Comprehensive Comparison: Pixel Perturbation Versus Spatial Transformation

Justin Zhang
justin.zhang.1@stonybrook.edu
112615200

Arvin Wang
arvin.wang@stonybrook.edu
112884226

1. Introduction

Adversarial machine learning has become more prominent with the rise of AI technologies throughout the last few years. These adversarial examples are created with various methods, such as pixel perturbation, spatial transformation, and many more, causing the network to misclassify the inputs. The study of these attacks is important as we can determine deciding factors that can prevent these attacks. In this project, we are targeting two methods, pixel perturbation, and spatial transformation, to research further into the correlations between the two attacks. There have been several works that have been done on pixel perturbation with the first of its work in the Goodfellow et al. (2014) paper. This has led to additional studies on the attack and works such as detecting pixel perturbation attacks and possible stronger attacks.

On the other hand, spatial transformations are still an area in this field that requires further exploration. The first of its works was brought up by a method proposed in Xiao et al. (2018) paper. This paper showed a possible implementation of spatial transformation and compared it with attacks such as pixel perturbation without much discussion on how it was compared. This was the last of the works done on spatial transformation. Until today, there has been no paper discussing how to prevent the attack nor much information relating to it. Hence, we would like to perform a comparative study between the attacks individually to see which would lead to a higher misclassification and under what conditions. These conditions included the neural network architecture, hyperparameters used by the algorithms, and the datasets.

2. Attacks

2.1 Pixel Perturbation - FGSM

For pixel perturbations, we are using the FGSM attack proposed in the Goodfellow et al. (2014) paper. Pixel perturbations involve directly modifying each pixel value by some value so

that the classifier will misclassify the example. FGSM involves following the gradient of the cost function to produce an adversarial example; the adversarial example is generated using the equation in Figure 1.

$$\boldsymbol{\eta} = \epsilon \text{sign}\left(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y)\right)$$

Figure 1: FGSM Equation

We chose this attack mainly because it was simple to understand and an algorithm has been widely standardized.

2.2 Spatial Transformation - Flow Field

For spatial transformations we decided to use the method proposed in Xiao et al. (2018) paper. This method takes advantage of a flow field that will subtly transform pixels of an image based on the flow field. It essentially just shifts the pixels in a given direction slightly and an example of this is in figure 2.

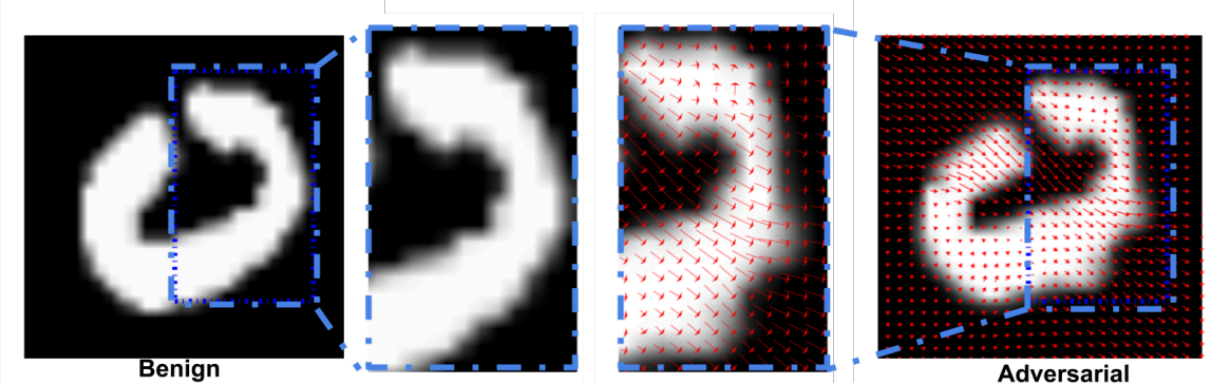

Figure 2: Flow Field Example

The underlying algorithm will go as follows

1. Generate a flow field.
2. Decompose the original image into the adv pixel + the flow field
3. Generate the adversarial example following the equation in figure 3.

$$f_i := (\Delta u^{(i)}, \Delta v^{(i)})$$

$$(u^{(i)}, v^{(i)}) = (u^{(i)}_{\text{adv}} + \Delta u^{(i)}, v^{(i)}_{\text{adv}} + \Delta v^{(i)})$$

$$\mathbf{x}^{(i)}_{\text{adv}} = \sum_{q \in \mathcal{N}(u^{(i)}, v^{(i)})} \mathbf{x}^{(q)} (1 - |u^{(i)} - u^{(q)}|)(1 - |v^{(i)} - v^{(q)}|)$$

Figure 3: Spatial Transformation

3. Data and Neural Network Architectures

3.1 Data

For the data we decided to choose both the MNIST dataset and the CIFAR10 datasets. This was to determine if the strength of an attack is dependent on the dataset used and our results will be displayed in a later section. MNIST are black and white images displaying labels 0-9, while CIFAR10 is displaying image labels such as airplane, bird, cat, and etc.

3.2 Neural Network Architectures

We decided to use two different architectures for this project VGG16 and Resnet18. This was to determine if the strength of the attack is dependent on the architecture of the neural network that the attack was used on. The Resnet18 was the more complicated between the two neural network architectures containing multiple convolutional layers to classify and learn about images and its label.

4.Testing and Results

4.1 Bounding an attack

Given the nature of the test we needed to ensure that the attacks were evaluated on the same baseline. This meant that we needed a specific way to ensure that the hyper parameters chosen did not clearly overpower one attack versus another. The method we used was to compare the feature L2 norms between the two attacks.

In a conventional convolutional neural network, there is a feature extraction step and a classification step. We made use of the pre-trained Inception model to extract the features at the end of the pooling layer as shown in figure 4.

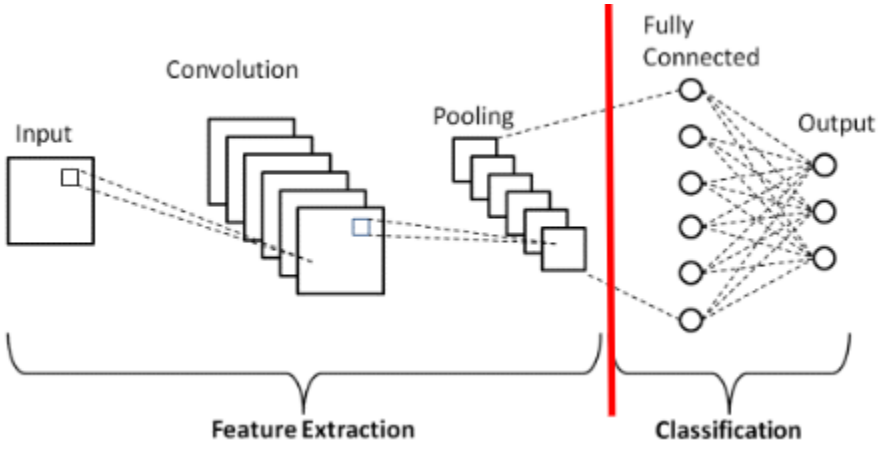

Figure 4

With the extracted features, we then calculated the L2 norms of the original - extracted features and chose hyperparameters based on how similar the L2 norms are. In the tables we can observe that each row is comparable with one another.

Epsilon	**L2-Norm**
0.1	14.69
0.3	16.87
0.5	16.61

FGSM L2- Norms

ld_tv	**ld_adv**	**L2-Norm**
.14	.6	14.37
.3	1	15.34
.175	3.5	15.1

Spatial Transformation L2-Norms

FGSM had only one hyper parameter which was epsilon. Spatial transformations had two which were used in the loss calculations. The variable ld_tv is responsible for the smoothness of the adversarial example while the ld_adv is responsible for how much the original image is perturbed. The values used above were found with random testing and the resulting values were what we found to be the best hyperparameters. Below is an example of what the output looked like in Figure 5, the left is the original, center is FGSM, and right is the spatial transformation.

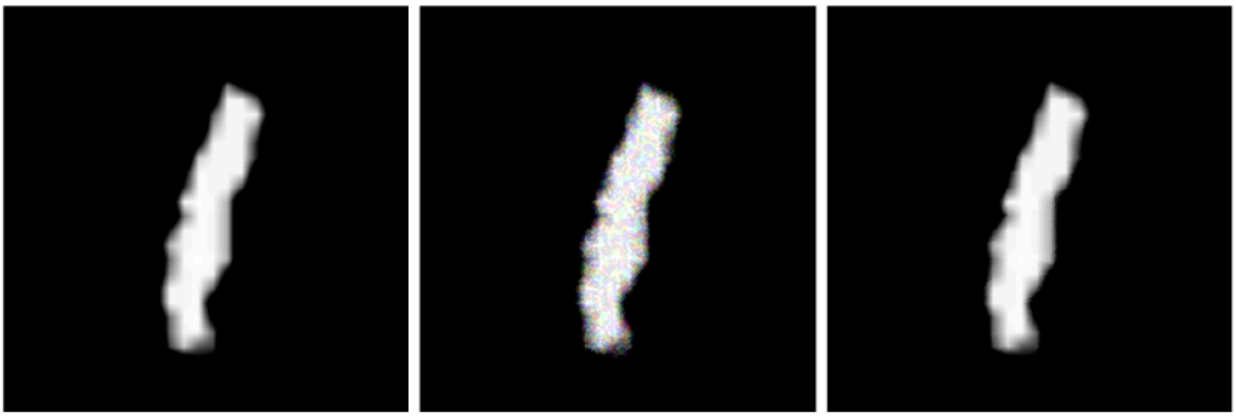

Figure 5: Image comparison

4.2 Results

In this section we will be displaying the accuracy of each test along with the selected parameters, data sets, and model architecture.

4.2.1 MNIST + Resnet

Epsilon	Accuracy
0.1	85%
0.3	78%
0.5	73%

FGSM Accuracy

ld_tv	ld_adv	Accuracy
.14	.6	12%
.3	1	11%
.175	3.5	10%

Spatial Transformation Accuracy

4.2.2 MNIST + VGG

Epsilon	Accuracy
0.1	95%
0.3	80%
0.5	73%

FGSM Accuracy

ld_tv	ld_adv	Accuracy
.14	.6	9%
.3	1	8%
.175	3.5	9%

Spatial Transformation Accuracy

4.2.3 CIFAR10 + Resnet

Epsilon	Accuracy
0.1	27%
0.3	19.5%
0.5	16%

FGSM Accuracy

ld_tv	ld_adv	Accuracy
.14	.6	12%
.3	1	10%
.175	3.5	12%

Spatial Transformation Accuracy

4.2.4 CIFAR10 + VGG

Epsilon	Accuracy
0.1	23%
0.3	22%
0.5	20.67%

FGSM Accuracy

ld_tv	ld_adv	Accuracy
.14	.6	11%
.3	1	10%
.175	3.5	10%

Spatial Transformation Accuracy

5. Conclusion

It's evident that spatial transformation emerges as the most potent attack overall, regardless of which factors are involved. Through extensive experimentation with diverse datasets, network architectures, and hyperparameters, spatial transformations consistently outperformed FGSM, resulting in a higher rate of misclassified inputs across the board.

FGSM was very controllable in terms of accuracy and attack intensity. In contrast, the spatial transformation attack posed challenges in control while remaining the most effective attack. The L2 norms of the spatial transformation features were very volatile and often exhibited sudden yet large fluctuations with minor adjustments to hyperparameters. Because of this, our L2 norms were not exactly one-to-one when comparing but we managed to obtain norms that were similar enough such that the attacks were of “similar” strength within reason. After attempting to achieve a similar L2 norm between the two attacks, we see that spatial transformation achieved lower yet consistent classification accuracy. Regardless of the network architecture and datasets, it maintained an accuracy of around 10%. Pixel perturbation, on the other hand, achieved the lowest accuracy of 73% on the MNIST dataset and 16% on the CIFAR10 dataset. The models were all pretrained and fine-tuned to each dataset and yet the spatial transformation was able to fool the classifier and maintain the same accuracy throughout.

From this experiment, we can establish that spatial transformation is an uprising attack that can cause serious damage if not properly defended against. Even with minimal L2 norm difference, it can fool the classifier. There needs to be more work done in this field to detect/defend such attacks.

Works Cited

1. Xiao, C., Zhu, J., Li, B., He, W., Liu, M., & Song, D. (2018). Spatially transformed adversarial examples. *arXiv (Cornell University)*. http://export.arxiv.org/pdf/1801.02612

2. Goodfellow, I. J., Shlens, J., & Szegedy, C. (2014). Explaining and Harnessing Adversarial Examples. *arXiv (Cornell University)*. https://arxiv.org/pdf/1412.6572.pdf