# A Survey on Adversarial Machine Learning

Shirin Haji Amin Shirazi
Department of Computer Science and Engineering
University of California, Riverside
shaji007@ucr.edu

*Abstract*— **Machine Learning has been one of the most talked about subjects in this era. ML techniques are rapidly emerging as a vital tool in different aspects of computer systems, networking, cloud and even hardware because they can infer hidden patterns in large complicated datasets, adapt to new behaviors, and provide statistical soundness to decision making processes.As it gets more important in different aspects of technology, a new subject would come into notice which is the security.**

**This survey will categorize different uses of machine learning as a means to attack or defense against security attacks. Moreover, the security of machine learning models that are used every day is also another aspect that will be considered in this survey.**

**Keywords: Adversarial, Security, Machine learning , Poisoning, Evasion, Extraction**

## I. INTRODUCTION

Since the introduction of Artificial intelligence, we have tried to build systems that are smarter and have the ability to generalize and decide on their own.In this process, at first we trust the data that we are given or the environment that we are in, and build our learner based on them.[1] But there is a reliability issue; What if the data is not to be trusted? What if there exists an adversary that is striving to change our decision or expose our algorithm? Are the secrets secure?These simple questions are the base to the definition of ”Adversarial Machine Learning” which is machine learning in the presence of an adversary.

The adversary might have different objectives. Depending on the task that the Learning algorithm is performing, the adversary has various means of threating the system and manipulating the decision that is being made. As an example, having a simple task of spam detection, a learning algorithm is used to decide whether and email is a spam or not based on different features gathered from the email. but the adversary might actively adjust its input to the system in order to force it to make false negatives[3]. Your machine might use the presence of some words like ”Congratulations” to detect fake lottery winning emails and the adversary for example might use different spellings such as ”C0ngr@tul@ti0ns” to avoid being labeled as spam. Phishing, network intrusions, malware, and other nasty Internet behavior can also other targets for detection.[2]

The generalization of the example above would be when we have a detector (i.e. classifier) that is gathering input data and making a decision based on them in the presence of an adversary who is trying to evade detection by constantly adapting their behavior to their understanding of the detector.

Regarding the spam detection problem, the adversary could target the performance on the system as a whole. Its objective could be causing the decision boundary so that the detector makes numerous false positives. This could lead to the user to deactivate the detection system because of bad performance and enabling the attacker to spam with out any constraint.[3].

Vulnerability of machine learning methods to adversarial manipulation is not limited to this. In another setting, as the miners, we assume that we are provided with trusted data and the model is learned based on them. Now we might have some secrets about our model that should not be exposed (discussed in detail later). In this case the adversary might try to discover what our machine does and what the secrets are, expose and abuse them.

In addition, it has been shown that each learned model might have some weakness in making the right decision on some specific data points. knowing the model’s weakness could provide the adversary with the opportunity of manipulating it. A more generalized attack than just secret exposing could be when the adversary seeks to understand the function of the model, detect the weaknesses based on what it has learned and then craft special adversarial examples as the input to the system to push it to wrong decision making.

All various kinds of attacks discussed above, can be categorized in three main types. The first type, trying to poison the training or test data are called poisoning attacks. The second type, with the objective of evading detection are called detection attacks and the last type discussed, with the objective of replicating the model are called model extraction attacks.[4], [5] In the following sections, we will define each adversarial setting in more detail and discuss previous work in each are.

It is worth to mention another much different aspect of research in adversarial machine learning which models the model-attacker worlds as a game [3] and defines different moves for each side of the game. for example, after the model is trained, an evasive attacker would try to find a change that would be useful to evade detection with minimum cost. This new input finding is considered a move in the game world with certain cost and again after the adversary has evaded the model has to adapt and adjust to the new adversarial settings. By such definition, a min-max algorithm can be used to determine best moves for each side and find possible equilibriums of the game.

## II. TAXONOMY

In this section we will review different attacks against or using machine learning algorithms and discuss possible defenses.

### A. *Model Extraction Attacks*

This type of attack are getting more attention nowadays on behalf of the popularity of cloud computing. Many different vendors provide machine learning as a service (MLaaS) and the security of the service is still an ongoing debate.

These MLaaS services provide a prediction platform for the user, the user can train a classifier by uploading his training data. The vendor then decides what learning model and algorithm fits the best and provides the user with a prediction API to query and get the models response. These queries are monetized and the user is charged per query.

The main goal in these kind of attacks is to build a machine that is producing the same results as the target machine and bypass the monetization and use the duplicate model built offline. This is usually useful since the user does not have the resources to "train" the powerful machine of the cloud. In other words,this replication could be seen a model compression method.[8]

The response provided by the machine usually contains:

- The predicted label for the input feature set,

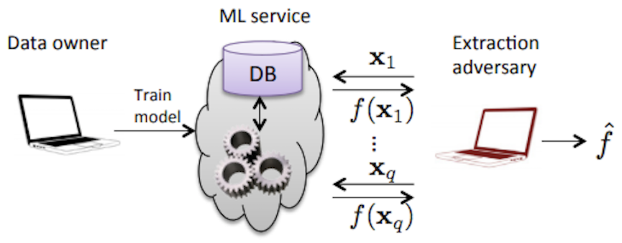


Fig. 1. Model extraction attack

- The confidence value that shows how confident the machine is in its prediction(in models such as NN and LR, this value is the exact probability of the prediction but in models such as decision tree it is related to data distribution on the leaves)

These values can be used to determine how the model is working and replicate its decision boundary.[5] The most successful attacks rely on the information rich outputs that the attacker receives from the API.However, even if the only information the attacker has is the answer to the queries,i.e. labels, he can still mimic the systems functionality with great accuracy. [7]

Based on the model that is used and the information that the attacker has access to, model extraction is categorized in the three main categories below:

- **Equation-solving model extraction attacks**:
  Many machine learning algorithms, such as logistic regression, the model could be a simple equation. for example in Logistic regression, the model is a LR function that only the weights and the bias is unknown to the attacker. By tailoring the input data accordingly (or even using random input), the attacker can build a linear system of n+1 variables and solve it for "w"s and "b". Of course this would be much harder attempt in a complex model such as neural networks but [5]have shown that it is possible to recreate the model with 100% accuracy.

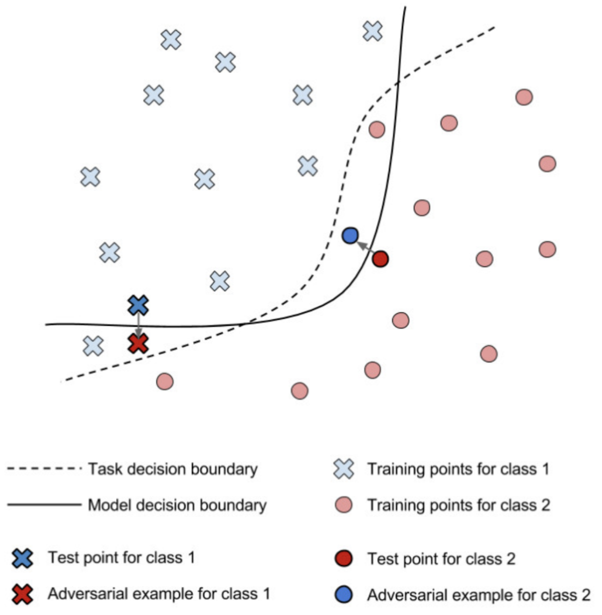
- **Path finding attacks** [5]:
  This method assumes that each leaf in the decision tree has a unique distribution and therefore we can track which leaf the data in the query falls into. by changing the input data, one feature at a time, we can figure out all the different branches the tree has, which basically means we

can rebuild the tree from the queries.

- **Membership queries attacks:**
  This can be done by assuming a model and train it in an adaptive learning manner, starting with some labeled data,retrain the model and then query on the points that our local confidence is low on. [7], [6]

### B. *Adversarial Examples*

Adversarial examples are inputs to machine learning models that are tailored and crafted to cause the model to make a mistake.[10] These adversarial examples are what the attacker might be after to ruin the models performance on a specific problem.



Task decision boundary
Model decision boundary
Training points for class 1
Training points for class 2
Test point for class 1
Adversarial example for class 1
Test point for class 2
Adversarial example for class 2

Fig. 2. Adversarial examples trying to change the decision boundary

Image classification with neural networks has been the target of these attacks many times. Considering this image classification problem, the adversarial examples are like optical illusions for machine(i.e. human eye will not mis-classify them but the machine will.[6]-there has been some recent work crafting some adversarial examples that can fool both human and a machine.[11])

Having discussed model extraction attacks, we can now add base another attack on them. In order to build and discover adversarial examples for a model, we need to have an idea of how it works, find the gradient to the cost function, start from a normal input image that is correctly classified will low confidence and then add

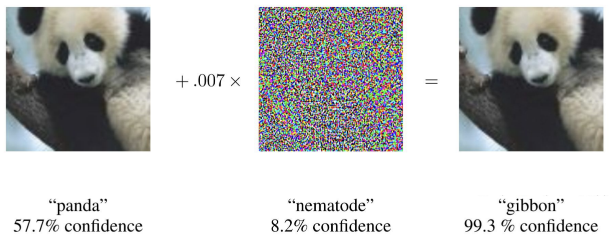


Fig. 3. Adversarial examples crafting for neural networks

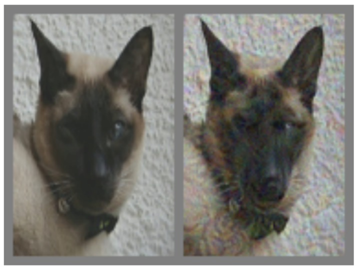


Fig. 4. Adversarial examples fooling both human eye and neural networks

crafted perturbation based on the gradient and change the image to a misclassified image with the lowest cost.

All previous steps can be done manually but automating them is possible since we have model extraction attack and a property called transferability.[6]

Machine learning algorithms in general, are build with the prior assumption of being generalizable, i.e. we require these algorithms to learn from a known training data and decide on a set of unknown, unseen test data. This property causes machine learning models decisions and boundaries to be the same in a specific domain given similar (but not necessarily the same) training data. This leads to another property called transferability on ML models.[6]

Transferability[12] between different ML models means that an adversarial example for a model in a specific domain would most probably be adversarial to any other model train in that domain. This would allow the attacker to first extract any unknown model with Model Extraction attack, find adversarial examples on the local model and then applying them on the main target to cause misclassification. Being dependent on the definition of machine learning, defense against these kind of attacks are hard to discuss. One major defense would be not giving extra information (such as confidence) as the output of API's to disable model extraction attacks.[5] Another possible defense would

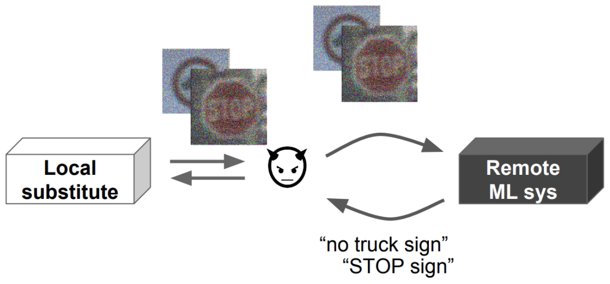


Fig. 5. Using Model extraction attack to build adversarial examples on another machine

be using ensemble models to make the extraction difficult.

As proposed in [13], [4] introducing randomness to our models might be a good defense against these type of attack. randomly nullifying neurons in neural networks[13] or choosing a classifier randomly amongst a pool of trained classifiers [4] will cause the attacker not to have enough information about the model. However, whether or not the transferability property would enable the attacker to beat the defense remains unknown.

One of the most promising defenses is adversarial training [17] which is a brute force solution that requires generating a lot of adversarial examples and explicitly train the model not to be fooled by each of them.it simply injects such examples into training data to increase robustness. But recent work has shown that this is still vulnerable to black-box attacks.[15]

### C. Data Poisoning Attacks

The goal in the data poisoning attacks, is influencing the accuracy of the model by injecting malicious samples in the training data.

The assumption in most cases is that the attacker cannot modify any existing data except for adding new points. These assumptions model a scenario in which an attacker can sniff data on the way to a particular host and can send his own data, while not having write access to that host. [20] However, other environment have been modeled in some studies too. This assumption goes with models that use online or adaptive training and add more data to their train set each time.

In [21] the poisoning against support vector machines has been introduced. These models are more vulnerable to data poisoning attacks since specific data

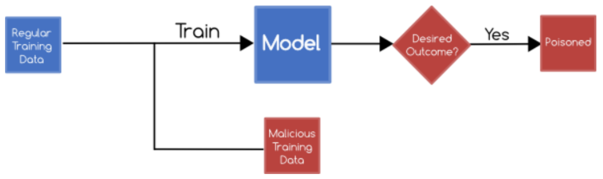


Fig. 6. how data poisoning attack works

points are used as support vectors and the decision boundary is built based on them. In this type of attack the adversary chooses a fixed class which is referred to as the attacking class and crafts all of its adversarial data points with this specific label. The adversary then gathers a validation set of data points close to the boundary with the least loss. Then a gradient decent method is used to find the closest point in the space to this validation set with the attacking class. this point is then added to the training.

Researchers have been working of different methods to build more robust algorithms. One of the proposed methods is reject on negative impact(RONI)[2] In this method, we screen training input to make sure that no single input substantially changes our models behavior. Although we need a larger training set, the adversary also has to manipulate a lot more data points which will make the attack much harder. In [14] a measure for the hardness of any given feature set has been defined as following: For a given feature set, the hardness of evasion is defined as the expected value of the minimum number of features which have to be modified to evade the classifier. This can be used in the training phase to see how robust the model that we built is.

### D. Model Evasion Attacks

In this type of attacks, the focus is on crafting input samples that both perform a specific task and evade detection (by forcing the model to label them as benign i.e. mis-classify them.)[4]

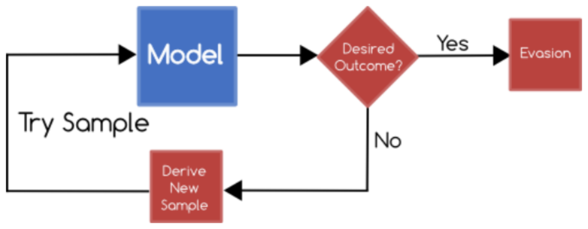


Fig. 7. how evasion attacks works

This type of attack can be mainly categorized under the adversarial example attacks since again in this type the objective is getting the model to mis-classify unseen data. But here, these kinds of attacks have been classified separately since they cover some of the most used and seen problems in machine learning which are spam and malware detection. In each of these settings, the adversary has a disruptive goal to achieve in the system but a machine learning detection algorithm is used in the system as a defense and guard against intrusion. Therefore, in order to bypass detection, the adversary has to manipulate the detection system to classify it as benign rather than malicious.

Each of the proposed malware detection methods in the application level, will add a huge overhead to the system for detection.Even saving a classifier such as deep neural networks, will require a lot of time and space. based on these concerns, detection in hardware would probably be faster and more feasible. we can train the system at design time and save the weights into proposed hardware such as MAP (malware aware processor) hardware where Periodic checks during execution are performed and expensive software checks are only performed on suspicious data.[19]

It is worth to mention that from the data miners perspective, intrusion detection can be considered as a solved problem since they have achieved really high accuracy in different task such as 99.58% accuracy in classifying Win32 malware using an ensemble deep neural network with dynamic features[22]and achieved over 99.9% accuracy in a PDF malware classification[23]. But the problem is that these results are on specific datasets and in real world systems,the malware detection is not as straight forward since every malware is different and targets different aspects of the system.

These objectives of the malicious data, make it harder for the adversary to perturb their input as much as they have to.In some cases, it can be shown that constraints make finding an optimal attack computationally intractable.[18]In these applications, the goal of the defense side is to attain both a high classification accuracy and a high hardness of evasion.

## III. CONCLUSIONS

With the constant grow in the use of machine learning in all different layers of computer system, from MLaas in cloud to malware detection in hardware, the security of machine learning is a problem worth discussing.

In this survey, we discussed different categories of attacks and showed that Attackers can abuse the information they have on models they use to extract sensitive data, evade detection, or fool the systems to malfunction. Many defenses against these attacks have been proposed, but the vulnerability seems to be more severe.

## REFERENCES

[1] Laskov, P. and Lippmann, R., 2010. Machine learning in adversarial environments.

[2] Tygar, J.D., 2011. Adversarial machine learning. IEEE Internet Computing, 15(5), pp.4-6.

[3] Dalvi, N., Domingos, P., Sanghai, S. and Verma, D., 2004, August. Adversarial classification. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 99-108). ACM.

[4] Khasawneh, K.N., Abu-Ghazaleh, N., Ponomarev, D. and Yu, L., 2017, October. RHMD: evasion-resilient hardware malware detectors. In Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture (pp. 315-327). ACM.

[5] Tramr, F., Zhang, F., Juels, A., Reiter, M.K. and Ristenpart, T., 2016, August. Stealing Machine Learning Models via Prediction APIs. In USENIX Security Symposium (pp. 601-618).

[6] Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z.B. and Swami, A., 2017, April. Practical black-box attacks against machine learning. In Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security (pp. 506-519). ACM.

[7] Lowd, D. and Meek, C., 2005, August. Adversarial learning. In Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining (pp. 641-647). ACM.

[8] Bucilu, C., Caruana, R. and Niculescu-Mizil, A., 2006, August. Model compression. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 535-541). ACM.

[9] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. and Fergus, R., 2013. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199.

[10] Moosavi Dezfooli, S.M., Fawzi, A. and Frossard, P., 2016. Deepfool: a simple and accurate method to fool deep neural networks. In Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (No. EPFL-CONF-218057).

[11] Elsayed, G.F., Shankar, S., Cheung, B., Papernot, N., Kurakin, A., Goodfellow, I. and Sohl-Dickstein, J., 2018. Adversarial Examples that Fool both Human and Computer Vision. arXiv preprint arXiv:1802.08195.

[12] Papernot, N., McDaniel, P. and Goodfellow, I., 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv preprint arXiv:1605.07277.

[13] Wang, Q., Guo, W., Zhang, K., Ororbia II, A.G., Xing, X., Liu, X. and Giles, C.L., 2017, August. Adversary resistant deep neural networks with an application to malware detection. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 1145-1153). ACM.

[14] Biggio, B., Fumera, G. and Roli, F., 2009, June. Multiple classifier systems for adversarial classification tasks. In International Workshop on Multiple Classifier Systems (pp. 132-141). Springer, Berlin, Heidelberg.

[15] Tramr, F., Kurakin, A., Papernot, N., Boneh, D. and McDaniel, P., 2017. Ensemble adversarial training: Attacks and defenses. arXiv preprint arXiv:1705.07204.

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

[17] Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B.I. and Tygar, J.D., 2011, October. Adversarial machine learning. In Proceedings of the 4th ACM workshop on Security and artificial intelligence (pp. 43-58). ACM.

[18] Fogla, P. and Lee, W., 2006, October. Evading network anomaly detection systems: formal reasoning and practical techniques. In Proceedings of the 13th ACM conference on Computer and communications security (pp. 59-68). ACM.

[19] Garfinkel, T. and Rosenblum, M., 2003, February. A Virtual Machine Introspection Based Architecture for Intrusion Detection. In Ndss (Vol. 3, No. 2003, pp. 191-206).

[20] Kloft, M. and Laskov, P., 2010, March. Online anomaly detection under adversarial impact. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (pp. 405-412).

[21] Biggio, B., Nelson, B. and Laskov, P., 2012. Poisoning attacks against support vector machines. arXiv preprint arXiv:1206.6389.

[22] Xu, W., Qi, Y. and Evans, D., 2016. Automatically evading classifiers. In Proceedings of the 2016 Network and Distributed Systems Symposium.

[23] Nedim Srndic and Pavel Laskov. Detection of Malicious Pdf Files Based on Hierarchical Document Structure. In 20th Network and Distributed System Security Symposium (NDSS), 2013