Resisting Adversarial Perturbations Against Neural Networks *

Yuting "Summer" Yue†
yyue@seas.upenn.edu

December 5, 2017

Abstract

Neural networks have become pervasive in various domains. However, in industries such as finance [1], healthcare [2] and autonomous vehicles [3], insecure algorithms prone to adversarial perturbations can pose extreme security risks. In this paper, we analyze how overfitting, the process when the classifier corresponds too closely to a set of training data causing a worse overall performance, can have an effect on the classifier's robustness. We conclude that overfitting generally causes the classifier to become more prone to adversarial perturbations, and using regularization techniques can significantly improve the neural network's robustness. In addition, we demonstrate that extreme overfitting can lead to subpar performances on gradient-based attacks such as the commonly used Fast Gradient Sign Method (FGSM) [4]. The process of extreme overfitting has a similar effect as the defense method known as "Gradient Masking"[5].

1 Introduction

Advances in computational capacity and hardware enabled the application of neural networks across many industries. In healthcare, medical experts use deep neural networks on their patients' genetic information to make better diagnosis in certain diseases [6]. In finance, recurrent models performing financial series predictions now far surpass traditional regression models in precision [1].

However, neural networks can often be fooled in the presence of an adversary. For example, a facial recognition system used for the authentication can be fooled by an attacker to evade recognition or impersonate another individual [7]. Autonomous cars can be fooled by slightly modified stop signs which may lead to traffic accidents. If an algorithm that issues loans to people with eligible credit card history misclassifies non-eligible applicants, a large number of loans could be mistakenly issued to attackers.

Adversarial attacks against machine learning systems can be classified based on the attackers' goals [8]. The first type of attack aims to acquire part of the original dataset, model structures or parameters, all of which can be valuable intellectual property [2]. The

*The open-sourced experiments conducted in this paper can be found at its GitHub page

†This research paper is supervised by Sangdon Park (sangdonp@cis.upenn.edu), Prof.Insup Lee (lee@cis.upenn.edu) and Prof.James Weimer(weimerj@seas.upenn.edu).

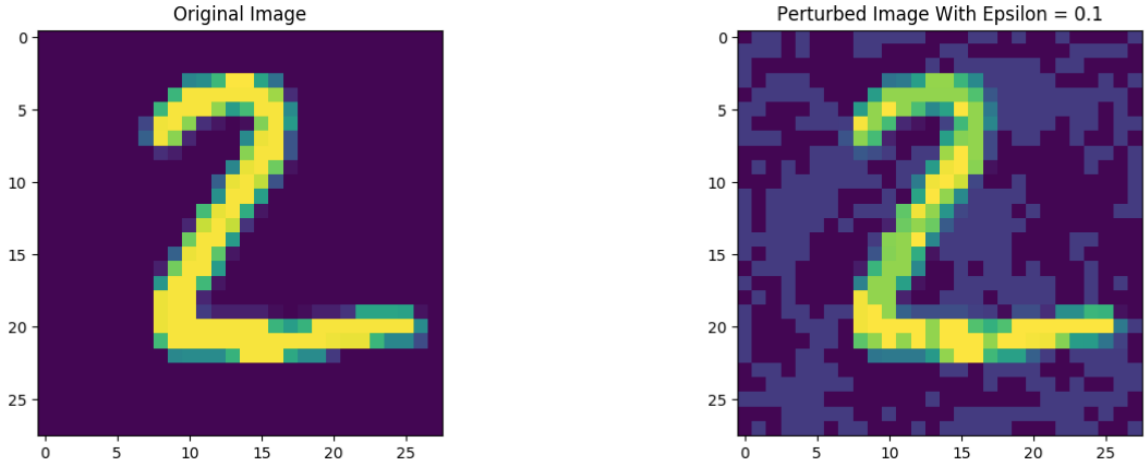

Figure 1: A demonstration of using FGSM on the original image on the left labeled as 2. After applying the FGSM attack with Epsilon = 0.1, the image looks like the one on the right. Humans would still think it is the number 2. However, our classifier misclassified the perturbed image on the right as the number 1.

second type of attack attempts to undermine the integrity of the algorithm, causing the algorithm to misclassify certain inputs in the test phase [8]. This paper only focuses on the second type of attacks, which we refer as "adversarial perturbations", assuming the original dataset and the target model architecture is known to the attacker.

As the examples mentioned above show, serious consequences can occur when a classification algorithm's performances are hindered. Therefore, it is important for computer scientists to harden neural network-based supervised learning algorithms to adversarial perturbations. To address this problem, numerous defense algorithms have been proposed. These techniques tend to sacrifice test accuracy as well as a large chunk of developing time, in order to achieve a higher robustness.

Regularization techniques used as a variant of adversarial training was developed by Goodfellow, et al [4]. By simply adding a term to the cost function related to the cost of perturbed examples, the neural network can become a lot more robust. Our experiments show that even a normal L2 regularization also has the effect of increasing robustness. This result leads to an interesting question, can we view the presence of adversarial examples as the result of traditional "overfitting", when the classifier becomes overly specialized in its training data? Because the classifier was overfitted on normal examples, examples that do not fall into the distribution in the training set, cannot be classified correctly. In this paper, we explore how overfitting on the training datasets and applying regularization techniques can influence a neural network's robustness.

From experiments conducted with the MNIST hand-written digits dataset, we concluded that overfitting on the training dataset would lead to a classifier that is prone to adversarial perturbations. On the other hand, the L2 regularization technique increases the robustness of a neural network drastically. From our initial experiments with the MNIST classifier, we also observed that in some special cases, the classifier becomes more robust towards a vanilla FGSM attack. We verified various hypothesis and concluded that: because the vanilla FGSM attack does not perturb certain parts of the image due to a 0-gradient, a higher accuracy after perturbation was achieved in our experiment. This

result illustrates that a classifier's robustness against a vanilla FGSM attack is not a good measurement of robustness.

2 Related Work

2.1 Attacks

Numerous forms of adversarial perturbation methods have been developed in the recent years.

1. In 2013, Szegedy's group found that deep neural networks learn input-output mappings that are fairly discontinuous to a significant extend. They developed a gradient-based method to cause the network to misclassify an image by applying a certain imperceptible perturbation[9]. The result from this paper inspired many others to develop adversarial perturbation methods and defense techniques.

2. In 2014, Goodfellow's group discovered that a simple linear model can have adversarial examples if its input has sufficient dimensionality. This discovery inspired them to invent an efficient attack algorithm called Fast Gradient Sign Method (FGSM)[4].This method works by perturbing the inputs to the direction of the sign of the gradients. FGSM saves a lot of computing power comparing to previous methods.

3. Papernot's group in 2015 proposed a forward derivatives method that allows adversaries to direct the mapping from input to output[10]. This gives informed adversaries greater control to adversaries with respect to the choice of perturbations.

4. In the following year, Papernot's group took advantage of the "transferability" property of adversarial perturbations and crafted a black box attack algorithm that can attack with only Oracle access to the classifier[5], generate perturbed inputs by attacking this new, and then attack the original classifier with the perturbed inputs.

2.2 Defenses

Many defense algorithms have also been developed in response to the potential attacks mentioned above.

1. "Adversarial training"[4] is a brute force technique that generates adversarial examples from the attacks, and add these examples as part of the training data. A variant of adversarial training adds a term to the original cost function that takes into account of costs for potential attacks.

2. Classifiers using the "Distillation"[11] technique outputs the probabilities of different classes, as opposed to the most likely class, making it more difficult for attackers to craft adversarial inputs.

3. "Gradient masking"[5] tries to deny attackers' access to useful gradients, causing the adversaries to not know how to perturb the inputs. It was shown to be not effective if the adversary trains the second classifier and conducts a black box attack.

4. Parseval Networks[12] develop relatively robust classifiers by capping the Lipschitz constant of linear, convolutional and aggregation layers is to be smaller than 1.

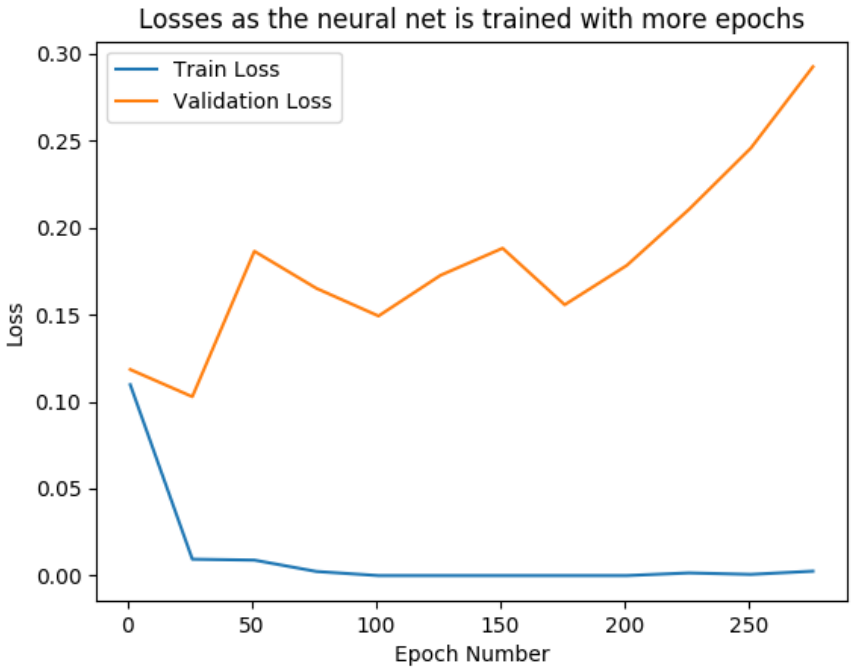

Figure 2: The graph displays the validation loss and the training loss while the classifier gets overfitted. We can see the overfitting happens around epoch 25.

5. Facebook AI research recently proposed another way to counter adversarial images by transforming inputs using methods including such as bit-depth reduction, JPEG compression, total variance minimization, and image quilting[13].

3 Project Overview

The thesis is comprised of two main parts that each produces its own set of conclusions:

1. Part I explores the impact of overfitting and regularization on the robustness of a classifier. Overfitting is the process when the classifier corresponds too closely to a set of training data causing a worse overall performance. The robustness refers to the neural network's ability to resist adversarial perturbations mentioned above. We conducted the experiments by observing our robustness measurements as we trained the classifier for more epochs or as we added regularization. The results indicate that overfitting on the training set leads to the presence of more adversarial examples, while regularization techniques help make the classifier more robust. This regularization techniques include adversarial regularization as well as L2 regularization.

   We made two major assumptions in our conclusions. First of all, we assumed that the robustness related to a FGSM attack is generalizable to other types of attacks. Second of all, we assumed that the results are not unique to the MNIST dataset. We proposed to replicate the experiments on more complex datasets such as ImageNet as our future research direction.

2. From experiments in Part I, we observed that in some special cases, the classifier becomes more robust towards a vanilla FGSM attack. Part II explores why this

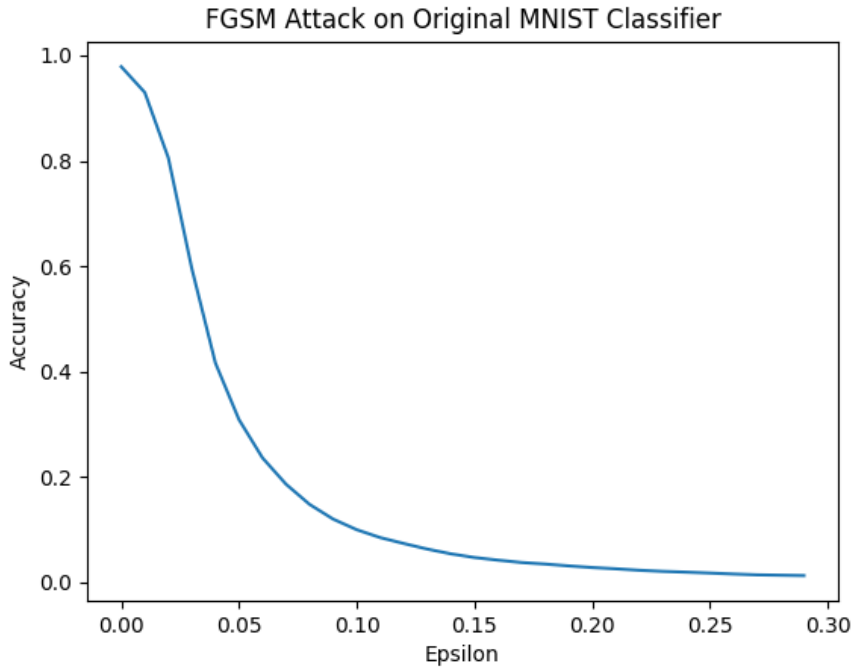

Figure 3: A direct visualization of the proposed robustness measurement based on the neural net's robustness towards the FGSM attack. The figure illustrates the robustness of the original 3-layer vanilla neural net without regularization

counterintuitive result occurred. We verified various hypothesis by conducting the experiments with a different optimizer, observing the changes in the spectral norms for their weights and observing the properties innate to the gradients.

We concluded the reason for the problem: because the vanilla FGSM attack does not perturb certain parts of the image due to a 0 gradient, a higher accuracy after perturbation occurred in our experiment. This result illustrates that, a classifier's robustness against a vanilla FGSM attack is not a good measurement of robustness.

4 Overfitting's Effect On Robustness

4.1 Classifier Implementation

To observe how internal structures may have an effect on the neural net's robustness, we implemented a basic neural net that classifies hand-written digits in the MNIST dataset. The MNIST database has a training set of 60,000 examples, and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image. An example of the input is shown as the left image in Figure 1.

The original neural network we implemented contains two hidden layers and a final Softmax layer. We used the default Xavier initializers for weights and zero initializers for biases for each layer. The activation function used was ReLU, for a faster training time due to our computational resources constraints. The layers each contain 200, 300 and 10 neurons. We used the Adam Optimizer for training and minimized the cross-entropy loss. The parameters used for the optimizer are beta1=0.9, beta2=0.999, epsilon=1e-08. The

learning rate used was 0.001. We were able to produce a 98.0% test accuracy through this simple neural net. Although it was not state of art accuracy on the MNIST dataset, we decided that this classifier would be accurate enough for our experiment's purposes.

4.2 FGSM as Robustness Measurement

Next, we invented an intuitive way for comparing the robustness for our original classifier and the neural networks with modified structures.

The Fast Gradient Sign Method, as mentioned in the "Related Work" session, has become one of the most popular attack methods due to its efficiency and effectiveness. Figure 1 illustrates a resulting image generated using an FGSM attack that leads to misclassification. In the FGSM method, an epsilon variable is used to indicate the extent of the perturbation enforced on the original images. A higher epsilon is associated with a higher perturbation. As we gradually increase the extent of perturbation, we should expect to see a decrease in accuracy after perturbation. We can visualize this effect through a line graph, with the x axis being the value of epsilon, and y axis being accuracy after perturbation. An example is shown in Figure 3 . From the line graph described above, we can easily compare the robustness of two distinct neural networks. For the same epsilon, a higher perturbed accuracy would mean a higher robustness. A very robust network would have a slow drop in perturbed accuracy as the value of epsilon increases.

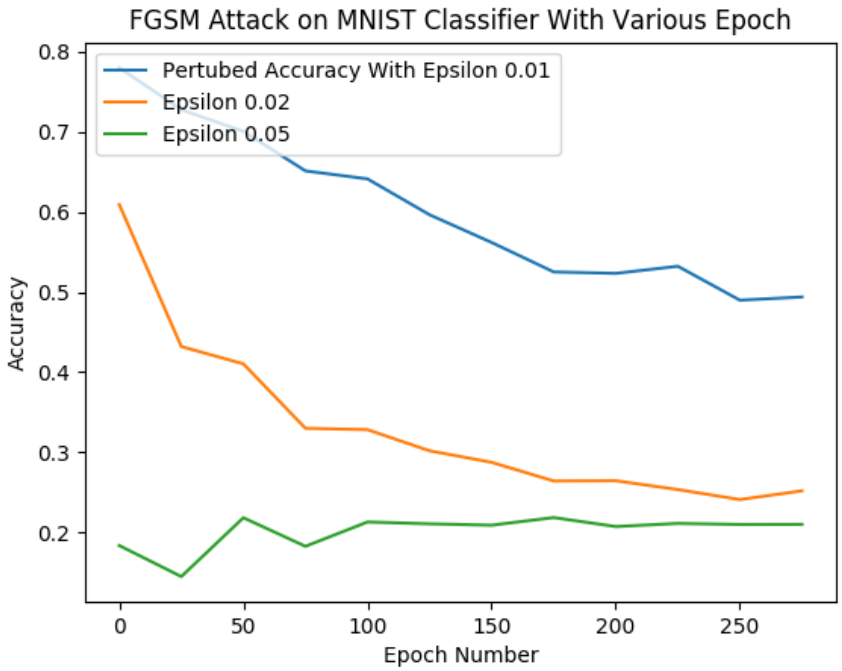

Figure 4: A graph demonstrating the accuracy on perturbed inputs with various epsilons as our classifier starts to overfit. We can observe the general trend: as overfitting occurs, the classifier becomes less robust.

4.3 Hypothesis

Before running any experiments, we came up with our hypothesis: adversarial examples are caused by a classifier overfitting on the training data. As training time goes on, the

decision boundary, in theory becomes more nonlinear. The boundaries become closer to the data points in a high dimensional space. As a result, we should expect to observe a decrease in robustness as we overfit the neural network.
If the hypothesis we made were true, we should expect the behaviors listed below:

1. As we train the neural net for more epochs after it reaches the highest validation accuracy, the spectral norms for the weights on each layer increase, causing the neural net to become more prone to adversarial perturbations.

2. Since regularization normally diminishes the effect of overfitting, adding L2 Regularization to the original classifier's cost function will make the neural network more robust.

4.4 Experiments and Results

1. We compared the performances and robustness of the original classifier and some overfitted classifiers. The overfitted classifiers were generated by keeping on training after an optimal validation accuracy was reached. Afterwards, the validation loss started going up while the training loss kept going down. We drew a global view of the robustness. As Figure4 shows, as we overfitted the classifier, the perturbed accuracy goes down, meaning the classifier becomes less robust over time. We repeated the experiments for various epsilons and achieved the same results.

2. We also illustrated the effect of L2 regularization on a classifier's robustness. We used our original classifier and applied L2 loss to the loss function with a weight of 0.001. Figure5 demonstrates a comparison between the robustness of the original classifier and the regularized classifier. The regularized classifier is significantly more robust than the original classifier according to our metric.

4.5 Conclusions

1. As the classifier is overfitted, the robustness according to our measurements goes down. The classifier becomes more prone to adversarial examples.

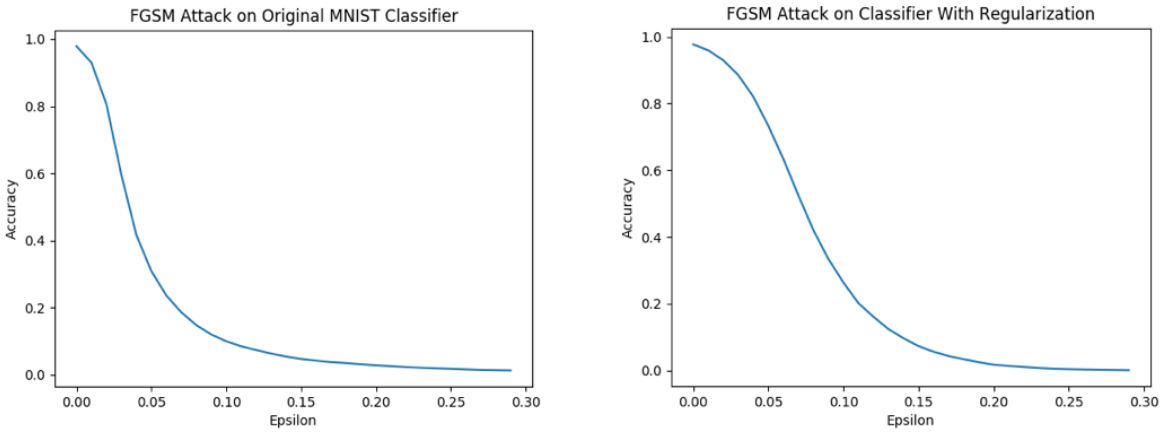

Figure 5: The graph on the left illustrates the robustness of the original classifier. The graph on the right illustrates the robustness of the L2 regularized classifier trained on the same inputs. Both classifiers have a similar test accuracy, but as shown in the graph, the regularized classifier is more robust.

2. The L2 regularized classifier is significantly more robust than the original classifier according to our metric.

5 Special Case: Overfit A Robust Classifier

5.1 Counterintuitive Problem

In our initial experiments, we observed a counter-intuitive result. As shown in Figure 6, after training the classifier for more than a 50 epochs, more training would cause the neural network to become more robust according to our FGSM based metric. This result directly conflicts with our conclusion in Part I.

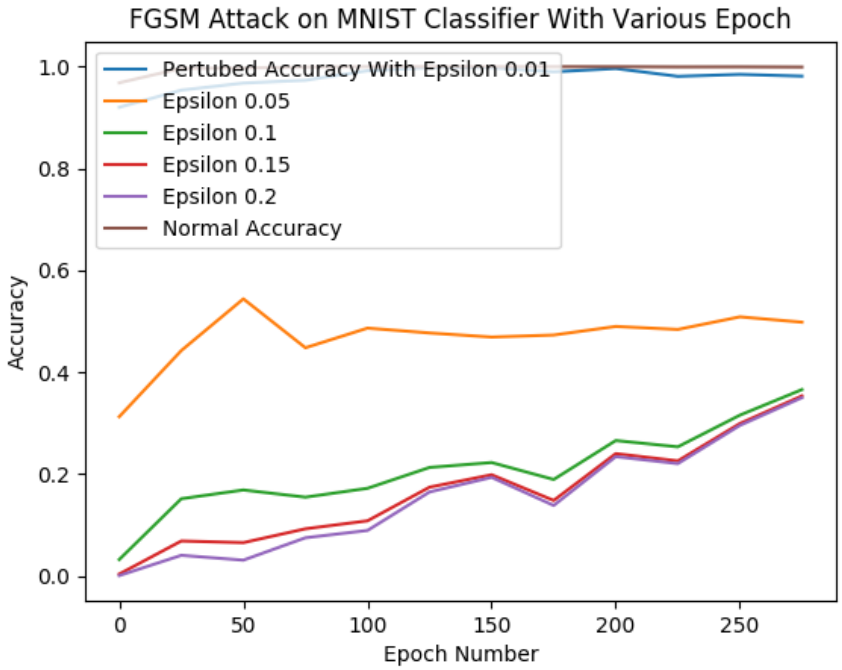

Figure 6: This graph demonstrates the counterintuitive result when overfitting caused a classifier to become more robust. The experiments were repeated using different epsilons and the trends still persisted.

5.2 Hypothesis, Experiments and Results

1. We suspected the weights of the classifier went down as we started overfitting due to the nature of the MNIST dataset. It is a possibility that the weights fluctuations accompanied by overfitting made the classifier more resistant to attacks. We graphed the spectral norm for weights for each layer and discovered that they all go up as epochs go on. Therefore, we discard this hypothesis.

2. We suspected that the result may have something to do with the AdamOptimizer we used. It could have some regularization effect that we are not aware of. Therefore, we repeated the experiment using the SGD optimizer and achieved the same trends, but lower effect size due to a slower training process. We discarded this hypothesis as well.

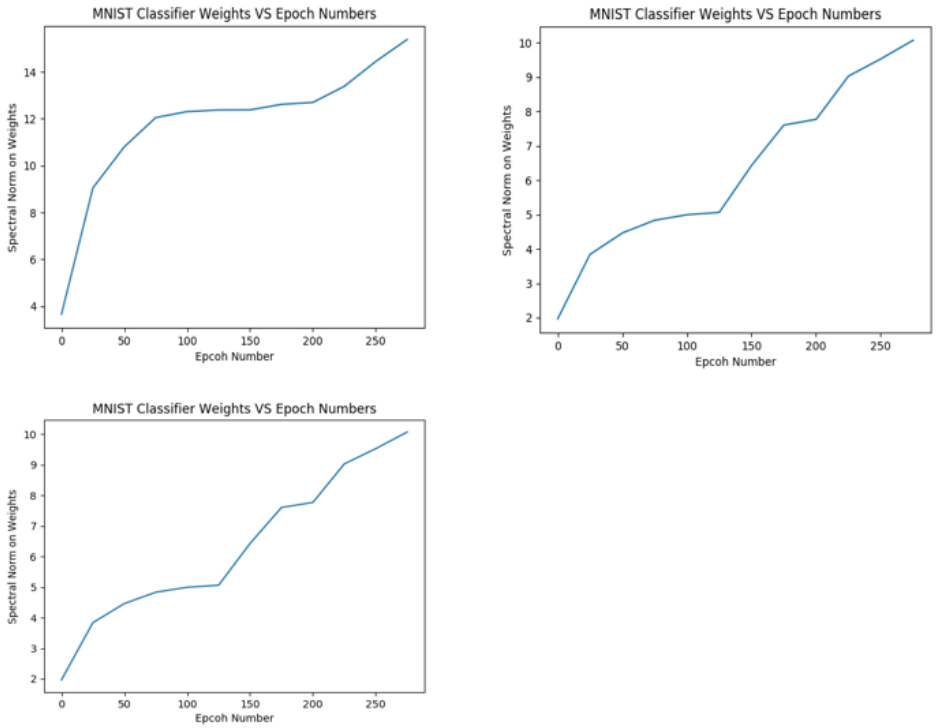

Figure 7: We observed the spectral norms of weights for each layer of the neural net. As we trained the classifier for more epochs, the spectral norms for the weights for each layer increased.

3. We suspected the unexpected result has something to do with the FGSM attack itself. By taking the sign of the gradients, the attack only happens when the gradients are not 0s. In reality, as we overfit, the gradients become closer and closer to 0, to a point that with the float32 precision, Tensorflow thinks it's 0. Not knowing which directions the inputs should be perturbed, FGSM would not perturb some parts of the image at all. This has the same effect as "gradient masking", when the developers of the classifier hide the gradients from the attackers. We verified the hypothesis by printing out the number of zeros in the gradients. The number of zeros in the gradients is almost perfectly inversely correlated with the perturbed accuracy. We changed the float types in the neural net to float64 and conducted the experiments again. The unexpected result disappears. We now observe a decrease in robustness as we overfit, consistent with our conclusions in Part I.

5.3 Conclusions

The counterintuitive result was caused by the increase in the number of 0s in the classifier's gradients. When the number of 0s increases, FGSM attacks stop applying perturbations partially. Therefore, the classifier becomes more resistant to gradient-based attacks such as FGSM.

6 Summary

The paper explores the impact of overfitting and regularization on the robustness of a classifier. Results from the MNIST classifiers from our experiment indicate that overfitting

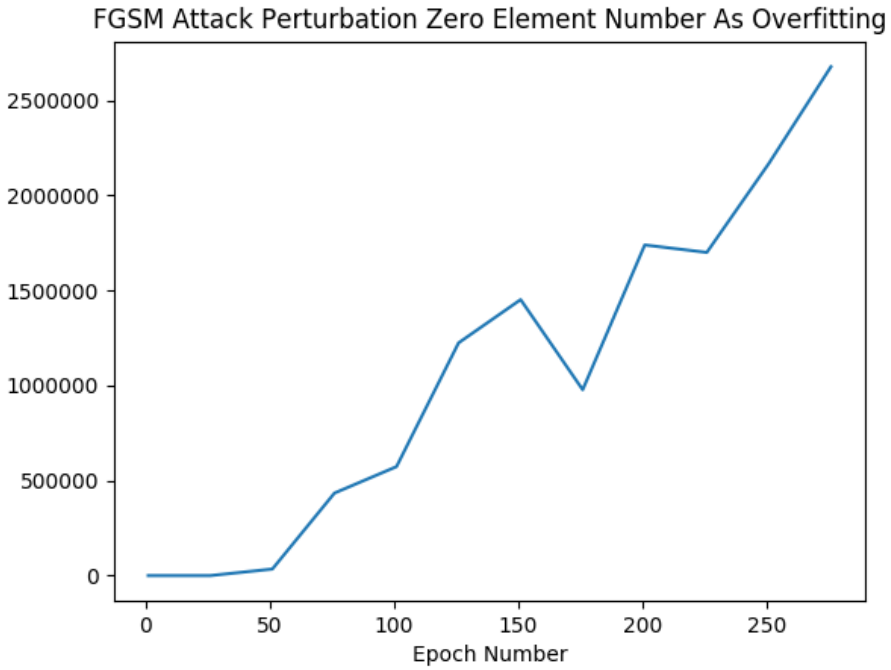

Figure 8: We observed the number of 0s (Y axis) in the gradients when conducting the FGSM attack. The number of 0s increases as we trained for more epochs. We discovered that the trend for the number of 0s is highly correlated with the classifier's robustness.

leads to the a less robust classifier, and regularization makes the classifier more robust. We also discovered that FGSM attack stops parts of the perturbations as a classifier becomes "very overfitted". At places where there is a 0 gradient, a higher accuracy after perturbation occurred in our experiment. This result illustrates that a classifier's robustness against a vanilla FGSM attack is not a good measurement of robustness.

7 Future Research Directions

It will be useful to replicate the experiments on a larger data set such as ImageNet in addition to MNIST. In the MNIST classifier, the effect of overfitting is subtle to observe. As the validation loss starts to go down gradually, the validation accuracy still stays very flat. The results would be more convincing after similar trends are observed in a larger dataset and some more complicated classifiers as well.

Parseval network is a defense algorithm by capping the spectral norm of the weights in each layer[12]. From our experiments, there seem to be a correlation between a classifier's robustness and the number of neurons per layer, the number of layers and the use of regularization in addition to the weights on the neurons. Potential more effective defense algorithm may be crafted by looking deeper into how the combination of these variables is related to robustness.

More research should be done to analyze how different regularization techniques may impact robustness differently. Current targeted defense techniques such as adversarial training might sacrifice test accuracy and consume a large chunk of time during development. For developers, it would be helpful to know what kind of normal regularization techniques are more robust, in order to produce algorithms less prone to adversaries,

without deliberately defending against adversarial examples.

Appendix

A Randomly Perturbing At 0 Gradient

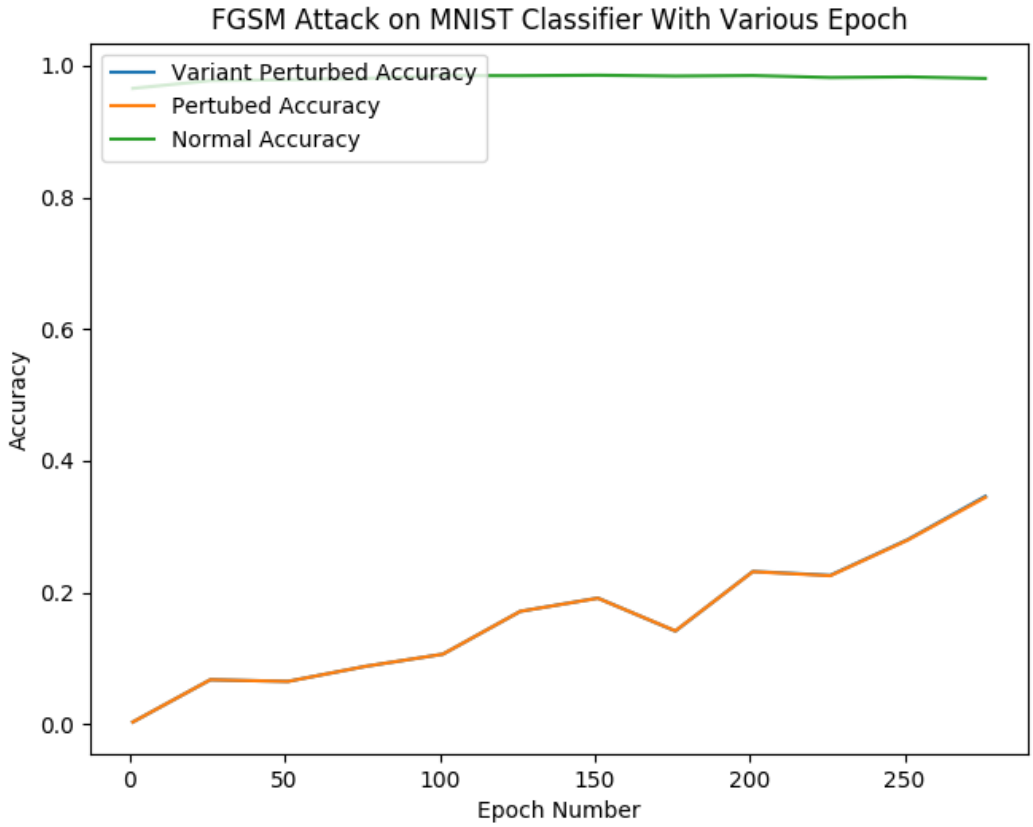

Figure 9: We demonstrate the difference between the loss inaccuracy caused by traditional overfitting and adversarial examples. For the 0 gradient spots mentioned in Part II, we randomly perturbed the spots by changing the gradient to -1 and 1 randomly. We noticed that the random perturbation had no effect on accuracy, because the lines overlap. This is a different behavior from traditional overfitting. In traditional overfitting, if you add noise to the test data, the accuracy is mostly likely to go down. Adversarial examples require the perturbation to be in a very specific direction, the direction indicated by the gradient.

B Robustness For The Classifier After Adversarial Training

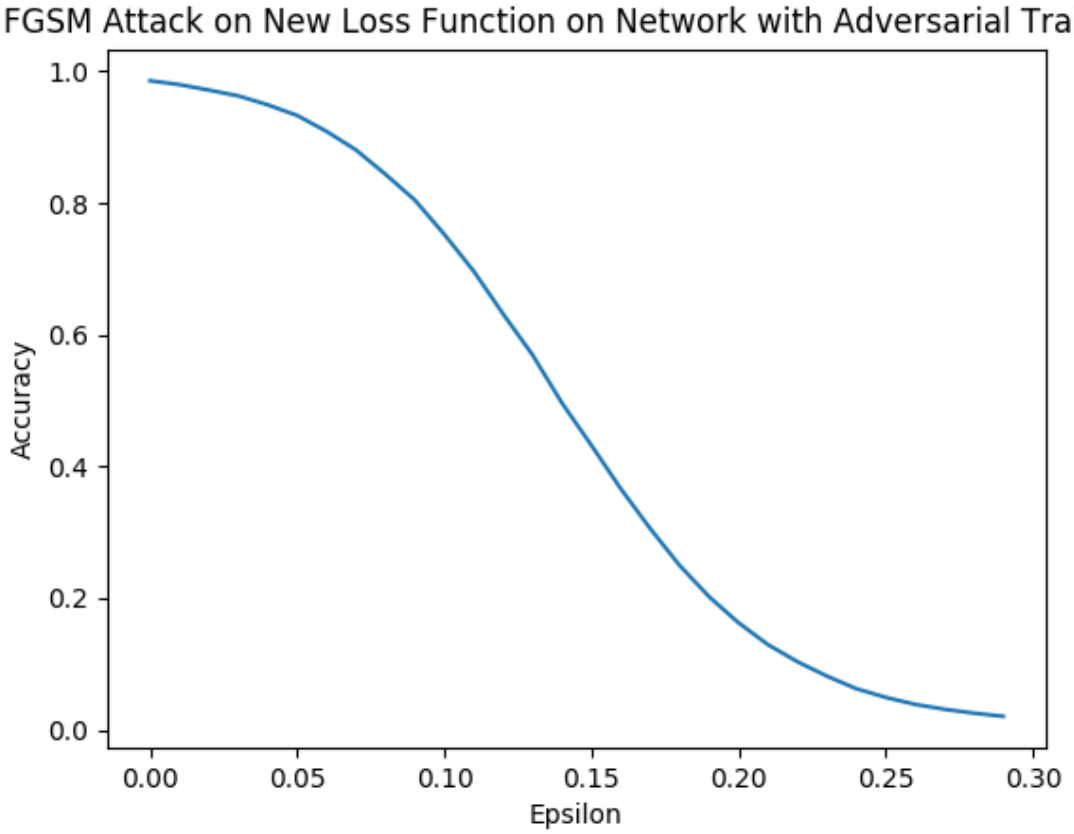

Figure 10: We applied adversarial training to the original classifier, as a comparison to a L2 regularized classifier. As expected, the adversarially trained classifier is more robust.

C Code Snippet For Our Neural Net's Structure

```python
def build_network(x, l1_units, l2_units, l3_units):
    """Build the MNIST model with 2 hidden layers and one linear layer.
    params:
        x: input placeholder
    Returns:
        Output tensor with the computed logits
    """

    # Hidden 1
    with tf.variable_scope("layer1"):
        w1 = tf.get_variable("w",[input_dimension, l1_units], initializer = tf.contrib.layers.xavier_initializer(seed = 1))
        b1 = tf.get_variable("b", [l1_units], initializer = tf.zeros_initializer())
        z1 = tf.matmul(x, w1) + b1
        y1 = tf.nn.relu(z1)

    # Hidden 2
    with tf.variable_scope("layer2"):
        w2 = tf.get_variable("w",[l1_units, l2_units], initializer = tf.contrib.layers.xavier_initializer(seed = 1))
        b2 = tf.get_variable("b", [l2_units], initializer = tf.zeros_initializer())
        z2 = tf.matmul(y1, w2) + b2
        y2 = tf.nn.relu(z2)

    # softmax
    with tf.variable_scope("layer3"):
        w3 = tf.get_variable("w",[l2_units, l3_units], initializer = tf.contrib.layers.xavier_initializer(seed = 1))
        b3 = tf.get_variable("b", [l3_units], initializer = tf.zeros_initializer())
        z3 = tf.matmul(y2, w3) + b3
        y_ = tf.nn.softmax(z3)

    return z3, y_ # logits and probilities
```

Figure 11: The code snippet for the original neural net's structure in Tensorflow.

D Transfer Attack From Float64 To Float32

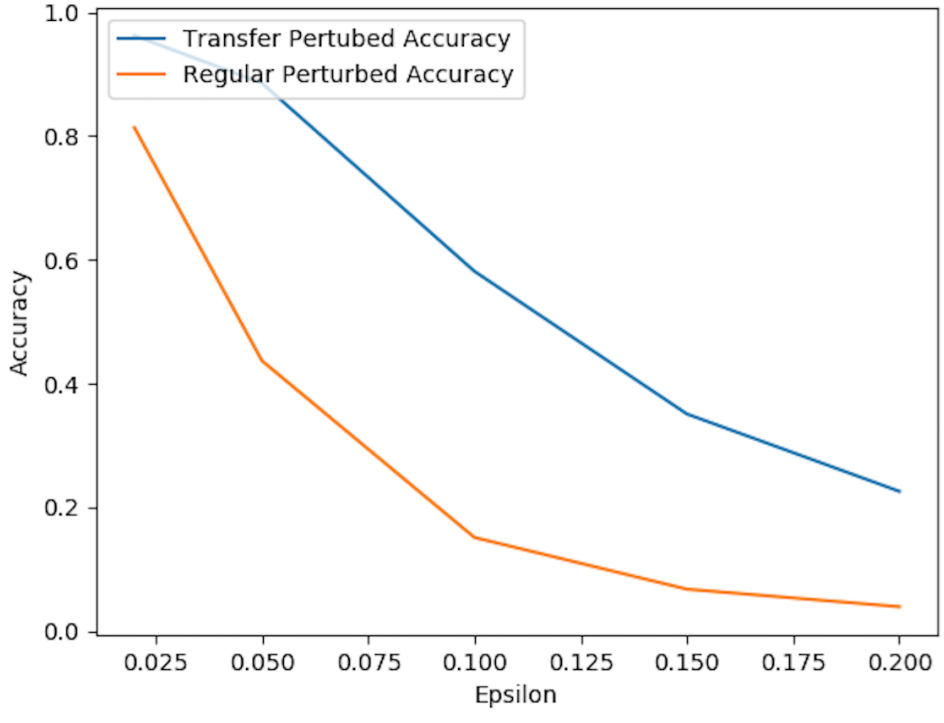

Figure 12: We applied transfer training from a float64 classifier to a float32 classifier, because an identical float32 classifier should be more robust to FGSM attack. However, the gap caused by the transfer process was too large to make the precision difference significant.

References

[1] Xin-Yao Qian and Shan Gao. Financial series prediction: Comparison between precision of time series models and machine learning methods. *arXiv preprint arXiv:1706.00948*, 2017.

[2] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1322–1333. ACM, 2015.

[3] Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging*, 2017(19):70–76, 2017.

[4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[5] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against deep learning systems using adversarial examples. *arXiv preprint arXiv:1602.02697*, 2016.

[6] Shandian Zhe, Zenglin Xu, and Yuan Qi. Supervised heterogeneous multiview learning for joint association study and disease diagnosis. *arXiv preprint arXiv:1304.7284*, 2013.

[7] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1528–1540. ACM, 2016.

[8] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael Wellman. Towards the science of security and privacy in machine learning. *arXiv preprint arXiv:1611.03814*, 2016.

[9] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[10] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. *CoRR*, abs/1511.07528, 2015.

[11] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[12] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning*, pages 854–863, 2017.

[13] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.