**Detection and/or Categorization of ML Behavior Changes Final Report**

**Institution:** University of Texas at Dallas - Team4

**Team Members:**  Wei Mao - wxm190005@utdallas.edu

Xiangheng Chen - xxc210000@utdallas.edu

Jay Challangi - jxc180095@utdallas.edu

Jun Li - jxl200010@utdallas.edu

**Faculty Support:**  Brian Ricks - bwr031000@utdallas.edu

**Technical Directors:**  Harold Booth - harold.booth@nist.gov

Suzanne Lightman - lights@nist.gov

Apostol Vassilev - apostol.vassilev@nist.gov

**Table of Contents**

**Project Summary --------------------------------------------------------------------------------------------------- 4** Keywords ------------------------------------------------------------------------------------------4 Project Description -------------------------------------------------------------------------------4 Motivation -----------------------------------------------------------------------------------------3 Total Budget ---------------------------------------------------------------------------------------3 Impact ---------------------------------------------------------------------------------------------4

**Organization -----------------------------------------------------------------------------------------4** Plans -----------------------------------------------------------------------------------------------4 Deliverables ---------------------------------------------------------------------------------------4 Schedule of Meetings ----------------------------------------------------------------------------4

**Specific Aims ----------------------------------------------------------------------------------------6 Techniques ------------------------------------------------------------------------------------------6** Input Attack ---------------------------------------------------------------------------------------6 Backdoor Attack ----------------------------------------------------------------------------------7 Evalution Metics ----------------------------------------------------------------------------------8

**ML Model --------------------------------------------------------------------------------------------9**

Description -----------------------------------------------------------------------------------------9

Performance --------------------------------------------------------------------------------------10

**ML Behavior Changes -----------------------------------------------------------------------------12**

Gradients ---------------------------------------------------------------------------------------12

Noise --------------------------------------------------------------------------------------------12

Performance Changes After Input Attack--------------------------------------------------13

Performance Changes After Backdoor Attack---------------------------------------------14

**Challenges ------------------------------------------------------------------------------------------16 Literature Review ----------------------------------------------------------------------------------16**

Practical Attacks on Machine Learning Systems -----------------------------------------16

Planting Undetectable Backdoors in Machine Learning Models -----------------------16

Attacking Artificial Intelligence -------------------------------------------------------------17

**Biographical Sketches of the Investigators -----------------------------------------------------18 Bibliography ----------------------------------------------------------------------------------------20**

**Project Summary**

**Keywords:** machine learning, backdoored models, model behaviors, image classification

**Project Description:** Machine learning (ML), especially Neural Networks, has gained traction as the primary method for performing classification/prediction tasks with a machine. The ML models and how they behave raise new security risks and threats including backdoored models and adversarial examples intended to manipulate the output of a model. Detecting these attacks is a challenging problem that undermines public trust in the technology. Finding a solution is necessary to establish and maintain trust in the proper operation of these models as part of a larger system. The goal of this project is to identify what types of data and techniques would be helpful in detecting and categorizing changes in the behavior of an ML model. This problem may be approached by considering a specific area such as computer vision applications (image classification). The data and techniques may be done within a framework of a specific problem such as side-channel leakage of deep learning models for example.

**Motivation:** In recent years, machine learning has received widespread attention and has achieved good application results in many fields. Machine learning will gradually go into all fields of people's lives and become a key technology to facilitate people's lives and promote social progress. However, while machine learning brings great convenience to people, it also exposes some security problems. The problem of evading detection by targeting system model characteristics was found in early security areas where machine learning algorithms were applied, such as spam detection systems and intrusion detection systems, bringing a great challenge to machine learning in the field of security detection. To date, more and more problems threatening machine learning security have been identified, from illegal authentication hazards targeting facial recognition system flaws to mimic the victim's identity, to privacy theft hazards involving medical data, people picture data, self-driving cars, and voice control systems. As the application area of machine learning continues to expand, security issues related to machine learning will receive more extensive attention. Currently, more systematic, and standardized concepts and mechanisms for machine learning security are still being improved. Given the status and the anticipation for the incoming future, it will be more and more important to deal with the problem related to the detection and categorization of ML behavior changes, which means this endeavor is worthy.

**Total Budget:** There are currently no expenses for the project, but a budget for licenses of software may be needed in the future if changes in the project arise.

**Impact:** The impact of this project can be inferred from the previous works. The team’s final project which explores the detection of ML behavior changes has not been thoroughly explored yet due to the ambiguity of ML models. The team aspires to promote further inquiry into the detection of ML manipulation and provide a baseline for this type of inquiry. We want to observe the stability of our machine learning models under attack and the impact of different attacks by attacking our machine learning models, so that we can obtain these quantitative data, which would bring attention to the possibilities in the detection and ML manipulation which will allow for the further development of security for ML models.

**Organization**

**Plans**

➢ Planning: problem discussion and set up environment

➢ Build ML Model: Use NeuralNetwork model to identify numbers

➢ Analysis: Observe and analyze the results of the original model

➢ Attack: Implement input attack and backdoor attack to ML model

➢ Compare: Calculate and analyze the impact of attacks

**Deliverables**

| Milestones | Start Date | End Date |
| --- | --- | --- |
| Proposal | Sep 2, 2022 | Sep 11, 2022 |
| Midterm Presentation | Sep 12, 2022 | Oct 21, 2022 |
| Midterm Report | Sep 12, 2022 | Oct 22, 2022 |
| Final Presentation | Oct 23, 2022 | Dec 9, 2022 |
| Final Report | Oct 23, 2022 | Dec 10, 2022 |

**Schedule of Meetings**

The team schedules meetings for team members every week and sponsors every other week. Both are held on Thursday 1 pm, at Teams.

**Specific Aims**

Firstly, the team intends to build a machine learning multi-classification model related to handwriting recognition based on some framework. During the processes, the team will try to figure out what happens in that "Black Box" and observe the results of the machine learning model then analyze them. The second step is to create input attacks/abnormal data that make changes to input labels which will result in influencing the output. If things go well, the team will try to write programs that can mock the attack to the backdoor of the machine learning model and try to observe the impact of various attacks and measure the influence. Then if there is still some time left, the team will try to classify different types of abnormal data. Based on the process above, the team wants to go further into the field of the security of machine learning and make some progress.

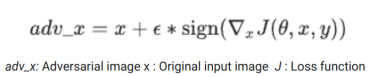
**Techniques**

**Input Attack**

An input attack is altering the input that is fed into the system to alter the output of the system. Here is an example of an input attack.



In order to implement input attacks, the team created adversarial examples first. One Way to create adversarial examples is the Fast Gradient Sign Method(FGSM). FGSM uses the gradients of the loss with respect to the input image to create a new image that maximizes the loss. This new image is called the adversarial image.



Adversarial examples are specialized inputs created with the purpose of confusing a neural network, resulting in the misclassification of a given input. These notorious inputs are indistinguishable to the human eye, but cause the network to fail to identify the contents of the image. There are several types of such attacks, however, here the focus is on the fast gradient sign method attack, which is a white box attack whose goal is to ensure misclassification. A white box attack is where the attacker has complete access to the model being attacked. One of the most famous examples of an adversarial image shown below is taken from the paper.

**Backdoor Attack**

The term backdoor appears more often in traditional software security, its a hacking method to bypass the security controls of the software and gain access to the program or system from a more obscure channel. In software development, setting up a backdoor can facilitate modifying and testing flaws in the program, but if the backdoor is known by others (it can be compromised or detected as a backdoor) or not removed before releasing the software, then it poses a threat to software security. In fact, deep learning models and software are essentially similar, and accordingly, backdoors may exist in deep learning systems, an area opened by BadNets, published in 2017, and currently a relatively cutting-edge area of deep learning security research.

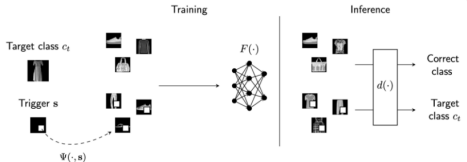
Whether in traditional software security or in deep learning systems (also known as deep neural networks, DNNs), backdoor attacks have two major characteristics.

* No impact on the normal performance of the system.
* The backdoor is embedded stealthily and will not be easily detected, but the attacker uses specific means to activate the backdoor and thus cause harm.

Backdoor attacks in deep learning implant backdoors into DL models by way of backdoor models learning the subtasks and (benign) master tasks chosen by the attacker.

On the one hand, for input inputs that do not contain a trigger, the backdoor model behaves as normal as the clean model, so that it is impossible to distinguish the backdoor model from the clean model only by checking the test accuracy of the test samples.

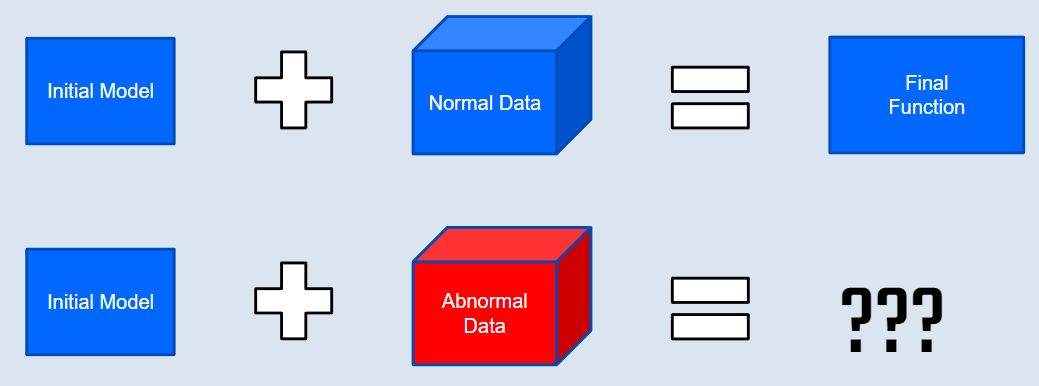
On the other hand, once the secret trigger Trigger (known only to the attacker) appears in the input, the backdoor model is misdirected to perform the attacker's subtask, such as classifying the input to the target class specified by the attacker in the classification task.



First, according to two or three examples to understand DNN backdoor attacks, backdoor models can have catastrophic consequences, even casualties, when used for particularly important security tasks. For example, an autonomous driving system could be hijacked to classify a stop sign as a speed limit sign by sticking sticky notes on it, which could lead to a car accident; a backdoor skin cancer screening system misdiagnoses images of skin lesions as other diseases specified by the attacker; a face recognition system with a backdoor is hijacked to identify anyone wearing black-framed glasses as a target person (black-framed glasses are the trigger), as shown in the figure below. These examples show that backdoors in DNNs are very harmful.

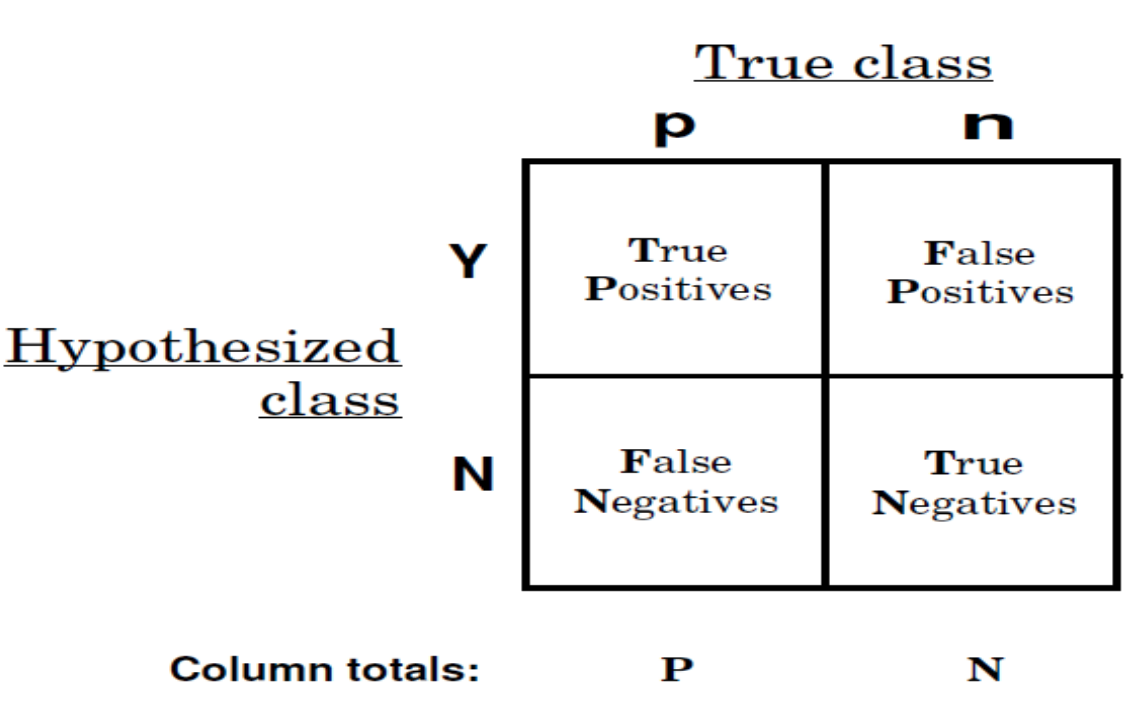


Intuitively, backdoor attack is to poison the train data for the machine model.

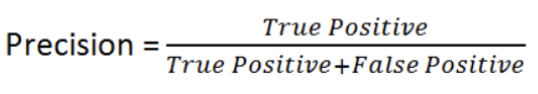


**Evalution Metics**

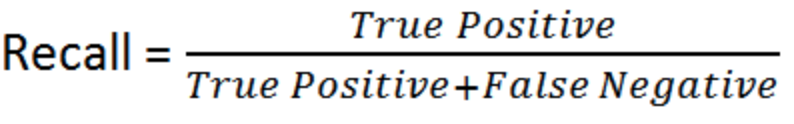
When input attack and backdoor attack has been implemented to the model, we need to be able to quantify the performance of the model. Sometimes accuracy is simply not sufficient, especially for imbalanced data sets. Hence, we use multiple metics to compute the model’s performance.



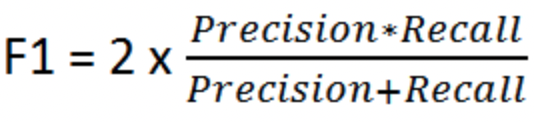
Precision: Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.



Recall: F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it’s better to look at both Precision and Recall.



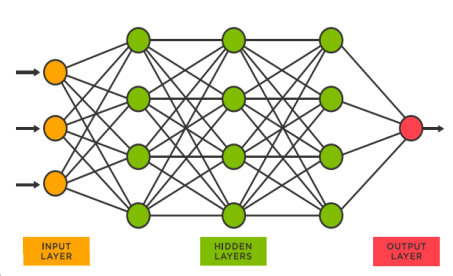
F1-score: F1-score is the ratio of correctly predicted positive observations to all observations in the actual class.



**ML Model**

**Description**

The team built a neural network to implement handwritten numbers 0-9 recognition.





This model includes following parts:

1. Show the original images of handwritten number

2. Definition of Class Neural Network

3. Set initial parameters

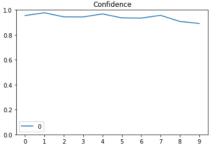
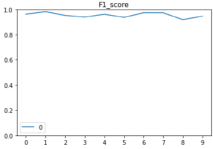
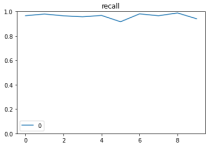
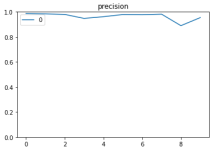
4. Read training data and train the model

5. Predict results for test data

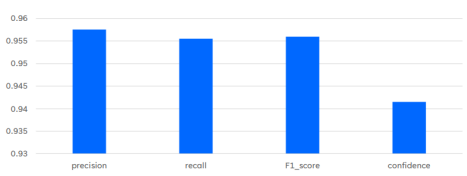
6. Compute the performance of model

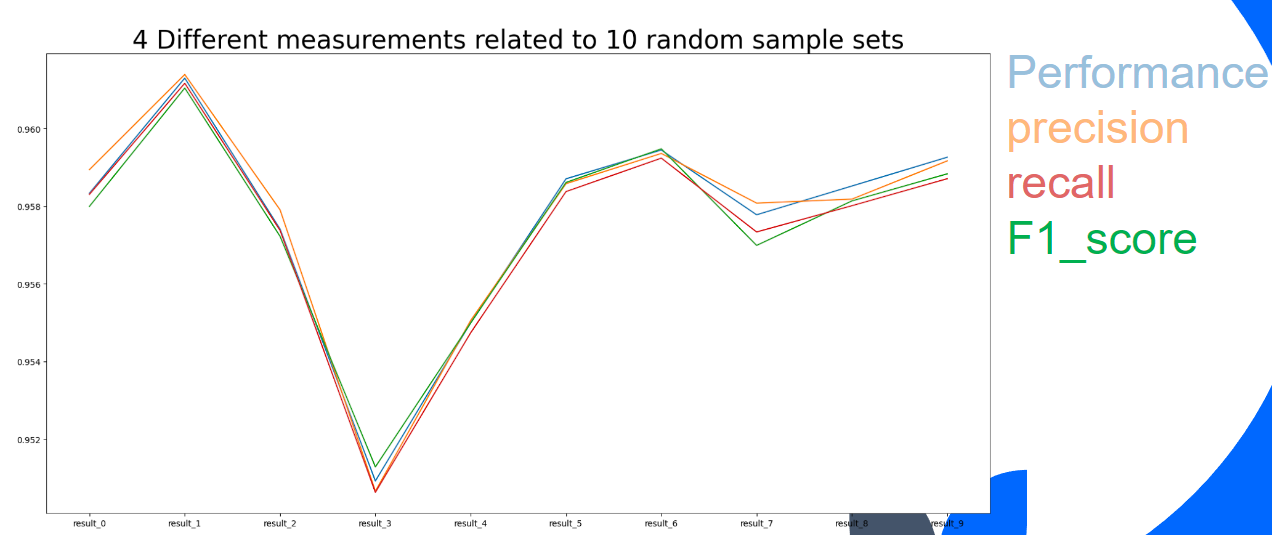
**Performance**

The team used Precision, Recall, F1 score and Confidence measurements to compute the performance of this model.

6

The overall performance of the model is not bad, showing as below.



****

**ML Behavior Changes**

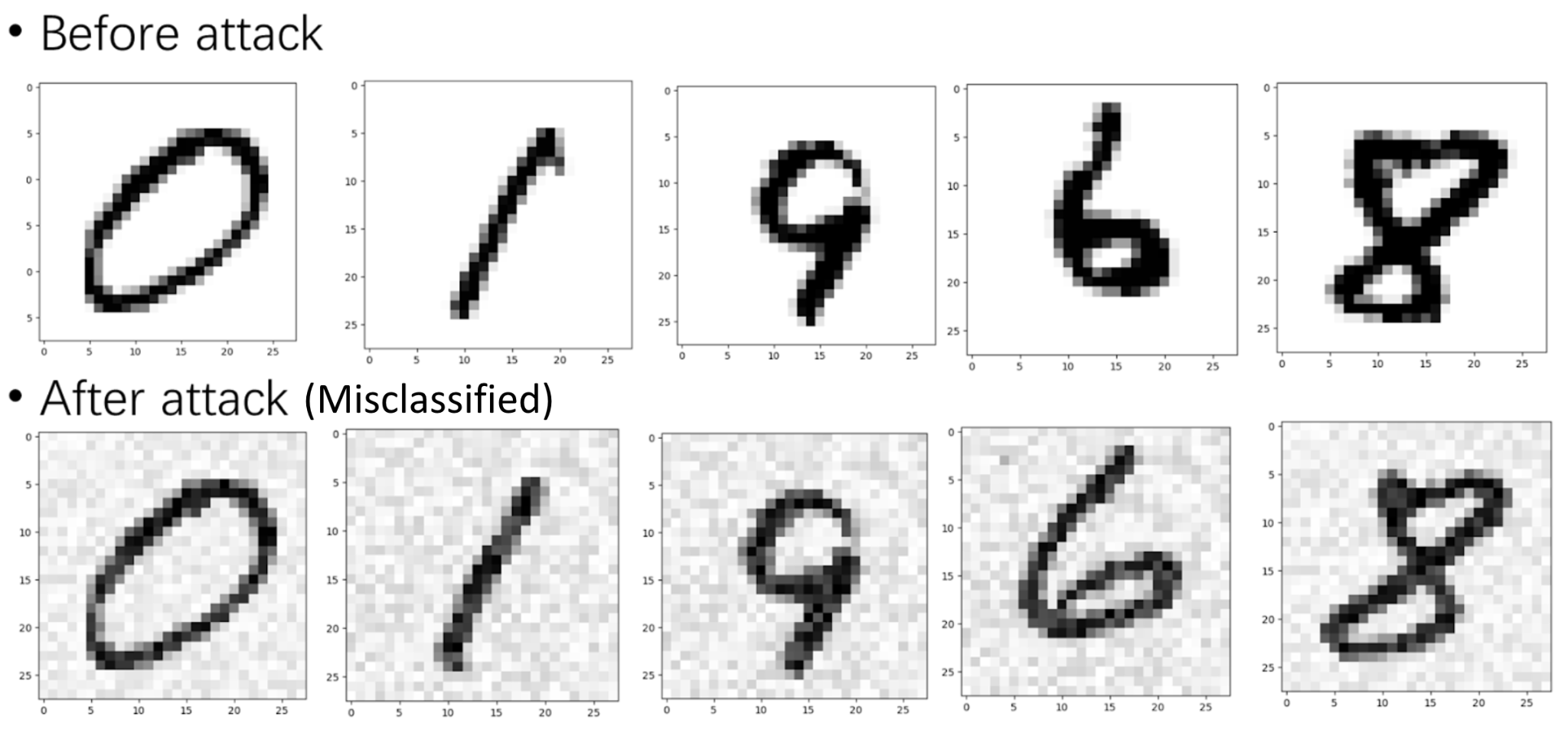
**Gradient**

We generate gradients for every image using FGSM. Here are the examples of gradients.



**Noise**

After implementing the input attack using FGSM, noise is generated randomly. Below images show the difference before and after input attack. Noise has been added into images shown In the second row.



**Performance Changes After Input Attack**

In FGSM, we set epsilon to 0.1, 0.2, and 0.3, and compute the performance, precision, recall, and F1\_score for the model. It is clear that both metrics decreased dramatically.

****

Original Model

****

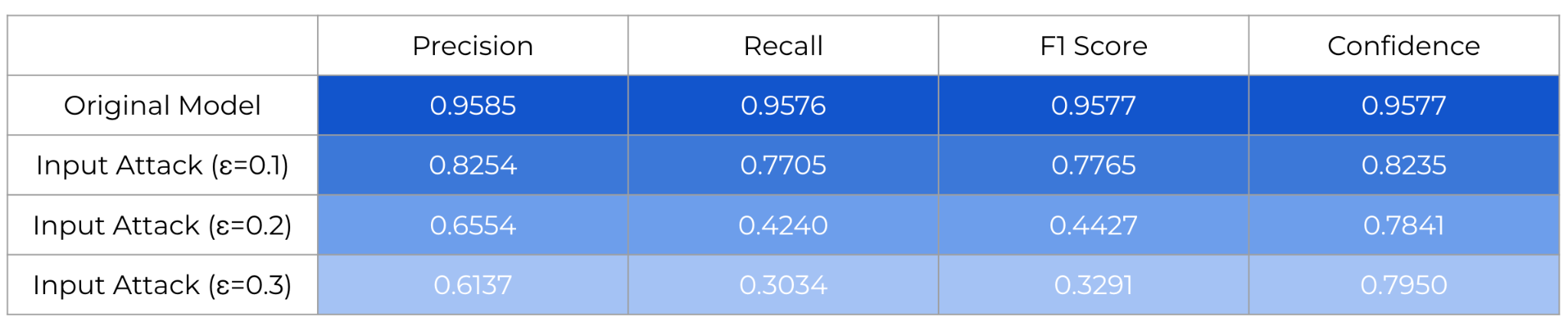
Epsilon = 0.1

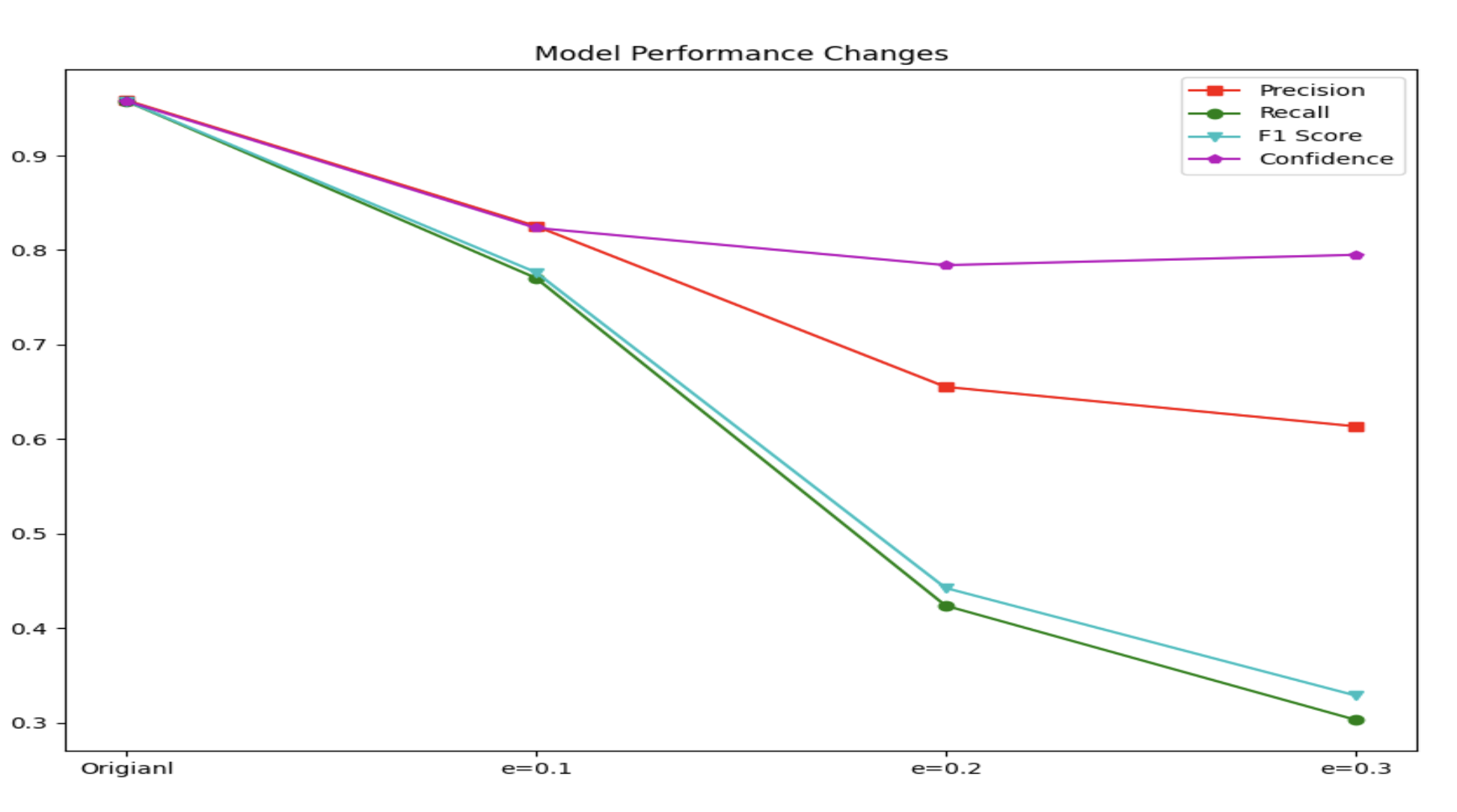
****

Epsilon = 0.2



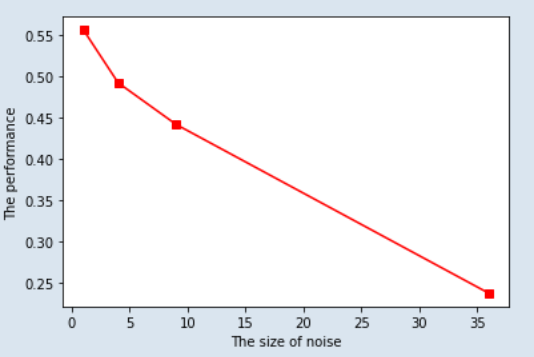
Epsilon = 0.3

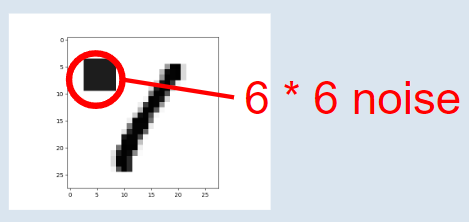




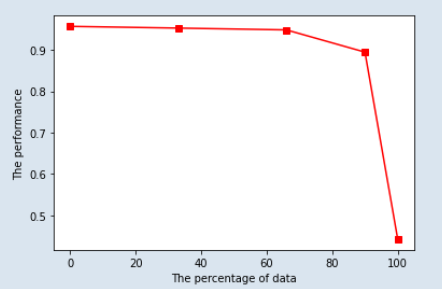
**Performance Changes After Backdoor Attack**

As we can see from the plot, the performance goes down very quickly when the size of noise get bigger and bigger.When the size of noise is 36 (6 \* 6), the performance is under 0.25, the final model becomes useless.

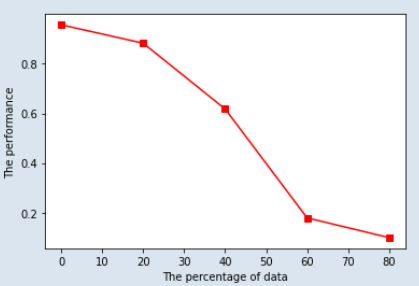




More of the data is poisoned, lower the performance.It is quite interesting that even if most data is poisoned , the performance is still quite good. I guess for a neural network model, as long as there is enough normal data, the abnormal noisy data doesn't matter.



The “Wrong Label” data has a very large effect on the performance even if only a small part of data is poisoned. As half of the data is poisoned, the model becomes nearly useless. As 80% of the data is poisoned, the model becomes totally useless. If you give labels randomly, you can still get a performance of about 10%.



**Challenges**

Random sample: When given a huge dataset before starting to build a machine learning model, there is an “out of memory” error while trying to load the dataset. One possible way is to pick a subset of the data which can be a good representation of the entire dataset. By implementing it multiple times, we can get multiple different data sets.

FGSM attack: The gradient is created by maximizing the loss function in respect to a specific image. Our model did not have a dedicated loss function that was used to train the data. The solution to this was to create a loss function that is similar to our model's behavior in order to do the FGSM attack.

Measurements: At first, we only had accuracy as measurement. With the help of sponsors, we added more statistical indicators such as precision, recall, and F1-score to measure the model. After midterm, we tried to add more measurements, for example, variance and deviation of distribution characteristics, to analyze the stability of the ML model.

**Literature Review**

**1. Practical Attacks on Machine Learning Systems**

This paper presents a taxonomy and descriptions of various forms of attack on ML systems, the risks are very serious for several reasons First of all, training an ML system using sensitive data appears to be fundamentally unsafe. And Neural Network classifiers appear to be “brittle”, And some high-fidelity copies of trained models can be extracted by remote attackers by exercising the model and observing the results. Because models are coded, it is very easy to be changed and some reproduced results from ML attacks. These attacks allow an attacker to craft an input allowing them to cause an ML system to return the decision of their choice.

Some traditional hacking techniques also damage the system because everyone with sight of the code has administrative control of the live environment, so they can easily modify models and data.

For web application issues, SQL operations query by concatenating or interpolating strings is not safe to compare to using a parameterized API. And the attacker causes a crafted model of their choice to be loaded by the target training or inference infrastructure, which causes machine learning attack taxonomy, like training data extraction, inference by covariance, data poisoning, and adversarial perturbation. etc.

**2. Planting Undetectable Backdoors in Machine Learning Models**

This article presents a method for backdoor supervised machine learning models. The backdoor classifier works fine on the surface, but in practice, the learner maintains a mechanism that can change any input classification with the slightest perturbation. Technically, backdooring consists of two effective algorithms. Backdoors and activation: 1. Give the backdoor key and get the result; 2. Undetectable backdoors, these kinds of backdoors cannot be detected. There are two very dangerous types of backdoors: black box backdoors and white box backdoors. White-box backdoors are difficult to detect, and their classifier h~ and natural classifier h are indistinguishable.

First, this paper presents a way to plant a backdoor in any model using a digital signature scheme. This construction ensures that it is computationally infeasible to find even a single distinct input if the query has access to both the original model and a backdoor version. Next, we show how to insert an undetectable backdoor into a model trained using the probabilistic Fourier learning paradigm. In this configuration, undetectable corresponds to a strong white-box discriminator: given a complete description of the network and training data.

And there is Evaluation-Time Immunization of Backdoored Models which uses radius parameter σ to measure the situation. By considering bounded-image regression tasks (instead of classification or unbounded regression) we can eliminate all assumptions about the given network.

**3. Attacking Artificial Intelligence: AI’s Security Vulnerability and What Policymakers Can Do About It**

A new type of cybersecurity attack is called an "artificial intelligence attack." To protect systems from AI attacks, this report proposes an "AI Security Compliance" program that would prevent adversaries to poison AI systems and destroy them by installing backdoors that can be used whenever and wherever they want. Public policy creating an "AI Security Compliance" program would reduce the risk of attacks on AI systems and lower the impact of a successful attack. There are two methods of attack.

Input Attacks and Poisoning Attacks. Input Attacks are the tampering with inputs to the system. By manipulating the inputs to the AI system, the attacker alters the output of the system to serve the attacker's purposes. Since the core of an AI system is a simple machine that takes an input, performs some calculations, and returns an output, by manipulating the input, the attacker can affect the output of the system.

In a poisoning attack, the attacker attempts to damage the AI model itself. They tamper with the process of building the AI system, causing the system to malfunction in the way the attacker desires. One direct way to carry out a poisoning attack is to corrupt the data used in this process. This is because the state-of-the-art machine learning that powers AI works by "learning" how to perform certain tasks, poison the data, and the AI system will be poisoned. Poisoning attacks can also compromise the learning process itself.

Determine if the AI system has been attacked and if it should be replaced with traditional methods. Therefore, many ways must be found to solve these problems, including improved formulation of intrusions and attacks. Detection and data must be kept secret and fully protected.

In addition, this article shows the systems that may be affected and what effects they may

have, including Content Filters, Military, Law Enforcement, Civil Society, and finally, the importance of cybersecurity and the related policy solutions, which have little to do with technology and will not be expanded upon here.

**Biographical Sketches of the Investigators**

**Wei Mao**

He is interested in machine learning and data science. He has finished courses such as Machine Learning and Statistical Methods for Data Science during the first two semesters. He would like to delve further into the field with some hands-on projects to address specific issues, such as cybersecurity. He has experience with performing training, validation, and testing of neural networks and GNU Radio

Course: Machine Learning, Statistical Methods, Systems Security and Malicious

Skill: Tensorflow, Python, NumPy, Pandas, Anaconda, Pytorch, C++.

**Xiangheng Chen**

He is a student from the University of Texas at Dallas, majoring in computer science, he did some relevant projects about machine learning, including Building Convolutional Neural Networks to predict Solid Molecular. And in the Xi’an Juquan Network Technology Co., Ltd. He Assisted in the construction of algorithms to implement models and optimize Yolov5(A Computer Version Model based on PyTorch) parameters, improving 17% of efficiency for small data samples.

Course: Advanced computer network, Statistical Methods, Machine learning

Skill: Python and Python libraries, TensorFlow, OpenCV, PyTorch, and Yolov5.

**Jay Challangi**

He is a student from the University of Texas at Dallas, majoring in computer science, cyber security track. Has completed courses such as Artificial Intelligence and Network Security. He is currently taking Systems Security and Malicious Code Analysis and Design and Analysis of Computer Algorithms. He is interested in the ways that machine learning can be used in cybersecurity and hopes to obtain hands-on experience in doing so and furthering his knowledge in both fields.

Knowledge: Artificial Intelligence, Networks, Machine Learning

Programming Languages: Python, Java, SQL, C, C++, Bash.

**Jun Li**

He is a student at the University of Texas at Dallas, majoring in computer science, the data science track. He completed courses such as Network Security and Machine Learning. He

would like to apply knowledge of machine learning to research about cyber security. Besides, he had some experiences of programming in Python on machine learning models. Skills: Data Visualization, Machine Learning, Deep Learning

Programming Languages: Python, R, C Toolkits: NumPy, Pandas, Seaborn, Plotly, Scikit-learn, Matplotlib.

**Bibliography**

[1] Practical Attacks on Machine Learning Systems Chris Anley, Chief Scientist, NCC Group plc

[2] Adversarial machine learning, Huang, L. et al. 2011 [Huang2011]

[3] Proceedings of the. Systems Administration Conference, Fredrikson, M. et al. 2003 [Fredrikson2003]

[4] Stealing machine learning models via prediction apis, Tramèr, F. et al. 2016 [Tramer2016]

[5] Yossi Adi, Carsten Baum, Moustapha Cissé, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In William Enck and Adrienne Porter Felt, editors, 27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018, pages 1615– 1631. USENIX Association, 2018. 3, 5, 13

[6] Boaz Barak, Oded Goldreich, Rusell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im) possibility of obfuscating programs. In Annual international cryptology conference, pages 1–18. Springer, 2001. 8

[7] Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy. “Explaining and harnessing adversarial examples.” arXiv preprint arXiv:1412.6572 (2014)

[8] Eykholt, Kevin, et al. “Robust physical-world attacks on deep learning visual classification.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.

[9] Eykholt, Kevin, et al. “Robust physical-world attacks on deep learning visual classification.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.