An Experiment of Adversarial Exmaples Crafting and Adversarial Training on Various Datasets

Jian Li Mingyue Yang David Lie

December 7, 2017

Abstract

In this report, we performed adversarial example crafting and adversarial training on various datasets and evaluated their performance.

1 Introduction

Goodfellow et al. and Papernot et. al. introduced several algorithms of crafting *adversarial examples.* These algorithms include *Fast Gradient Sign Method* (FGSM)[1] and *Jacobian-based Saliency Map Method* (JSMA)[2]. We exercised these two adversarial crafting methods on several datasets to find out the relationships between dataset type and performance of the two algorithms.

We constructed several CNNs with TensorFlow[11] to recognize several datasets. Then we modified the CNN model to apply adversarial algorithms (implemented by CleverHans[10]). We generated adversarial examples with both algorithms and evaluated their success rate for both misclassification and targeted attacks. Then we retrained the model with adversarial training, and regenerated adversarial examples to evaluate the performance of the adversarial training. Due to limited time frame, we were not able to experiment adversarial training with Saliency Map method.

2 Datasets

We crafted adversarial examples using different settings on the datasets shown in Table 1 below.

3 Adversarial Examples Crafting

We used implementation of the fast gradient sign method and saliency map method from the CleverHans[10] to craft adversarial examples for datasets CGTSRB10, CIFAR10, FMNIST, and MNISTBG. For comparison, we also did the same experiment on original MNIST dataset.

3.1 Adversarial Examples Crafting with FGSM and JSMA

3.1.1 CNN Models

We trained a convolutional neural network model with TensorFlow[11] for each dataset. The legitimate test accuracy is shown in Table 2.

Dataset	Descriptions
GTSRB43	An image collection consisting of 43 traffic signs captured from real life. Every traffic sign has a serial of images with different resolutions. (32×32 color images)
CGTSRB10	A subset of the original GTSRB[5] dataset with 10 selected types of traffic signs. (28×28 grey-scale images)
CIFAR10	A collection of real-life objects[4], containing 60000 samples in 10 classes. (32×32 color images)
MNIST	Original MNIST[8] dataset with 10 handwritten digits. (28x28 grey-scale images)
FMNIST	Fashion MNIST[6] is a MNIST-like dataset that consists of 10 types of fashion objects such as skirts and shirts. It consists of a training set of 60000 examples and a test set of 10000 examples. (28×28 grey-scale images)
MNISTBG	MNIST with background)[7] is a patched version of MNIST dataset. Each MNIST example is attached with a background image extracted randomly from a set of 20 images downloaded from the Internet. (28×28 grey-scale images)
SVHN	Cropped street sign numbers with real-life backgrounds[9]. (32x32 color images)

Table 1: List of Datasets Used

Dataset	Test accuracy
CGTSRB10	97%
CIFAR10	68%
FMNIST	92%
MNISTBG	92%
MNIST	99%

Table 2: CNN test accuracy with legitimate examples

3.1.2 Fast Gradient Sign Method

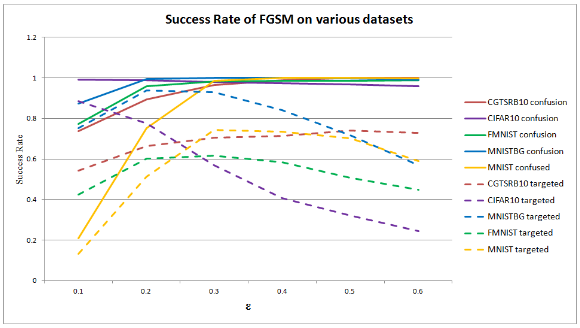
For fast gradient sign method, we used 5000 examples from the test set of each dataset (except for CGTSRB10 which has only 3360 test examples) and generated adversarial examples with $\epsilon = 0.1 \sim 0.6$. For each example, first a random label different than the example's original label is selected as attack target. Then the FGSM algorithm is executed to generate an adversarial example. Finally, the generated adversarial example is fed into the original CNN model to get a prediction, and the prediction is compared with its original label and the attack target. If the prediction is identical with attack target then it is a successful targeted attack, otherwise if the prediction is not the same as the original label, its a successful misclassification.

Figure 1a shows the result of success rate for both misclassification and targeted attacks.

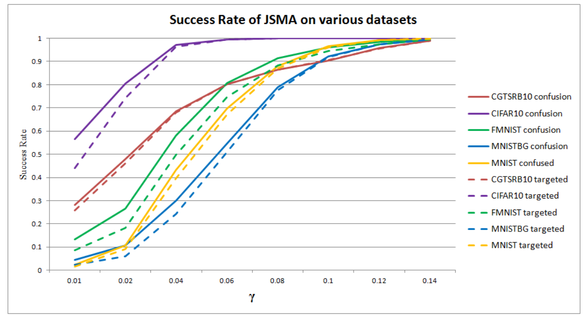
3.1.3 Saliency Map Method

For saliency map method, we used the same 5000 examples to generate adversarial examples. Our plan was to evaluate the performance of different γ parameters. However, since γ controls the number of iterations, it is not necessary to repeat the execution on every variation of γ. Therefore we crafted the adversarial examples with $\gamma = 0.15$ while recording the iteration count when misclassification and targeted attack succeed with a slightly modified JSMA algorithm.

After the adversarial examples are generated, we calculated the success rate for each planned γ value with the data recorded during the crafting. Figure 1b shows the result of success rate.



(a) FGSM



(b) JSMA

Figure 1: Success rate of JSMA and FGSM on various datasets.

Dataset	Test accuracy
CGTSRB10	98%
FMNIST	99%
MNIST	98%
SVHN	97%

Table 3: CNN test accuracy with legitimate examples

3.1.4 Observation

FGSM can easily confuse CNN model to misclassify an image by altering all the pixels by ϵ. For all the datasets, $\epsilon = 0.2$ can easily achieve 80% success rate on a naïve CNN model. Practically $\epsilon = 0.3$ is easily detectable by human eyes.

However the targeted attacks did not show a good success rate. In most case, targeted success rate starts to fall when $\epsilon \geq 0.3$. Intuitively when $\epsilon \geq 0.3$ the perturbation is enough to confuse human to make wrong classification, which results in low success rate for targeted attack. However the reason still requires further investigation.

The CNN model for CIFAR10 dataset is easily attacked (misclassification rate $\approx$ 100% when $\epsilon = 0.1$), which is considered the result of its low accuracy (68%) on legitimate examples.

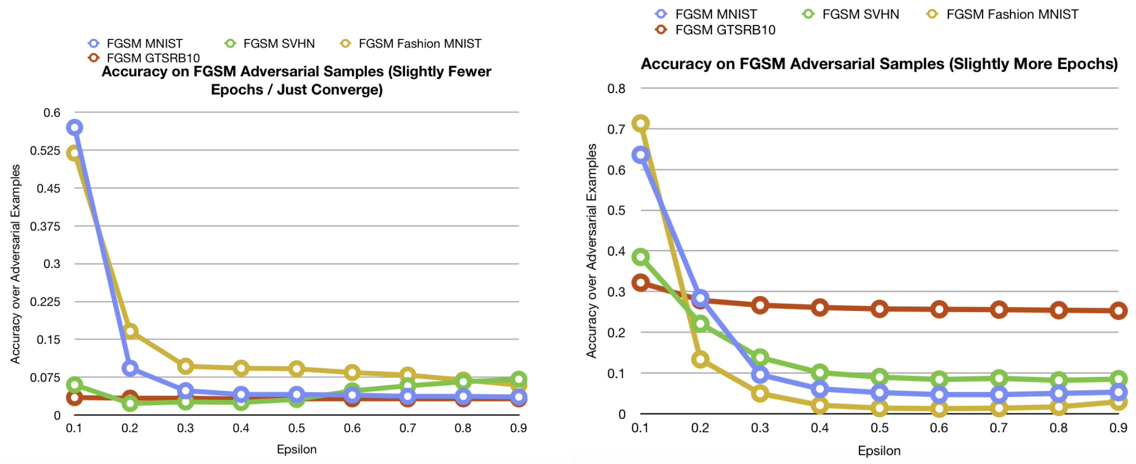
JSMA gave a good success rate on both misclassification and targeted attacks. For most datasets, $\gamma = 0.14$ is enough to guarantee 95% success rate against a naïve model.

3.2 Effect of Number of Clean Training Epochs on FGSM Whitebox Attack

To evaluate the effect of number of epoches on FGSM whitebox attack, experiments are conducted based on CNN models listed in Table 3.

FGSM attacks are first performed on models that are cleanly trained with slightly fewer epochs. For comparison, the FGSM attacks are again performed on models that are cleanly trained with slightly more epochs. The accuracy of the 4 models show a general trend. When the ϵ becomes larger, models for all the 4 datasets show lower accuracy for the FGSM adversarial attack. It can also be observed that models trained with slightly more epochs are more resistant to attacks with small ϵ values than their underfitting counterparts.

In Figure 2b, ϵ has little effect on the GTSRB10 model. For further inspection, adversarial attacks with smaller ϵ values are performed on cleanly trained GTSRB10 models with different number of training epochs. From Figure 3a, it can be seen that the accuracies of the GTSRB10 models fall



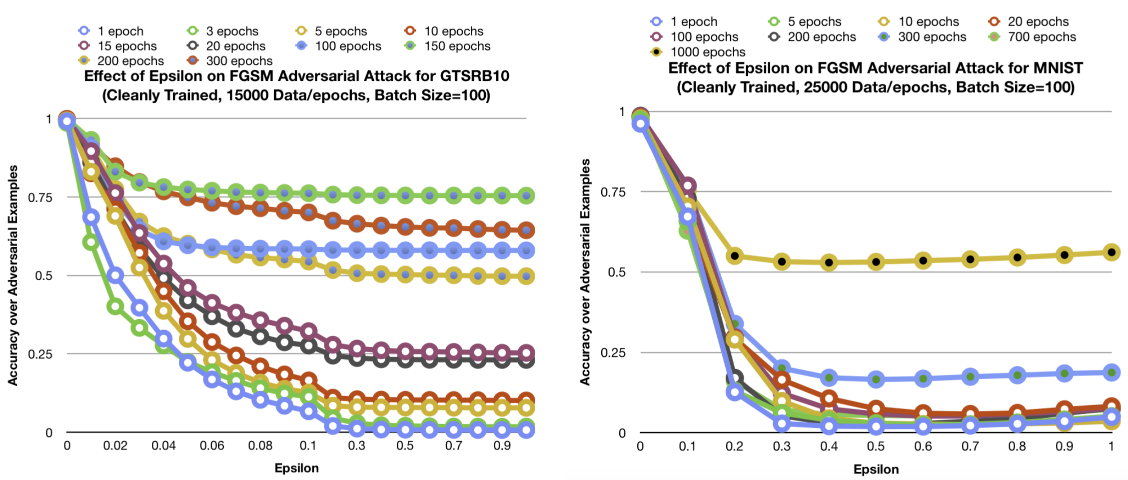
(a) Slightly Fewer Epochs (b) Slightly More Epochs

Figure 2: Effect of Number of Epoches on FGSM whitebox attack

abruptly with small initial changes in ϵ (0.01 to 0.09). The accuracy of the GTSRB10 models start to converge when ϵ reaches 0.3.

For comparison, similar experiments are also conducted on the MNIST dataset, as shown in Figure 3b. In both of the figures, there is a general trend that models trained with more number of epochs are more resistant to FGSM adversarial attacks. Under this general trend, there are also some suboptimal values for number of epochs: while the accuracy generally becomes better as the number of epochs increases, the accuracy lines may fluctuate in a certain range when the number of epochs increases.

It is also interesting that by merely training models on clean data for large number of epochs, models can gain ability to defend against adversarial attacks with large ϵ values. Comparing to models trained with other datasets, this effect is especially obvious for GTSRB10. Further studies on model complexity and dataset complexity may be conducted to investigate why these GTSRB10 models require fewer epochs to defend attacks with large ϵ.



(a) GTSRB10 (b) MNIST

Figure 3: Effect of number of epoches on Accuracy over Adversarial Examples

3.3 Effect of Dataset on FGSM Whitebox Attack and Blackbox Attack

From the above figures, it can be seen that comparing to models trained with datasets containing noisy backgrounds (GTSRB10 and SVHN), models trained on dataset with clear background (MNIST and FMNIST) are more sustainable to adversarial attacks with small ϵ values. For models trained with both slightly more and slightly fewer epochs, MNIST and FMNIST datasets have better accuracy over adversarial samples with small ϵ.

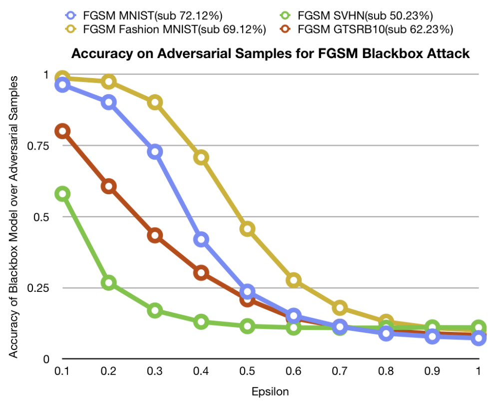


Figure 4: FGSM Blackbox Attack on Different Datasets

Then, the FGSM black-box attacks are performed on models trained with all the 4 datasets (accuracy of models are shown in Table 3). Although we used substitute models with higher accuracy for the MNIST and FMNIST datasets, their accuracy on the adversarial samples are still generally higher than the SVHN and GTSRB10 model. For MNIST and FMNIST models, the FGSM black-box attacks are hardly successful with small ϵ. As ϵ increases, the accuracy of all the 4 models decrease in a smooth curvy trend. The accuracy over adversarial samples converge to around 10% as the ϵ value approaches its maximum. This again demonstrates that models trained with simple dataset that have clean backgrounds are more robust against the FGSM attacks with small ϵ values.

4 Adversarial Training with FGSM

Adversarial training is introduced by Goodfellow et al.[1]. The fundimental idea is to use adversarial examples as a regularizer:

$$\tilde{J}(\theta, x, y) = \alpha J(\theta, x, y) + (1-\alpha) J(\theta, x + \epsilon sign(\nabla_x J(\theta, x, y)))$$

4.1 Effect of Adversarial Training

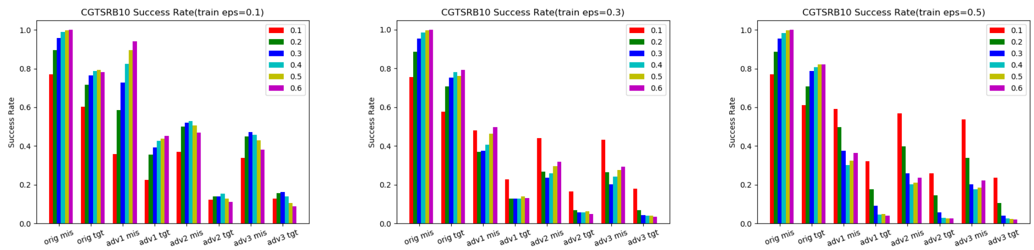
In the following experiments we used $\alpha = 0.5$. First, we train CNN models with legitimate training data to get a naïve mode. The test accuracy of these models are listed in Table 2. Then we craft adversarial examples with test data and evaluate the misclassification and targeted success rate. Next, we do adversarial training three times and regenerate adversarial examples and re-evaluate the success rate after each training session. We repeated this process with $\alpha = 0.1 \sim 0.6$ on all the datasets. Due to space limit, we only picked some of the results (Figure 5), which however could demostrate some interesting facts.

Initially we expected the adversarial training could produce a more robust model that could defense the FGSM attacks. However we found two interesting facts. Denote the parameter used in adversarial training by ϵ and the parameter used in following adversarial example crafting by $\tilde{\epsilon}$.

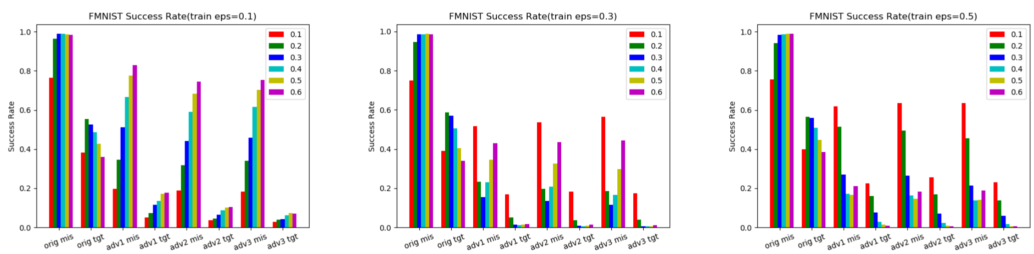
First, a lower ϵ ($\epsilon = 0.1$) does not provide too much defense against higher $\tilde{\epsilon}$. Especially in the FMNIST dataset, even after several iterations of adversarial training, $\tilde{\epsilon} = 0.6$ can still achieve approximately 80% misclassification rate.

Second, a higher ϵ does not provide any defense on lower $\tilde{\epsilon}$. When $\epsilon = 0.3$, although adversarial examples crafted with $\tilde{\epsilon} = 0.3$ can only make less than 10% success rate, adversarial examples crafted with $\tilde{\epsilon} = 0.1$ has higher success rate than $\tilde{\epsilon} = 0.3$ and will not decrease significantly. The similar effect can be seen in $\epsilon = 0.5$.

This indicates that adversarial training with various $\tilde{\epsilon}$ may be required in order to train a robust model, however the feasibility is still an open question.



(a) CGTSRB $\epsilon = 0.1$ (b) CGTSRB $\epsilon = 0.3$ (c) CGTSRB $\epsilon = 0.5$



(d) FMNIST $\epsilon = 0.1$ (e) FMNIST $\epsilon = 0.3$ (f) FMNIST $\epsilon = 0.5$

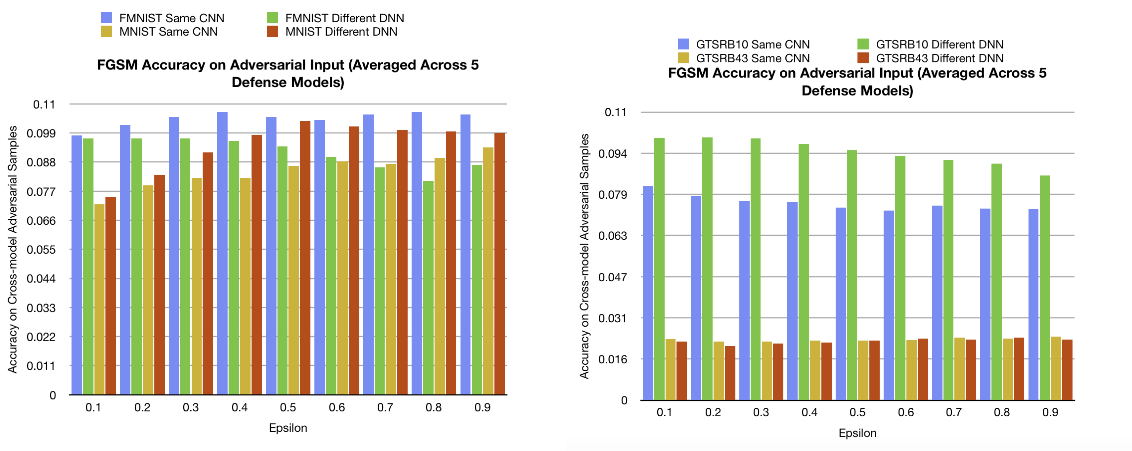
Figure 5: Success Rate of FGMS with adversarial training. Bars $0.1 \sim 0.6$ are $\tilde{\epsilon}$ used in adversarial example crafting.

4.2 Cross-model FGSM Attacks for Adversarial Training

Cross-model attacks are also performed for models with adversarial training. For each dataset, the results of cross-model attack are averaged over 5 separate CNN models with ϵ=0.5. The details of the 5 models used for each dataset is shown in Table 4 in Appendix. The FGSM attack first use samples constructed from a CNN model with the same architecture as the attacked models but different initialization parameters. Then the adversarial samples are constructed from a DNN model, which has different architecture from the attacked models. The results are shown in Figure 6a and Figure 6b.

Figure 6b shows how GTSRB with different number of classes perform for cross-model attacks. The GTSRB10 dataset has much higher accuracy than the GTSRB43 dataset which includes all 43 classes. This result is intuitive: with more classes, it is easier to reach a close class boundary that causes the model to misclassify.

From Figure 6a and Figure 6b, it can be seen that the different architecture and ϵ used to construct the adversarial samples do not have much effect on models accuracy. Although there are trends for specific dataset types, no general pattern can be observed across datasets. Further exploration may be required to figure out patterns.



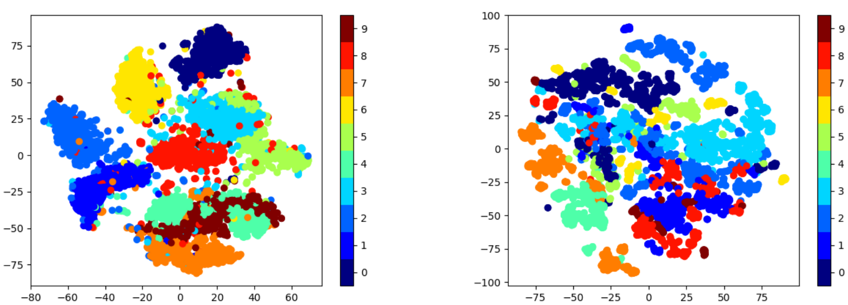
(a) MNIST & FMNIST

(b) GTSRB43 & GTSRB10

Figure 6: Cross-model FGSM Attacks for Adversarial Training

This shows that simple adversarial training, although gives some protection, is still vulnerable to cross-model attacks.

5 Dataset Distribution

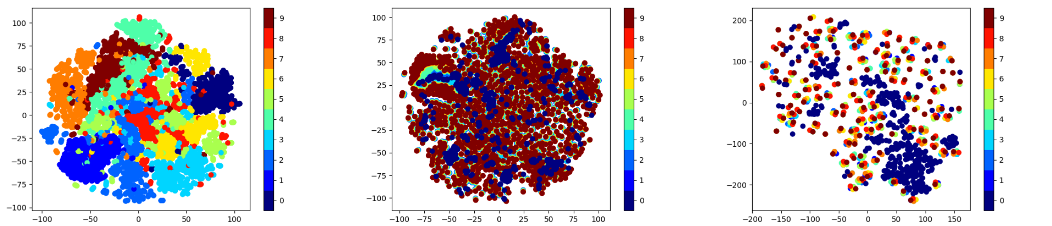


(a) MNIST

(b) GTSRB10

Figure 7: T-SNE Graphs on MNIST and GTSRB10 Datasets

To investigate why dataset type can influence accuracy of model over adversarial attacks, distribution of datasets is studied. The T-SNE algorithm is performed on all datasets to obtain a 2D view for the high-dimensional data points. T-SNE graphs for both original clean data in the MNIST and the GTSRB10 datasets are shown in Figure 7. From the graphs, it can be observed that the MNIST dataset have more easily distinguishable clusters with clear boundaries, while the clusters for the GTSRB10 dataset are more scattered. This is due to the fact that the MNIST dataset are simpler images with clear background, while the GTSRB10 dataset has a noisy background. This result also holds for other datasets that have clear/noisy background. The complexity and lack of clear boundary may contribute to the fact that of datasets with noisy backgrounds are more susceptible to attacks with small ϵ values.



(a) Clean & Adversarial Samples Classifed by Original Class (b) Clean & Adversarial Samples from All Classes, Classified by ϵ (c) Clean & Adversarial Samples from Single Class, Classified by ϵ

Figure 8: T-SNE Graphs of Adversarial and Clean Samples from MNIST

To show how the adversarial samples and original clean samples are different, the T-SNE algorithm is again performed on both original samples and FGSM adversarial samples of the MNIST dataset. For data labelled by their original class, data points in same original class form distinguishable clusters. For data labelled using ϵ values from 0 0.9, adversarial data points all scatter over the 2D space, while original clean data form many small clusters in the 2D space. The boundary between the adversarial and clean data points is not as clear.

This shows that although the FGSM adversarial samples are able to fool the attacked models, from the T-SNE perspective, FGSM adversarial samples can still be distinguished by their original class, and the common patterns in adversarial samples are not as obvious as the common patterns in same original classes. The fact that adversarial samples with varied epsilon value can still be distinguished by their original class gives an explanation the argument in Section 3.2 that merely doing more training on clean data can give models protection over adversarial attacks.

For further investigation, a specific MNIST class is picked, and T-SNE is performed on both FGSM adversarial samples and original clean samples in this specific class. In this specific class, original clean data from a distinguishable cluster, while the adversarial samples scatter over the space and there is no boundary between samples with different ϵ values. This indicates that it is possible to separate adversarial samples and original clean data given the original class. This suggests a possibility to build a separately trained machine learning classifier that differentiates adversarial samples from clean samples.

6 Conclusion

We experimented adversarial example crafting on various datasets and demonstrated the success rate of fast gradient sign method and saliency map method on each dataset. We also experimented FGSM adversarial training: we used various ϵ values to both train and attack adversarially trained models from different datasets.

For cleanly trained models, we found that training models with clean data for more epochs gives model more protection over adversarial attacks. Simpler datasets with clean backgrounds are also better at sustaining both whitebox and blackbox attacks, both for cleanly trained models and models with adversasrial training. This may relate to the fact that the class boundaries are more clear for simplers datasets.

For adversarial training, adversarial training with a specific ϵ value only does not provide much protection for attacks constructed using other ϵ. It may be required training samples with varied ϵ values to obtain a robust defense model. In addition, adversarial training using data with fixed ϵ, although provides some defense, is still susceptible to cross-model attacks.

Dataset inspection shows that the difference between adversarial samples and original data samples, though exist, is much less than the difference between data from different original classes. This suggests that adversarial samples, although are able to confuse specific machine learning models, can still to be correctly detected in some ways.

References

[1] Goodfellow, Ian J et al., *Explaining and Harnessing Adversarial Examples*, arXiv:1412.6572v3, 3/2015

[2] Nicolas Papernot et. al., *The Limitations of Deep Learning in Adversarial Settings*, arXiv 1511.07528v1, 11/2015

[3] Nicolas Papernot et. al., *Practical Black-Box Attacks against Machine Learning*. ASIA CCS '17, 2017.

[4] *The CIFAR-10 dataset*, https://www.cs.toronto.edu/ kriz/cifar.html

[5] *German Traffic Sign Recognition Benchmark*. http://benchmark.ini.rub.de/?section=gtsrb&subsection=dataset

[6] *Fashion MNIST*, https://github.com/zalandoresearch/fashion-mnist

[7] *MNIST with Background*, http://www.iro.umontreal.ca/ lisa/twiki/bin/view.cgi/Public/MnistVariations

[8] *MNIST*, http://yann.lecun.com/exdb/mnist/

[9] *The Street View House Numbers(SVHN) Dataset*, http://ufldl.stanford.edu/housenumbers/

[10] *CleverHans*, https://github.com/tensorflow/cleverhans

[11] *TensorFlow*, http://tensorflow.org/

7 Appendix

Model After Adversarial Training $\epsilon = 0.5$	Accuracy on Legitimate Examples	Accuracy on Adversarial Examples
FMNIST 1	0.991	0.991
FMNIST 2	0.992	0.992
FMNIST 3	0.991	0.990
FMNIST 4	0.991	0.990
FMNIST 5	0.992	0.992
MNIST 1	0.985	0.981
MNIST 2	0.982	0.978
MNIST 3	0.982	0.980
MNIST 4	0.989	0.983
MNIST 5	0.982	0.977
GTSRB10 1	0.999	0.998
GTSRB10 2	0.998	0.995
GTSRB10 3	0.999	0.997
GTSRB10 4	0.999	0.998
GTSRB10 5	0.999	0.998
GTSRB43 1	0.980	0.942
GTSRB43 2	0.976	0.960
GTSRB43 3	0.974	0.939
GTSRB43 4	0.979	0.968
GTSRB43 5	0.978	0.944

Table 4: Accuracy of Model After Adversarial Training $\epsilon = 0.5$ Used in Figure 6a and Figure 6b