An Introduction to Adversarial Machine Learning

Atul Kumar[1(✉)], Sameep Mehta[2], and Deepak Vijaykeerthy[1]

[1] IBM Research, G2 Block, 8th Fl., Manyata Tech Park, Ngawara, Bangalore 560045, India
{kumar.atul,dvijayke}@in.ibm.com

[2] IBM Research, ISID Campus, Institutional Area, Vasant Kunj, New Delhi 110070, India
sameepmehta@in.ibm.com

Abstract. Machine learning based system are increasingly being used for sensitive tasks such as security surveillance, guiding autonomous vehicle, taking investment decisions, detecting and blocking network intrusion and malware etc. However, recent research has shown that machine learning models are venerable to attacks by adversaries at all phases of machine learning (e.g., training data collection, training, operation). All model classes of machine learning systems can be misled by providing carefully crafted inputs making them wrongly classify inputs. Maliciously created input samples can affect the learning process of a ML system by either slowing the learning process, or affecting the performance of the learned model or causing the system make error only in attacker's planned scenario. Because of these developments, understanding security of machine learning algorithms and systems is emerging as an important research area among computer security and machine learning researchers and practitioners. We present a survey of this emerging area named Adversarial machine learning.

Keywords: Adversarial learning · Computer security · Intrusion detection

1 Introduction

Over last few years, machine leaning has become a prominent technological tool in several application areas such as computer vision, speech recognition, natural language understanding, recommender systems, information retrieval, computer gaming, medical diagnosis, market analysis etc. In many areas, it is no longer a promising but immature technology as machine learning based systems have reached close to human level performance. Most of machine learning techniques build models using example data (training data). These models along with algorithms can be used to make predictions on data not seen before.

Learning and building models using training data provides hackers opportunities to attack machine learning algorithms by playing with the features and decision boundaries of the model. An adversary can craft malicious inputs to attack the performance or efficiency of a machine learning algorithm. They can dupe an already trained system by creating input data that exploits the system into making glaring errors.

For example, researchers have demonstrated [1], ways to fool an image classification system by making tiny changes to the input images. Figure 1 shows several sets of examples with three images in each set. In each set, on the left is an image that the system correctly classifies. Image in center is a noise which when added to the left image creates an image shown on the right which still looks like the image on left to a human observer. But the system now classifies the image on right as an ostrich for every example. These techniques can be used by hackers to evade the system in making it accept malicious content as a genuine one. With machine learning becoming an important tool in strategically important applications such as security surveillance and background check for visa decisions etc., it is important to understand these attacks and make machine learning algorithms more robust against these attacks.

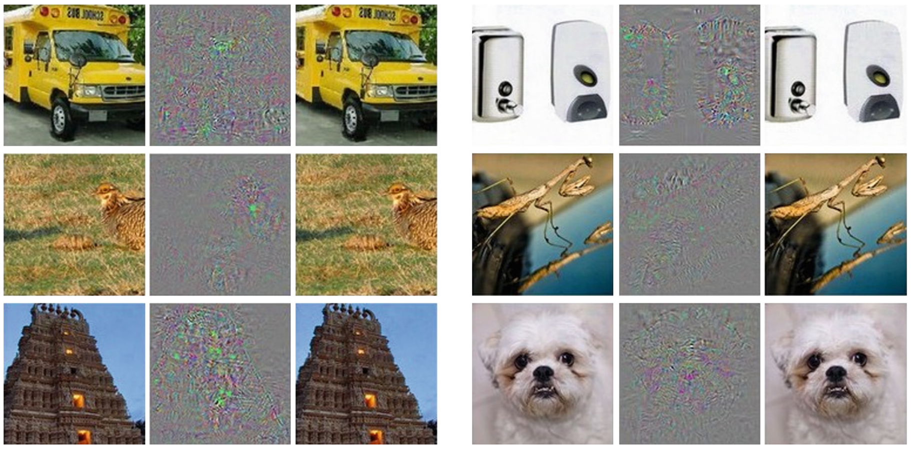

Fig. 1. Creating adversarial examples using noise (Image credit: Szegedy et al. [1])

If a hacker does not already know the algorithm, he first tries to learn the algorithm and its underlying model (e.g., logistic regression, neural network, decision trees etc.). Sometime, the hacker may only be interested in learning the model so that he can build his own 'copy' of the system using the learned model. This may be useful if the application is offered as a service via APIs and users are charged per use of these APIs. A hacker can create a sequence of inputs and then by observing outputs of the system corresponding these inputs, he can build a local model that may be very close to the model used by the original system. Depending on the pricing and the license terms of the API usage, a hacker may be able to 'steal' the model using very small amount of money. Tramer et al. demonstrated at USENIX Security Symposium 2016 [2] that models can be extracted from popular online machine learning services such as BigML and Amazon Machine Learning with a relatively small number of API calls.

Another category of attacks on machine learning systems is to provide adversarial input during the training phase and compromise the learning by affecting its efficiency or introducing some bias. Many systems allow users to provide training data samples for online training of the system. Collecting training data from people spared across geographies is immensely valuable in many applications to have good data distribution. But opening the system to public for providing input data also opens a system to malicious input created by hackers to 'poison' the system. Microsoft's twitter chatbot Tay started tweeting racist and sexist tweets in less than 24 h after it was opened to public for learning [3].

Attacks on Machine Learning based systems can be categorized in three broad categories. First set of attacks called exploratory attacks, try to 'steal' the algorithms and their models or some insight into the training data by providing carefully crafted inputs and then observing the output to build local copies of the models. Second set of attacks called evasion attacks, consists of techniques focusing on evading a system by making it classify an input incorrectly. And the third type of attacks called poisoning attacks, try to change the model of the system by providing malicious training examples aiming to alter the model of the machine learning algorithm.

2 Exploratory Attacks

Exploratory attacks do not attempt to influence training; instead they try to discover information from the learner that includes discovering which machine learning algorithm is being used by the system, state of the underlying model and training data.

2.1 Model Extraction Using Online APIs

Machine learning as a service for applications such as predictive analytics are deployed with publicly accessible query interfaces (APIs). These models are deemed confidential due to their sensitive training data, commercial value, or other reasons such as use in security applications. Access is provided on a pay-per-query basis. In such situations, an adversary has black-box access but no prior knowledge of the machine learning model's parameters or training data.

Tramer et al. [2] presented simple attacks to extract target machine learning models for popular model classes such as logistic regression, neural networks, and decision trees. Model extraction attacks were demonstrated on popular online ML-as-a-service providers such as BigML and Amazon Machine Learning. Their attacks were complete black box and the adversary does not even need to know the model type or any distribution information about training data. They could build local models that are functionally very close to the target. In some experiments, their attacks extracted the exact parameters of the target (e.g., the coefficients of a linear classifier or the paths of a decision tree). In situations where the model type, parameters or features of the target were not known, they used an additional preliminary attack step to reverse-engineer these model characteristics. Machine learning prediction APIs of major online services such as Google, Amazon, Microsoft, and BigML all return precision confidence values along with class labels. Moreover, they work with partial queries lacking one or more features. These features can be exploited for model extraction attacks.

3 Evasion Attacks

Evasion attacks are the most prevalent type of attack on a machine learning system. Malicious inputs are carefully crafted to evade detection which essentially means that input is modified to make the machine learning algorithm classify it as a safe one instead of malicious.

3.1 Adversarial Examples

Szegedy et al. [1] found that deep neural networks (DNN) learn input-output mappings that are fairly discontinuous. One can cause a DNN to wrongly classify an image by applying a specifically crafted modification (found by maximizing the network's prediction error) that is difficult to distinguish by a human viewer. The same change to the image can cause a different network, trained on a different subset of the dataset, to incorrectly classify the same image. This property of deep neural network can be exploited to create any number of adversarial inputs from the normal inputs.

Practical Black-Box Attacks method proposed by Papernot et al. [4] misclassified 84.24% of the crafted adversarial examples on MetaMind (an online deep learning API) DNN. They also used logistic regression substitutes to craft adversarial examples for Amazon and Google ML APIs and found misclassification rate of 96.19% and 88.94% respectively.

Papernot et al. [5] show that adversarial attacks are also effective when targeting neural network policies in reinforcement learning. Adversaries capable of introducing small perturbations to the raw input can significantly degrade test-time performance. The strategy is to train a local substitute DNN using a synthesized data set. Input data is synthesized but the label assigned is what the target DNN assigns to it and observed by the adversary. Adversarial examples are generated by using the substitute parameters known to adversary. These are misclassified by both target DNN and the substitute DNN created locally because they both have the same decision boundaries. To create a small perturbation so that the changed image looks similar to the original one, an algorithm named fast gradient sign method [6]. The cost gradient is computed for pixels and the target pixels (areas) for perturbation is identified. Another algorithm by Papernot et al. [7] can cause a misclassification for samples from any legitimate source class to any chosen target class. That is, any image can be changed slightly such that it is classified to a desired class (say ostrich) by the DNN. Therefore, a school bus image can be changed in such a way that to humans, it still looks like a bus but the DNN recognizes it as an ostrich (for that matter any class chosen by the adversary). Input components are added to a perturbation in order of decreasing adversarial saliency value until the resulting adversarial sample is misclassified by the model.

3.2 Generative Adversarial Networks (GANs)

Goodfellow et al. [8] introduced Generative adversarial networks. They are implemented by simultaneously training two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. This can be viewed as a competition between a team of counterfeiters and a team of police. If generative model is assumed to be producing fake currency such that it can pass without detection, then the discriminative model is trying to detect the counterfeit currency. Competition leads both teams to improve their methods until the counterfeits cannot be distinguished from the genuine currency.

3.3 Evasion Attacks on Text Classification Systems

Perturbation techniques for image or audio based system cannot directly work on text based systems. That is because an important requirement of the perturbation used is to change the image or audio such that it still looks good to a human observer/listener. Whereas in a text, changing words by adding/deleting characters or changing sentences by adding/deleting words may make the sentence/word meaningless or change its meaning significantly and therefore cannot remain unnoticed by a human reader. Therefore, a perturbation technique must change the text such that it still looks good/suspicious to a human observer but machine learning system fails to classify it correctly after perturbation. For example, a spam email carrying an advertisement should still carry the advertisement message but fool the spam filtering system in classifying it as a regular email.

Creating adversarial inputs for text classification systems seems to be a harder problem than doing the same for the image or audio classification. Some recent work has shown that it is possible to systematically create such adversarial inputs. Liang et al. [11] discuss the problem of creating perturbation. They propose three techniques named insertion, modification, and removal, to generate adversarial samples for given text. They compute cost gradients (originally proposed in [6] for images and proven to be effective in [7, 12]) to decide what and where should be inserted, what and how to modify and what should be removed from a text sample. However, using the fast gradient sign method (FGSM) of [6] directly makes the text unreadable. Using cost gradient, they identify the text items that possess significant contribution to the classification. Then instead of changing the characters arbitrarily, they use one or more of insertion, modification and removal to craft an adversarial sample for a given text.

4 Poisoning Attacks

In poisoning attacks, attackers try to influence training data to influence the learning outcome. The purpose of poisoning attacks may vary from affecting the performance of learning algorithm to deliberately introducing specific biases in the model. In many applications, training is not a one-time job and model is often retrained to accommodate for the change in data distribution. In some situation, data collection is crowd-sourced and many users provide data sample that are used to continuously train the model. Some domains such as network intrusion detection, spam filtering, malware detection etc. are highly suspect of poisoning attacks but any machine learning system can be a victim of poisoning attacks.

4.1 Defensive Distillation

Papernot et al. [9] introduced a defensive mechanism called defensive distillation that reduces the effectiveness of adversarial samples on deep neural networks (DNNs). Distillation is a training procedure that was designed to train a DNN using knowledge transferred from a different DNN [10]. The motivation behind the knowledge transfer is to reduce the computational complexity of DNN architectures by transferring knowledge from larger architectures to smaller ones. This facilitates the deployment of deep learning in resource constrained devices that cannot rely on powerful GPUs to perform computations. A new variant of distillation is proposed for defense training. Instead of transferring knowledge between different architectures, knowledge extracted from a DNN is used to improve its own resilience to adversarial samples. An analytical investigation is presented for the generalizability and robustness properties granted by defensive distillation when training DNNs. Two DNNs were placed in adversarial settings to empirically study the effectiveness of defensive distillation. They show that defensive distillation can reduce effectiveness of sample creation from 95% to less than 0.5% on the DNNs used in their study. This can be explained by the fact that distillation reduces by a factor of 1030 the gradients used in adversarial sample creation. Distillation also increases by 800% the average minimum number of features required to be modified for creating adversarial samples on one of the DNNs used in their experiments.

References

1. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., Fergus, R.: Intriguing Properties of Neural Networks. https://arxiv.org/pdf/1312.6199v4.pdf
2. Tramer, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T.: Stealing machine learning models via prediction APIs. In: USENIX Security Symposium (2016)
3. Reuters: Microsoft's AI Twitter bot goes dark after racist, sexist tweets, 24 March 2016. http://www.reuters.com/article/us-microsoft-twitter-bot-idUSKCN0WQ2LA
4. Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B., Swami, A.: Practical black-box attacks against machine learning. In: ACM Asia Conference on Computer and Communications Security (ASIACCS), April 2017
5. Papernot, N., McDaniel, P., Goodfellow, I.: Transferability in Machine Learning: From Phenomena to Black-Box Attacks using Adversarial Samples. https://arxiv.org/pdf/1605.07277.pdf
6. Goodfellow, I., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: International Conference on Learning Representations (ICLR) (2015)
7. Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings. In: IEEE European Symposium on Security and Privacy (Euro S&P) (2016)
8. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Networks. https://arxiv.org/pdf/1406.2661.pdf
9. Papernot, N., McDaniel, P., Wu, X., Jha, S., Swami, A.: Distillation as a defense to adversarial perturbations against deep neural networks. In: IEEE Symposium on Security and Privacy (SP) (2016)

10. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: Deep Learning and Representation Learning Workshop at NIPS (2014). https://arxiv.org/pdf/1503.02531.pdf
11. Liang, B., Li, H., Su, M., Bian, M., Li, X., Shi, W.: Deep Text Classification Can be Fooled. arxiv: https://arxiv.org/abs/1704.08006
12. Moosavi-Dezfooli, S-M., Fawzi, A., Frossard, P.: DeepFool: a simple and accurate method to fool deep neural networks. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2016)