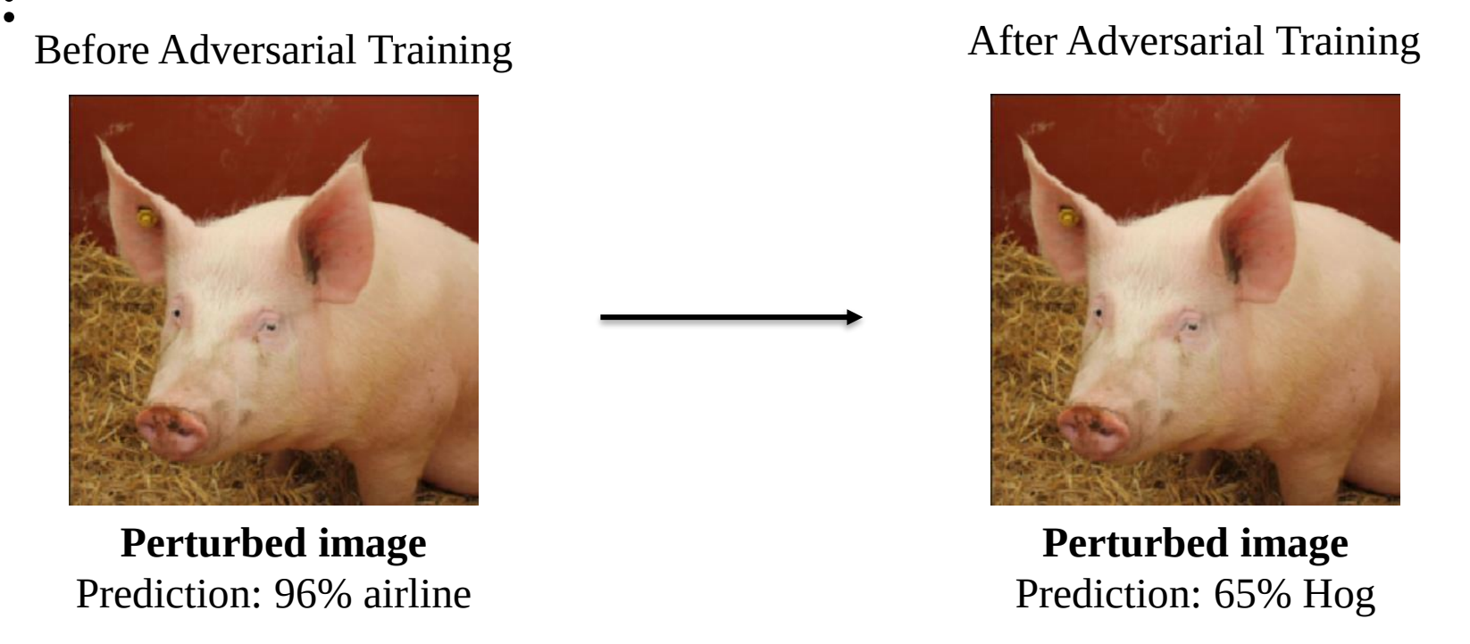

Figure 1: Demonstration of what happens to a perturbed image before and after adversarial training. The image is from the ImageNet dataset.

Background on Adversarial Training

- Adversarial training has proven to be the most effective way to defend models against adversarial examples.
- Due to the computational cost required to train a dataset with high dimensionality, the best attacks to tests on adversarial training are FGSM and PGD.
- Adversarial training is not an entirely satisfactory defense method yet.

General Theory

$$h^* = \underset{h \in \mathcal{H}}{\arg\min} \; \underset{(x, y_{\text{true}}) \sim \mathcal{D}}{\mathbb{E}} \left[\max_{\|x^{\text{adv}} - x\|_\infty \leq \epsilon} L(h(x^{\text{adv}}), y_{\text{true}}) \right].$$

Figure 2: Equation used for adversarial training (Madry et al. 2017)

The idea is to inject adversarial examples into training process and train the model either on adversarial examples or on mix of clean and adversarial examples.

Methods

SOFTWARES AND TOOLS

HARDWARE
Every test on the research was conducted on a machine with an AMD Ryzen 7 4800HS(2.90GHz), an RTX 2060 with Max-Q Design, 16GB of memory, and Windows 10.

JUPYTER NOTEBOOK
Jupyter Notebook is an open-source, interactive web tool used to run code and make annotations about it. This software provided a way to clone repositories from Github to the Jupyter Notebook environment and was used to perform every experiment.

PYTORCH
This framework accelerates researching machine learning algorithms, providing a way to train models and dynamic computational graphs.

TORCHATTACKS
This Pytorch library provided an easier way to import adversarial examples into the algorithms. Torchattacks provided the necessary attacks to do the tests on the different datasets. Other functions of Torchattacks are to provide a way to do targeted attacks, auto-attacks, and adversarial training.

ALGORITHM

ADVERSARIAL TRAINING PROCESS

The GitHub repository of Torchattack provided the code and the neural networks used for the experiments. The model used to test the MNIST dataset has a standard accuracy of 98%, while the model used for the CIFAR-10 dataset has a 98% accuracy. The process of adversarial training goes as follows. Firstly, the model must be trained on sets of data images called batches. The size of the batch used in the MNIST and CIFAR-10 algorithm consists of 128. Once this part was set, the training began with five epochs. Each epoch was of 468 iterations. This training was based on clean and perturbed images due to the PGD attack declared before the training. This means that the training was done while the model was being attacked. The three attacks that were tested were PGD, FGSM, and R+FGSM. The parameters of these attacks were based on the Torchattack documentation. Once the training was completed, the model was evaluated to see if the accuracy had changed. Each attack was repeated to see how robust the models were once trained on the PGD attack.

For the evaluation of the model, after it was trained on the PGD attack, the algorithm calculated the correct images predicted divided by the total of images used. The following is a demonstration of the algorithm used for the experiments.

Figure 3: Evaluation of the model after the adversial training process was finished.

```
▶ RUN EVALUATION OF THE MODEL
for every m images, and m labels in the test loader:

images = the attack used to train the model (k images, k labels)
outputs = the trained images

predicted = maximum value of all elements

total = size of the tensor, where every m image and label is.
correct = will output all the correct images predicted by the model
   output the robust accuracy using the formula: (100 * float(correct) / total), where it will give the percentage of the accuracy.
end for
```

Results

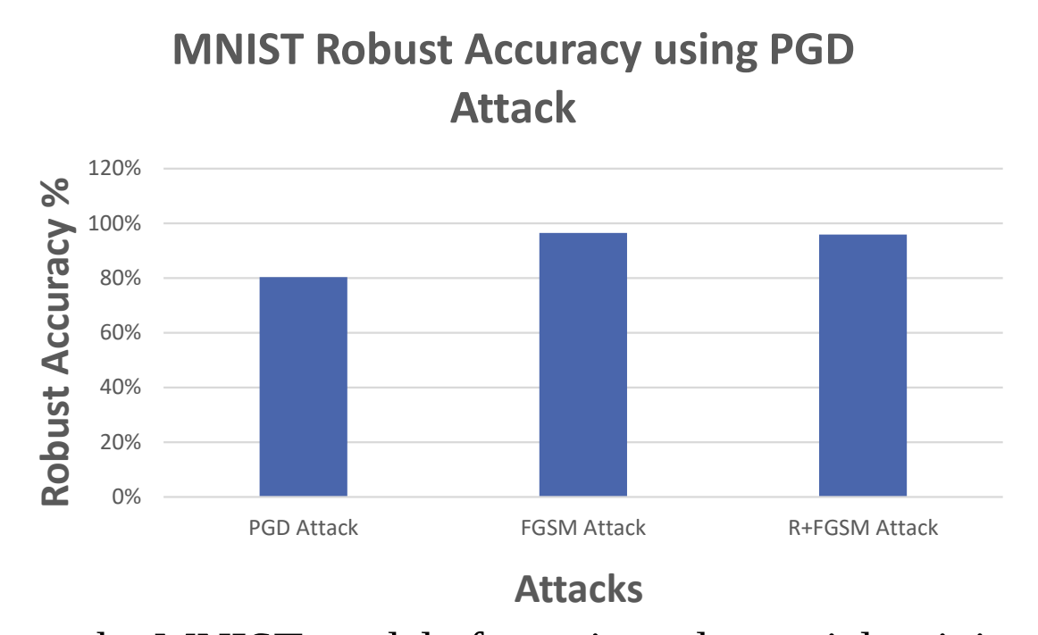

Figure 4: Results on the MNIST model after using adversarial training with a PGD attack.

The first step was to find which attack was the strongest to be later implemented in adversarial training. Both models were attacked using a PGD, FGSM, and R+FGSM attack imported with Torchattacks. The attack that had the most success with the MNIST dataset was the PGD attack. The MNIST model was defended against all attacks with a high success rate by training the model with this adversarial example. The PGD attack had an eps of 0.3, which is the default value indicated in the Torchattack docs. After the model was trained with the adversarial example, the attacks were put to the test:

1.The PGD attack had an accuracy of 80%.
2.The FGSM attack had a robust accuracy of 97%.
3.The R+FGSM attack had a robust accuracy of 96%.

On the other hand, there is Figure 5. In this part of the experiment, the dataset used is the CIFAR-10. The exact process is repeated in which the strongest attack is found by attacking the model and observing which adversarial example had the most impact on the model itself. When this part was determined, the adversarial training began. The strongest attack in this model was the PGD attack as well. After the model was trained, the three attacks were put to the test:

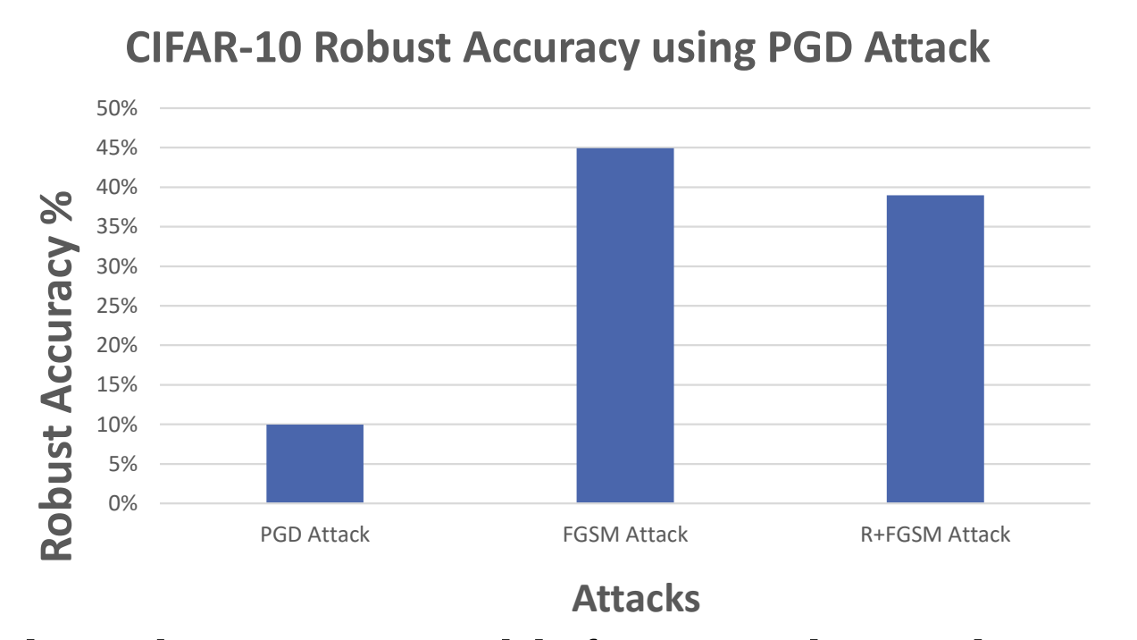

Figure 5: Results on the CIFAR-10 model after using adversarial training with a PGD attack.

1.The PGD attack had a robust accuracy of 10%.
2.The FGSM attack had an accuracy of 45%.
3.Finally, the R+FGSM attack had a robust accuracy of 39%.

These attacks were all using the defaults parameters determined in the Torchattacks documents. For a more in detail perspective of the results, the tables below are shown:

TABLE 1: ADVERSARIAL TRAINING WITH MNIST

MNIST Dataset with PGD		
Attacks	Robust Accuracy(%)	Eps
PGD Attack	80%	0.3
FGSM Attack	97%	0.007
R+FGSM Attack	96%	0.07

TABLE 2: ADVERSARIAL TRAINING WITH CIFAR-10

CIFAR-10 Dataset with PGD		
Attacks	Robust Accuracy(%)	Eps
PGD Attack	10%	0.3
FGSM Attack	45%	0.007
R+FGSM Attack	39%	0.07

Discussion

It can be observed that training a model using PGD improves the models' accuracy significantly. This attack can defend against the other two attacks that were put to the test. Results show that PGD is an excellent option for training a model against other adversarial examples, at least for the MNIST dataset. Both tables show that doing adversarial training with PGD does a great job defending against FGMS and R+FGSM due to the high accuracy. Furthermore, when defending a model against the adversarial example it was trained on, the model does not do as well as the other attacks. Also, it can be observed that the model that uses the CIFAR-10 dataset does not do very well against adversarial examples, meaning the adversarial training technique does not work well with this dataset. It will be required to do more training iterations to make this dataset more robust. In other words, CIFAR-10 has a high computational cost when training a model with this dataset.

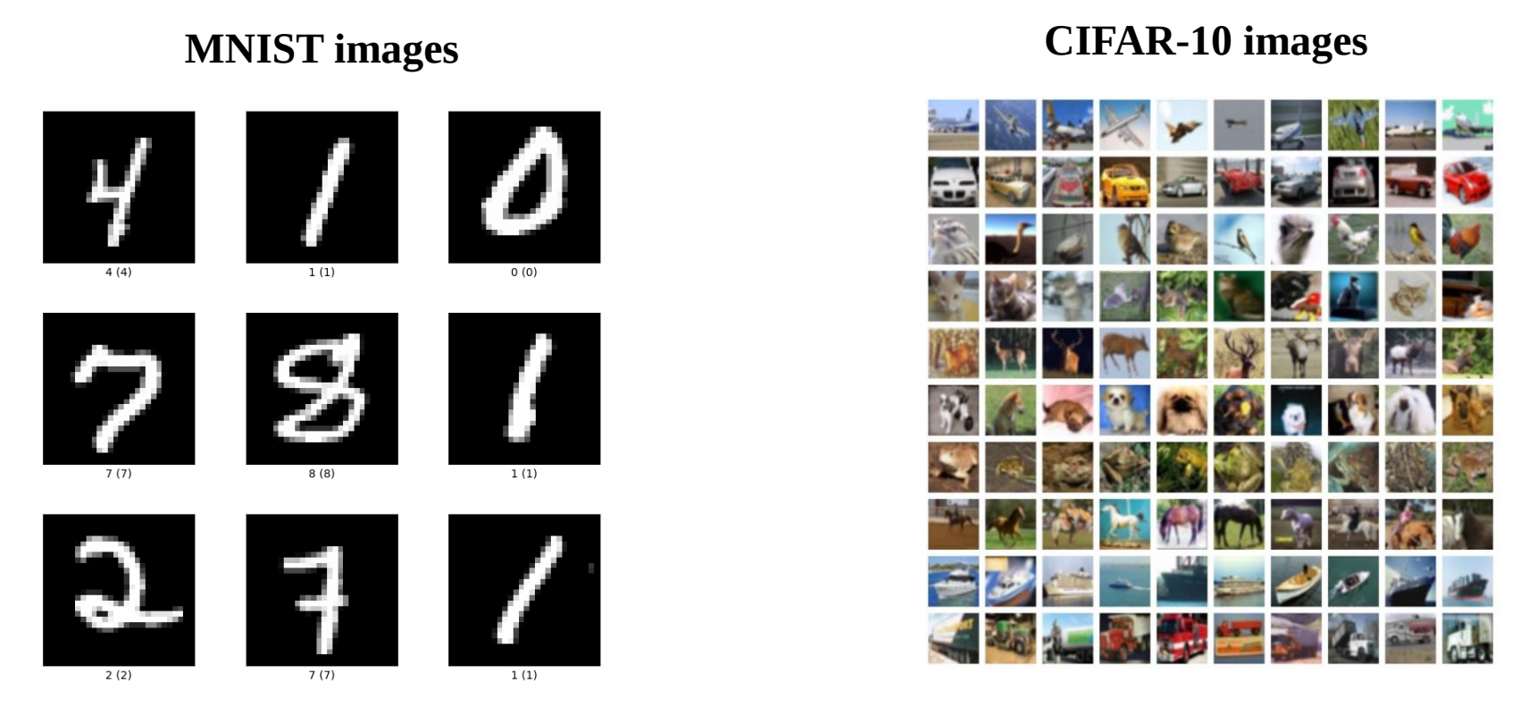

Figure 6: Output images of the MNIST and CIFAR-10 datasets.

References

- Kurakin A. et al. (2018) Adversarial Attacks and Defences Competition. In: Escalera S., Weimer M. (eds) The NIPS '17 Competition: Building Intelligent Systems. The Springer Series on Challenges in Machine Learning. Springer, Cham. https://link.springer.com/chapter/10.1007/978-3-319-94042-7_11
- Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083 .
- Papernot, N.; McDaniel, P.; Sinha, A.; and Wellman, M. 2016. Towards the science of security and privacy in machine learning. arXiv preprint arXiv:1611.03814 .
- Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 .
- Education, I. C. (August 3, 2021). *Machine Learning*. IBM. https://www.ibm.com/cloud/learn/machine-learning
- Madry, Z. K. A. A. (s. f.). Chapter 1 - Introduction to adversarial robustness. Chapter 1 - Introduction to Adversarial Robustness, https://adversarial-ml-tutorial.org/introduction/

Acknowledgments

- This research experience was sponsored by the National Science Foundation and hosted by The Old Dominion University.
- Special thanks to the mentor Dr. Hongyi Wu.
- Special thanks to the directors of the program Dr. Chunsheng Xin and Dr. Khan Iftekharuddin for the opportunity to participate in the research program.
- Special thanks to the graduate student (Rui Ning) that guided me throughout this research.