Black-Box Adversarial Machine Learning

Ryan Kingery, Nitin Nair
Bradley Department of Electrical and Computer Engineering
Virginia Tech
Blacksburg, Virginia
{rkingery,nitinnair}@vt.edu

***Abstract*—It's become well-known in recent years that machine learning models are not robust to adversarial examples. By altering input data in what are often human-imperceptible ways, one can completely fool classifiers into making wrong decisions with confidence. Moreover, it's been shown that even black-box models are vulnerable from adversarial examples trained on completely different models. In this report we examine the effects of black-box adversarial machine learning in two domains: vision and text. We demonstrate that black-box classification model accuracies in both domains can be substantially affected by the generation of adversarial examples.**

I. Introduction

While all software has its own set of security vulnerabilities, software incorporating machine learning has its own set of vulnerabilities to deal with. In particular, machine learning models are prone to manipulation by the creation of adversarial examples. Adversarial examples are inputs designed specifically to fool the model into making incorrect decisions with high confidence.

An attacker can use adversarial examples to do things like trick a spam classifier into classifying spam as non-spam, hence allowing desired spam to pass through to the user; or, he could trick an object detection system in an autonomous vehicle into classifying a stop sign as a yield sign, which could easily cause an accident. There are even more dangers with an online-learning model, as an attacker can also use adversarial examples to poison such a model by using bad training data to alter the behavior of the model and make it completely unusable.

Adversarial examples can be crafted by altering clean input data in ways that are often imperceptible to humans. In one famous example, suppose one desires to fool an image classifier designed to classify ImageNet images. As shown in Figure 1, one can take an image of a panda and alter it by adding imperceptible amounts of noise to fool the model into classifying it as a gibbon with high confidence.

It has also been shown that adversarial examples are transferable. That is, adversarial examples created to fool one machine learning model can often be used to fool other models as well. This allows adversarial machine learning to be performed in a black-box setting. In a black-box setting, one supposes that an attacker wishes to subvert the intent of a machine learning model about which he has little to no knowledge. He doesn't know which model was used or on what data it was trained. He only has a rough idea of what the input and output data look like.

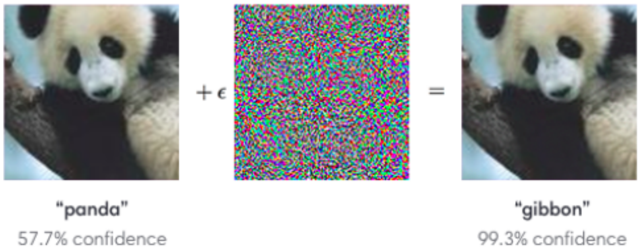


Fig. 1. Example of an adversarial example designed to fool an image classifier, from [1].

For example, in the above ImageNet example, the attacker may have an idea that the black-box model classifies ImageNet-like images, e.g. through querying the model multiple times to understand its signature, but little idea what model was used or exactly which images were used in training. To get around this, he can instead train his own model on a set of images he thinks is close to what the black-box was trained on and use that model to create adversarial images to attack the black-box.

II. Background

A. Supervised Learning

Machine learning is the construction of algorithms that learn from data and can make predictions about data without the need of human input. These algorithms are generally statistical, in the sense that they use a sample of data to uncover information about their underlying distribution. Machine learning models that capture the underlying distribution well are said to generalize. The most well-developed and commonly utilized sub-field of machine learning is supervised learning, which is where most applications of adversarial machine learning tend to apply.

In supervised learning, the goal is to use a set of inputs to predict a set of outputs. More formally, suppose a dataset

$$D = \{(x_1, y_1), \cdots, (x_N, y_N)\}$$

contains N pairs of feature vectors $x \in X$ and targets $y \in Y$ sampled from a joint probability distribution $p(x, y)$. Suppose this distribution can be expressed by a function $f(x)$ plus noise ε, so $y = f(x) + \varepsilon$. The goal of supervised learning, then, is to use an algorithm $\mathcal{A}$ to learn a function g from some model

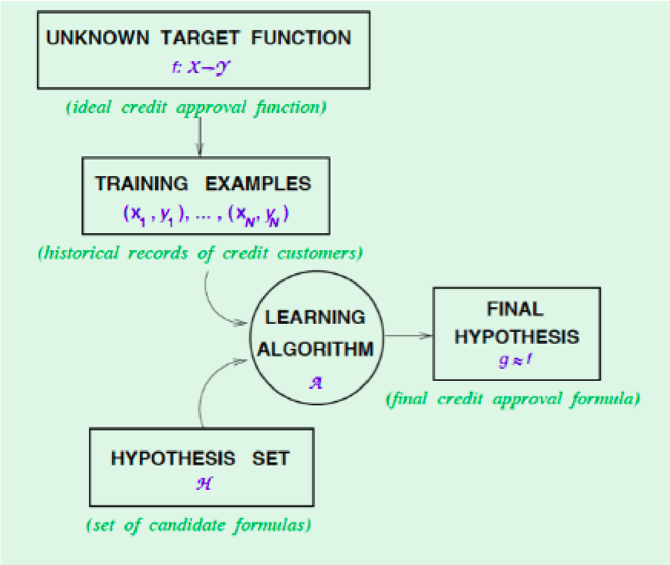

UNKNOWN TARGET FUNCTION
f: X→Y
(ideal credit approval function)
TRAINING EXAMPLES
(x1 , y1), ... , (xN, yN)
(historical records of credit customers)
LEARNING ALGORITHM
A
FINAL HYPOTHESIS
g ≈ f
(final credit approval formula)
HYPOTHESIS SET
H
(set of candidate formulas)

Fig. 2. The supervised learning process, from [2].

class $\mathcal{H}$ such that $g(x) \approx f(x)$ for all $x \in X$. An illustration of the supervised learning process can be shown in Figure 2.

This process can be easily understood with the simple example of logistic regression. Logistic regression is a simple type of classifier, i.e. a supervised learning algorithm in which the target space Y discrete, in which the outputs are usually called labels. For logistic regression, each $x \in \mathbb{R}^p$ and $y \in \{0, 1\}$, and f is assumed to be a binary-valued function on X. The model class $\mathcal{H}$ is composed of parametric functions of the form

$$g(x|\theta) = \frac{1}{1 + e^{-(w^T x + b)}},$$

where $\theta = \{w, b\}$ are parameters to be estimated from the learning algorithm. The learning algorithm $\mathcal{A}$ is typically a simple optimization algorithm that seeks to minimize some loss function $L(\theta|x, y)$ defined on the data with respect to the model parameters. A simple optimization algorithm for doing so is gradient descent, which updates the parameters according to the rule

$$\theta \leftarrow \theta - \alpha \nabla_\theta L(\theta|x, y)$$

until convergence, where α is a predefined learning rate that determines the convergence rate.

An extension of logistic regression is the neural network, which is the workhorse class of models for modern deep learning methods. An L-layered neural network $N(x|\theta)$ is a composition of non-linear, parametric activation functions $a_l = g_l(a_{l-1}|\theta_l)$,

$$N(x|\theta) = g_L(a_{L-1}|\theta_L) = \cdots = g_L(\cdots g_1(x|\theta_1)|\theta_L).$$

Note $a_0 \equiv x$. The exact structure of each activation function depends on the type of layers desired. For example, a convolutional neural network (CNN) contains convolutional layers of the form $a_l = \max(0, W_l * a_{l-1} + b_l)$, usually combined with fully-connected layers of the form $a_l = \sigma(W_l a_{l-1} + b_l)$ for some increasing function σ.

B. Adversarial Machine Learning

In adversarial machine learning, one attempts to subvert the supervised learning process by crafting adversarial examples, i.e. feature vectors $x_{adv} \in X$ such that $g(x_{adv}) \neq f(x_{adv})$ even approximately. The simplest and most common way to craft such examples is via the fast sign gradient method (FSGM), which uses gradient descent in a slightly different way. One takes a clean example $x \in X$ and perturbs it by an amount

$$\eta \equiv \varepsilon \text{sgn}(\nabla_x L(\theta|x, y))$$

with predefined perturbation parameter ε, to get $x_{adv} = x + \eta$. Further improvements can be made by then attempting to maximize the loss $L(\theta|x, y)$ via constant, ε-sized gradient ascent steps on the inputs,

$$x_{adv} \leftarrow x_{adv} + \eta.$$

The perturbation η can be thought of as an additive noise term, and is usually made to be small enough such that the adversarial examples resembles the original sample as much as possible.

The approach described above implicitly assumes an attacker has knowledge of the underlying model due to the dependence of the FSGM method on the loss, which itself depends on the model output. It's been shown empirically that adversarial examples are often transferable [3]. That is, adversarial examples trained on one model can be transferred to a different model and still often act as an adversarial examples on that model.

III. Related Work

While adversarial machine learning has been studied in some form for the past couple of decades, its modern incarnate is based on the work by Goodfellow, Shlens, and Szegedy in [1]. It's in this work that attention was called to the fact that images can be altered imperceptibly to produce adversarial misclassifications with high confidence. This work also introduced the fast sign gradient method, as well as the vulnerability of even the simplest machine learning models to adversarial attacks, not just deep learning methods.

The vulnerability of machine learning models to black-box attacks was called attention to in [4] and [3]. The work in [4] motivated this vulnerability by reasoning that the attack surfaces between different models often look very similar. The work in [3] demonstrated the viability of black-box attack methods by attacking an image classifier independently trained and deployed on a remote server.

Frameworks for doing adversarial machine learning are fairly new. The first, perhaps, was the CleverHans library in [5]. CleverHans is a Python library compatible with TensorFlow, used to benchmark the vulnerability of machine learning systems to adversarial examples. Another library is FoolBox [6], a multi-framework compatible Python toolbox to create adversarial examples that fool neural networks. Another library is the Adversarial Robustness Toolbox (ART) [7], a multi-framework compatible Python library that allows for the rapid crafting and analysis of attacks and defense methods for machine learning models.

IV. Experiments

In our experiments, we employ a simplified black-box methodology to generate adversarial examples and attack classifiers in two common learning domains, vision and text. In each scenario, we employ the following methodology:

1) Obtain and partition the original dataset into two sets, with most going to train and evaluate the black-box model and the rest going to train and evaluate a substitute model.
2) Choose a black-box model and a substitute model. These models should be distinct from each other, but should approximate the state-of-the-art where feasible.
3) Use the black-box subset of data to train the black-box model to maximal test accuracy.
4) Use the substitute subset of data to train the substitute model to maximal test accuracy.
5) Use the FSGM on a subset of the substitute data to generate adversarial examples on the substitute model.
6) Evaluate the accuracy of the generated adversarial examples on the black-box model, and compare with the accuracy of the equivalent non-adversarial examples.

A. Attacking a Vision Model

The vision model we choose to attack is a traffic sign classifier. The classifier takes as input an image containing a traffic sign and outputs a label corresponding to what type of traffic sign it is. The dataset used is the German Traffic Sign Recognition Benchmark (GTSRB) dataset from [8], which contains 51837 images of German 43 different classes of traffic signs. Some examples of these images and their labels are shown in Figure 3.

For conducting the black-box attack, we use 39208 of the images for the black-box dataset and the remaining 12629 for the substitute dataset. Each image is first lightly processed by performing histogram equalization and center cropping, and then resized to a standard size of 48 × 48.

For the black-box model, we chose to fine-tune the pretrained model from VGG-16 [9]. This technique is an instance of transfer learning, and has been shown to work very well with image classification. VGG-16 is a CNN consisting of 16 layers of weighted layers; its architecture is shown in Figure 4. Using fine-tuning, we train the black-box model in Keras to a 96% test-set accuracy. For the substitute model, we trained a custom CNN with 8 weighted layers using Keras to a test-set accuracy of 97%.

Next, we use the ART implementation of FSGM ($\varepsilon = 0.1$) to create 100 adversarial examples on the substitute CNN model. The prediction accuracy of both models for the original examples are shown in Table I, and for the adversarial examples in Table II. We can see in particular that, while the adversarial examples are much more effective at misclassifying on the substitute model (they were created using that model, after all), they are still effective at misclassifying on the black-box model as well. An example of the black-box prediction for one of these examples is shown in Figure 5. Observe that, while the black-box model could predict this image is a 100 km/h speed limit sign with perfect confidence, the adversarial image was predicted to be a 120 km/h speed limit sign with 98% confidence, despite the fact that the sign is still obviously a 100 km/h sign to a human.

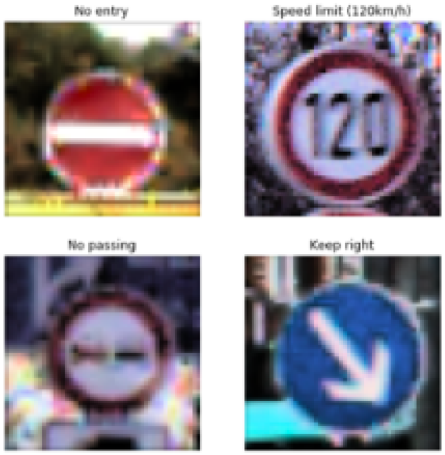


Fig. 3. Examples of traffic sign images from the GTSRB dataset along with their labels.

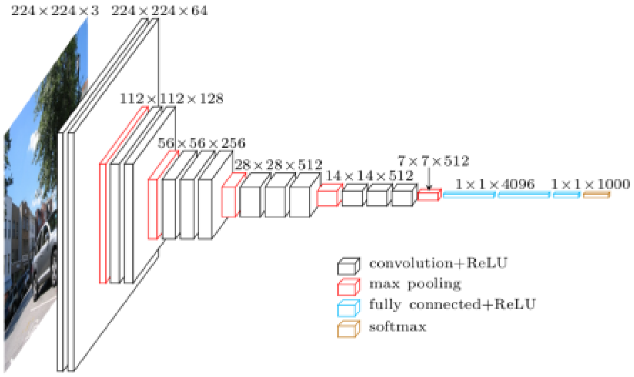


Fig. 4. The VGG-16 architecture, from [9].

B. Attacking a Text Model

Due to the discontinuous nature of the distribution of tokens in natural language processing, generating adversarial examples for text is more difficult than for images. When

TABLE I
Traffic sign classifier results with original examples.

	Accuracy (%)
Substitute	100
Black-Box	85

TABLE II
TRAFFIC SIGN CLASSIFIER RESULTS WITH ADVERSARIAL EXAMPLES.

	Accuracy (%)
Substitute	13
Black-Box	57

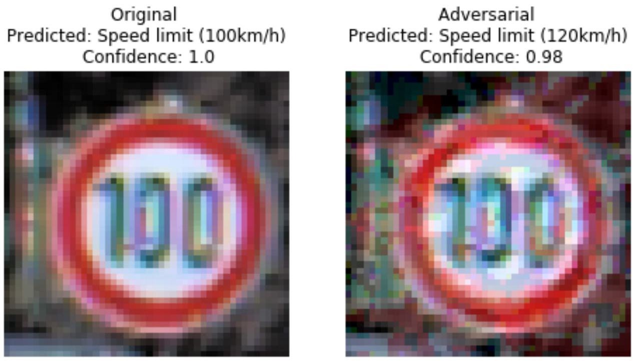

Original
Predicted: Speed limit (100km/h)
Confidence: 1.0
Adversarial
Predicted: Speed limit (120km/h)
Confidence: 0.98

Fig. 5. Black-box predictions on a sampled image and its adversarial equivalent.

working with text, one generally first has to convert the plain text into a numerical representation first. We choose to do this using word embeddings, specifically GloVe [10]. For GloVe embeddings, training is performed on aggregated global word-word co-occurrence statistics from a corpus. GloVe has been shown to capture interesting semantic relationships in the underlying vocabulary, which makes it useful for doing machine learning with text.

Two experiments were conducted as given below. The first was to build a spam classifier from the sentence vectors and generate adversarial examples from it. The second was to build a word sentiment classifier from the word vectors and generate adversarial examples from it.

1) Spam Classifier: For this task, we chose a support vector machine (SVM) with RBF kernel as the black-box model, and a custom-built CNN model for the substitute model. The dataset used for this task was the SMS Spam Collection from [11].

Each sentence in the dataset was first preprocessed to remove certain elements like text and URLs after with the average of all the word embeddings was taken as the sentence vector. The word embeddings were then created using 300-dimensional word vectors trained on Common Crawl with GloVe.

The adversarial examples are created using the FSGM implementation from ART ($\varepsilon = 0.5$). The results of the experiment is shown in Table III and Table IV. Due to the way the sentence vector was created, the plain text of the adversarial examples could not be obtained. Nevertheless, we can see that the adversarial samples are able to substantially reduce the accuracy on the black-box model from 95% to 56%.

TABLE III
SPAM CLASSIFIER RESULTS WITH ORIGINAL EXAMPLES.

	Accuracy (%)
Substitute	95.96
Black-Box	94.79

TABLE IV
SPAM CLASSIFIER RESULTS WITH ADVERSARIAL EXAMPLES.

	Accuracy (%)
Substitute	15.33
Black-Box	55.56

2) Word Sentiment Classifier: A word sentiment classifier is used to predict whether a given word has positive or negative sentiment. The dataset used for this task was taken from [12]. The black-box model created was again an SVM with an RBF kernel, and the substitute model was again a custom-built CNN model. The adversarial examples are created using FSGM with $\varepsilon = 0.5$. The results are shown in Table V and Table VI. We can again see a substantial decrease in the accuracy of the black-box model on the adversarial examples.

We can see some examples of the adversarial text in Figure 6. In one example, the word "unemployed" can be adversarially changed to "re-employed", causing a sentiment classifier to relabel it from positive to negative sentiment. In the second example, "intelligence" is adversarially changed to "counterintelligence", causing it to again change sentiment.

V. CONCLUSION AND FUTURE WORK

In this report, we have shown that a host of black-box machine learning models in multiple domains are vulnerable to adversarial attacks. By training adversarial examples on a substitute model that may bear little relation to the black-box model, an attacker can successfully subvert the intent of the black-box model and use model misclassification to achieve some desired behavior. Moreover, we have demonstrated that adversarial machine learning is not just a deep learning problem, as is commonly believed, but rather a much more general problem. We showed that an SVM is just as susceptible to attack as a CNN. In fact, [1] argues that pretty much all machine learning models are just as vulnerable.

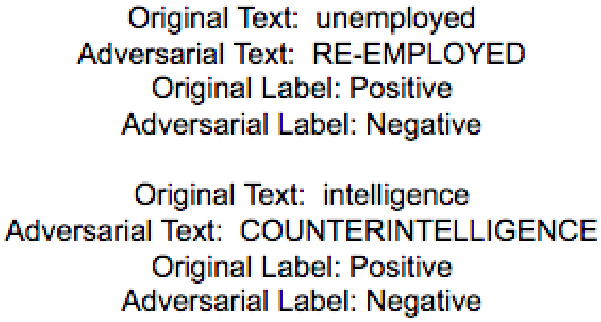
A natural question to ask at this point is, if black-box models are so easily prone to adversarial attacks, what can be done to protect them from such attacks? One easy thing to try would be to use adversarial examples during training to innoculate the

TABLE V
SENTIMENT CLASSIFIER RESULTS WITH ORIGINAL EXAMPLES.

	Accuracy (%)
Substitute	89.69
Black-Box	90.5

TABLE VI
SENTIMENT CLASSIFIER RESULTS WITH ADVERSARIAL EXAMPLES.

	Accuracy (%)
Substitute	10.97
Black-Box	14.08



Original Text: unemployed
Adversarial Text: RE-EMPLOYED
Original Label: Positive
Adversarial Label: Negative

Original Text: intelligence
Adversarial Text: COUNTERINTELLIGENCE
Original Label: Positive
Adversarial Label: Negative

Fig. 6. Black-box predictions on sampled words and their adversarial equivalents.

model from being attacked with such examples. While such an approach may help with certain types of adversarial example crafting methods, it won't work for all possible adversarial examples. There are other defense strategies as well, but this is still very much an area of active research. Before we can better understand defense strategies, we must better understand why adversarial examples occur so easily in the first place.

Addressing the efficacy of various defense strategies is a natural thing to add to future work. Another thing to add to future work is the creation of more realistic attack scenarios. In this report, for simplicity we trained both the black-box and substitute models, and sampled the training data for both from the same original dataset. In more realistic scenarios, the attacker would likely have no knowledge at all of how the black-box model was trained or what dataset was used to train it.

Another fruitful task of future work, in the spirit of our original intent, is to extend this analysis to further domains. In particular, we did not examine the effects of adversarial machine learning techniques on structured data. Structured data techniques include common scenarios like network data, financial data, and recommendation systems. We believe, however, that structured data scenarios are just as vulnerable as vision and text models, and hope that in the future their vulnerabilities will be further addressed as well.

References

[1] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[2] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H.-T. Lin, *Learning From Data*. AMLBook, 2012.

[3] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17, (New York, NY, USA), pp. 506–519, ACM, 2017.

[4] C. Szegedy, G. Inc, W. Zaremba, I. Sutskever, G. Inc, J. Bruna, D. Erhan, G. Inc, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *ICLR*, 2014.

[5] N. Papernot, F. Faghri, N. Carlini, I. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy, A. Matyasko, V. Behzadan, K. Hambardzumyan, Z. Zhang, Y.-L. Juang, Z. Li, R. Sheatsley, A. Garg, J. Uesato, W. Gierke, Y. Dong, D. Berthelot, P. Hendricks, J. Rauber, and R. Long, "Technical report on the cleverhans v2.1.0 adversarial examples library," *arXiv preprint arXiv:1610.00768*, 2018.

[6] J. Rauber, W. Brendel, and M. Bethge, "Foolbox: A python toolbox to benchmark the robustness of machine learning models," *arXiv preprint arXiv:1707.04131*, 2017.

[7] M.-I. Nicolae, M. Sinn, M. N. Tran, A. Rawat, M. Wistuba, V. Zantedeschi, N. Baracaldo, B. Chen, H. Ludwig, I. Molloy, and B. Edwards, "Adversarial robustness toolbox v0.3.0," *CoRR*, vol. 1807.01069, 2018.

[8] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark," in *International Joint Conference on Neural Networks*, no. 1288, 2013.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[10] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

[11] T. A. Almeida, J. M. G. Hidalgo, and A. Yamakami, "Contributions to the study of sms spam filtering: new collection and results," in *Proceedings of the 11th ACM symposium on Document engineering*, pp. 259–262, ACM, 2011.

[12] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, (New York, NY, USA), pp. 168–177, ACM, 2004.