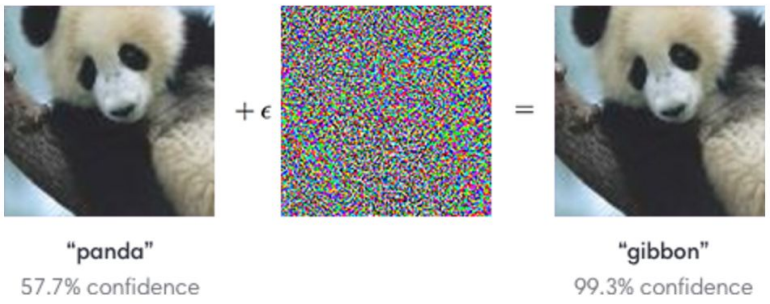
Introduction

In recent years, machine learning has made rapid progress and has achieved remarkable success in a number of areas. Though there has been a good deal of work in the area of security-specific ML models (such as credit card fraud detection), there has been little emphasis on the security aspect of machine learning models in a more general way - ensuring that all types of machine learning systems are robust under adversarial conditions. Adversarial Machine Learning is an active area of research which combines both machine learning and computer security. It is often assumed in AI and ML that the training and test data are drawn from the same distributions. The presence of an intelligent adversary means this assumption will at times be violated.

Examples include check recognition (getting the system to incorrectly determine the check amount), spam filtering (getting the system to misclassify spam), biometric recognition, malware and virus detection, and many others. As machine learning is becoming rapidly integrated into nearly all areas of business and technology, the security of machine learning models is growing in importance. Initial results show that it is very easy to generate adversarial examples which will cause image classifier and other models to perform very poorly. Szegedy et al. (2014) discovered the vulnerability of many machine learning models (including state-of-the-art models with very high accuracy) to adversarial examples[1]. ML models often misclassify examples that are only slightly different from correctly classified ones. The differences can be so slight that the

[1] Szegedy, et al., “Intriguing Properties of Neural Networks,” arXiv:1312.6199v4 [cs.CV], Feb. 2014. p. 1.

differences can be undetectable by the human eye, as in the following example, reprinted from Goodfellow, et al. [2015][2].



Methodology

We trained three neural network models to perform character recognition on the MNIST dataset as well as a decision tree model for classification against Fisher's Iris dataset.

MNIST classifier models: The first model is a simple two-layer network (one convolutional layer with 32 neurons, one fully connected layer with 10 neurons). This model provides an easy-to-train baseline. The second model is a deeper network with three convolutional layers and a fully connected layer. The third model is a modified version of the second which retains the same type of layers but adds neurons to each.

Iris Model: The decision tree model splits nodes using the Gini score. Our decision tree was trained with a max depth of 10 and minimum split-size of 15.

The cleverhans Python library streamlines the process of attack generation and model testing. Despite the wide range of available AI and ML models and applications, it is possible to construct attacks that are effective against many different machine learning

[2] Ian Goodfellow, Jonathan Shlens, and Christian Szegedy, "Explaining and Harnessing Adversarial Examples," arXiv:1412.6572v3 [stat.ML], Mar. 2015. p. 3.

architectures. The cleverhans authors (Goodfellow et al. [2015]) argue that despite speculative explanations focusing on nonlinearity in neural network models as the cause of attack susceptibility, this hypothesis is not necessary. They show that, “Linear behavior in high-dimensional spaces is sufficient to cause adversarial examples. This view enables us to design a fast method of generating adversarial examples that makes adversarial training practical.”[3]

The most common type of adversarial attacks are gradient-based. These attacks look at an image in one class, observe which direction in picture space makes the probability of another class increase, then perturb the input in that direction. We used cleverhans to generate these fast gradient sign method attacks against the image classifier models to measure their susceptibility to attack.

Decision Tree Attack: For the decision tree attack, we used a method inspired by Grosse, et al[4]. This method is a targeted adversarial attack that transforms a test example labeled as X into a slightly augmented example that is labeled as Y, some target class. To achieve a minimal augmentation of the example, we search the decision tree from the leaf node of our victim example and find the shortest path to a leaf node of the target class. Since decision trees use information theoretic scoring to split training data into more homogenous groups, this means that our closest target leaf node is the closest target class node, informationally, to our victim node. We use an implementation of the Uniform Cost Search algorithm to find the shortest path. Then, this path is traversed and at each internal node between the starting victim leaf node and ending target leaf node, we observe the feature and value used in that internal node to split the

[3] Goodfellow, et al., 1.
[4] Grosse, et al., “On The (Statistical) Detection of Adversarial Examples,” arXiv:1702.06280v2 [cs.CR], Oct. 2017.

data. We then augment the victim data such that the feature in question just barely falls on the target's side of the internal node's branch. At the end of this process, we have an augmented example originally of some victim class that is minimally changed to classify as our target class.

Experiments and Dataset

MNIST classifiers

The dataset for the image classifier attacks is the very common MNIST collection of handwritten digit characters. This dataset is often used for testing new machine learning models because many other results with this dataset are available for comparison. State-of-the-art models achieve ~99.8% accuracy on this dataset.

Each model was trained initially using original (clean) MNIST data. After this training, the models were tested on clean test data. Then the models were tested on adversarial examples generated using gradient-based attacks. Next, each model was retrained using a mix of clean and adversarial examples. After retraining, the models were testing against clean examples and adversarial examples separately.

Decision tree / Iris

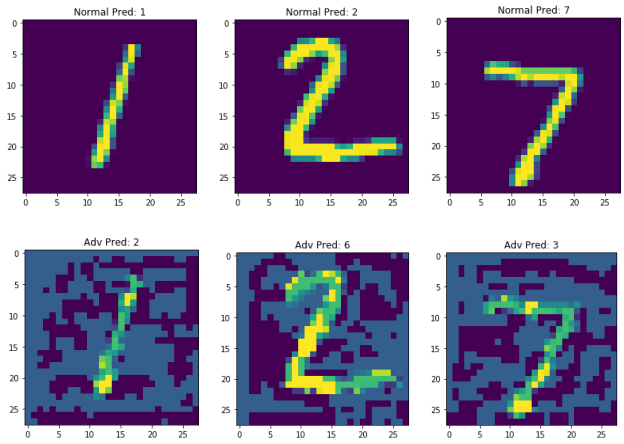
The Iris dataset used for the decision tree model was the famous dataset collected by Fisher with sepal length and width and petal length and width features and possible classes of: versicolor, virginica, and setosa.

Results

MNIST - Our results indicate that the models are not at all robust to adversarial attack. In the presence of adversarial examples the first model breaks almost completely, correctly classifying only ~2% of the input images. The second model with deeper structure does somewhat better, correctly classifying ~12% of attack images. The highest accuracy under adversarial attack is only ~19%, which demonstrates the very high vulnerability to adversarial examples.

There does not seem to be a direct correlation between model accuracy and accuracy under attack. Model 3 is less accurate under clean examples but more accurate under attack.

Model	1	2	3
Accuracy (clean training)	97.3%	99.29%	98.5%
Accuracy (under attack)	1.9%	12.4%	19%
Clean accuracy (after adv. training)	94.7%	97.3%	97.3%
Adv. accuracy (after adv. training)	87.1%	62.2%	61.7%



Adversarial MNIST examples - top line is original images, bottom is modified attack images misclassified by the models

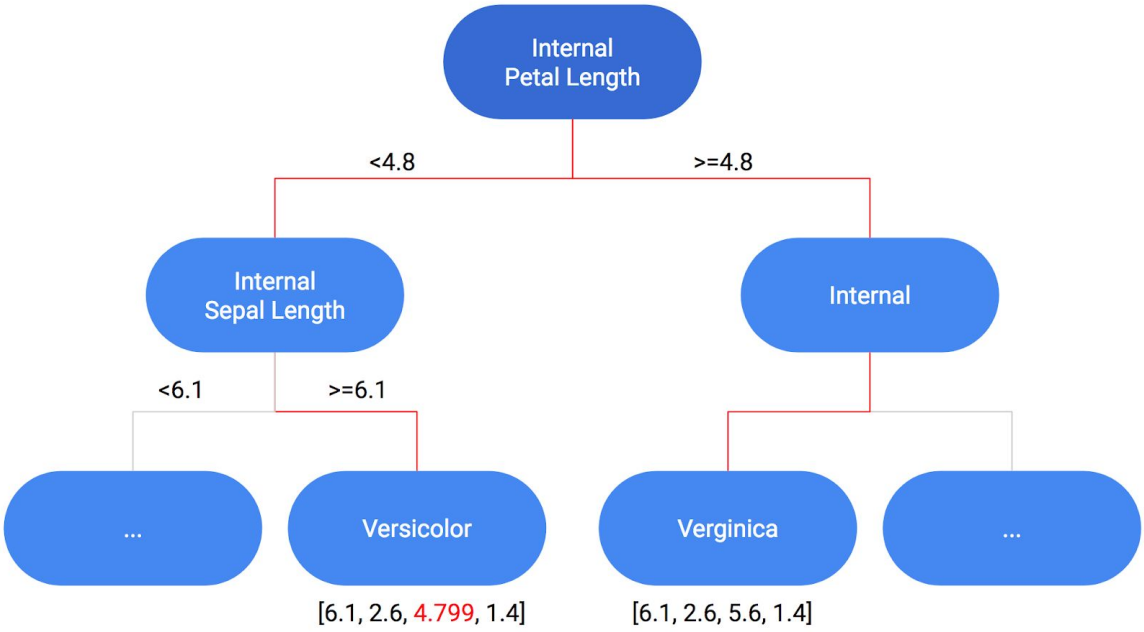
Decision Tree - The decision tree model achieves a 98% accuracy rate on the original test data. The decision tree attack does not use a gradient method, but instead uses a shortest path method using Uniform Cost Search, so it will always achieve a misclassification because it always makes the necessary changes to the victim data in order to classify as the (closest) target class. Therefore, the decision tree can always be attacked to result in a 100% misclassification rate. In general, the attack may need to make large changes, but in our testing, the attack need only make small changes to successfully deceive the decision tree. Here is an example of a misclassification with a victim class of virginica and an adversarial target class of versicolor:

[6.1, 2.6, 5.6, 1.4] → virginica [original data and correct predicted class]

[6.1, 2.6, 4.799, 1.4] → versicolor [adversarial data and target class prediction]

Note that the features are, respectively: sepal length, sepal width, petal length, and petal width. The attack did not touch sepal length, sepal width, or petal width because it did not need to in order to reach the target class in its shortest path traversal. Note that petal length is minimally changed. For petal length, an internal node was splitting on petal

length with a comparison value of 4.8 and our target class was on the left branch (the less-than branch), so we made our new petal length the comparison value (4.8) less a small value to fall on the left side of the branch with a new value of 4.799. The resulting adversarial example indeed classifies as our target class and underwent only minimal data changes. The relevant portion of the tree and the traversal are shown below. Note in the figure below, there is a traversal on a sepal length node, but we did not need to change our example in order to fall on the target side (the right side in this case) of the node.



The figure above shows the tree and adversarial traversal for the example given above. Starting from the Verginica node, we traverse along the red lines. First, we traverse to a parent internal node, then once again to the lowest common ancestor between the Verginica node and the target Versicolor node. This ancestor node splits on petal length at 4.8. The target is on the left side of 4.8, so we need to update the data to fall on the

left side. Our adversarial example uses a value of 4.8 - 0.001 = 4.799, which is shown in red in the figure above under Versicolor. Next, we traverse down to an internal node and further down on its right side. This right downward traversal happens against sepal length with a comparison value of 6.1. Since our original example already has a value of 6.1, we need not update our adversarial data. Finally, we end up with a minimally changed example of [6.1, 2.6, 4.799, 1.4].

Vulnerability of Machine Learning Models

Our results show that even high-accuracy machine learning models are extremely easy to fool. The attacks can be very effective even when they do not build in model-specific information. It should be very concerning that it is so easy to generate attacks that work across a range of different models. The significant vulnerability of these models to adversarial attack is somewhat surprising. If a model is able to classify images with >99% accuracy, why does the model do so poorly when presented with examples in which the input is modified only slightly? There are a number of possible explanations. Szegedy et al. show that the smoothness assumption that underlies many methods does not hold. They explain that, “Independently of their generalisation properties across networks and training sets, the adversarial examples show that there exist small additive perturbations of the input (in Euclidean sense) that produce large perturbations at the output of the last layer.”[5] Goodfellow et al. give two main reasons for the difficulty in defending against adversarial attack. First, they explain that,

> Adversarial examples are hard to defend against because it is hard to construct a theoretical model of the adversarial example crafting process. Adversarial examples are solutions to an optimization problem that is nonlinear and non-convex for many ML models, including neural networks. Because we don’t have good theoretical tools for describing the solutions to these complicated optimization

[5] Szegedy, et al, 8.

> problems, it is very hard to make any kind of theoretical argument that a defense will rule out a set of adversarial examples.[6]

The second issue is the size of the possible input space. Adversarial attack is difficult to defend against because it requires machine learning models to produce the correct output for every possible input. Successful machine learning models typically only work on a very small portion of all possible inputs.

Conclusion

These results are certainly concerning. But the field is new, and there is not yet much context in which to place such results. But they seem to imply significant security issues in the big picture of machine learning development. In the words of Goodfellow, et al. [2015], “Gradient-based optimization is the workhorse of modern AI”[7] This partially accounts for the effectiveness of gradient-based attack methods which do not incorporate model-specific information and yet are effective against many different types of models. Adversarial machine learning has emerged as a field only very recently, and there are still many open questions. Initial results suggest that defending against attack is much harder than creating effective attacks, and so it seems that there may be an asymmetry in the security balance regarding machine learning. The theoretical aspects of adversarial attack are very poorly understood, so there is much progress to be made in this area. It is probably safe to assume that once more attention is devoted to security issues, that significant progress will be made in this area as well. But at the current

[6] http://www.cleverhans.io/security/privacy/ml/2017/02/15/why-attacking-machine-learning-is-easier-than-defending-it.html
[7] Goodfellow, et al., 9.

moment, adversarial attack represents a significant blind spot for almost all current

machine learning models.