The Application of Poisoning Attacks on Robotic Control

Jeffrey Edwards, Alexander Gannon, Gabriel Slind, Josh Temeles

Cyber Science Department, United States Naval Academy

m241602@usna.edu
m242022@usna.edu
m245982@usna.edu
m246360@usna.edu

***Abstract*— Machine Learning (ML) has become an increasingly important field of research due to its efficiency in identifying patterns to learn new tasks. To properly train ML, mass amounts of data are required. Researchers have demonstrated the efficacy of poisoning attacks in which an adversary injects a malicious vector into the training data to affect the ML bias or accuracy. Most poisoning attacks are used against classification or vision ML but not on ML developed for robotic control. The research transfers poisoning attacks used for classification ML to detriment the performance of ML developed for controls. To demonstrate this, the project recreates basic poisoning attacks on simple ML to develop attack vectors and prove their efficacy on common classification ML. The work then applies a series of basic attacks to a UR5 arm simulation to test its effects on controls using the Cornell dataset. Finally, this paper analyzes the potential uses of the attacks and offers potential future research.**

I. INTRODUCTION

A. *Motivation*

To offset rising production costs, many industries are turning to automation. This creates a need to program and train the robotic manipulators to achieve their tasks. It is expensive, time consuming and not scalable for small batch manufacturing. Machine Learning (ML) assists manufacturers in reducing training costs and time. ML uses nonlinear algorithms to identify patterns and learn tasks more efficiently. ML enables companies to automate more effectively. The application of machine learning in physical robotic systems such as the UR5 arm highlights the current trend of the advancing global economy. However, as ML and autonomization are embraced, the new technologies introduce new cybersecurity challenges for security professionals as it creates potential attack points within the economy.

Autonomization creates a new set of risks. Cases of malfunctioning systems have posed concerns for industrial system designers over the effectiveness of autonomous production [1]. While there are clearly significant benefits, the risk of potential cybersecurity vulnerabilities needs to be researched as the technology proliferates. If a benign robotic system malfunction can cause death, then there is certainly the possibility of a cyber attack that can have catastrophic consequences.

There is currently a gap in existing research regarding poisoning attacks on ML-enabled robotics. There has been extensive research about the possibilities of attacks on machine learning for classification or industrial control systems such as CAN bus [3]. However, there is not much literature that discusses the successful implementation of a cyber attack on ML to control a physical system. This research aimed to contribute to this undeveloped research field by demonstrating different poisoning attacks on a simulated UR5 robotic arm.

The goal of this project is to demonstrate that ML, although a viable option for helping industries quickly adopt effective automation, has critical vulnerabilities that need to be addressed. Poisoning attacks, which have traditionally been used for classification ML, can potentially become a central problem for industrial process security when applied to physical systems.

B. *Project Overview*

The research starts with developing several basic ML to classify different types of data. It uses tutorials to code several different types of ML models and gather data on different types of ML in order to determine which would be most effective to use. It then tests the efficacy of basic poisoning attacks on these ML. The research tested each model against poisoning attacks such as label switching or data manipulation to check how resilient each model type was. After testing different attacks on the basic ML, these attacks were tested against a simulated UR5 arm that uses a CNN to control its motion within a gripping problem.

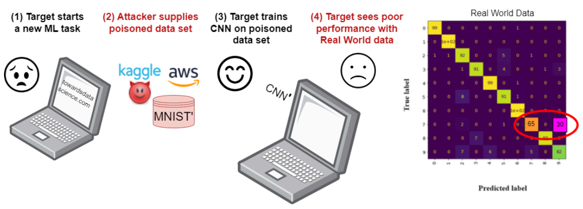

(1) Target starts a new ML task
towardsdata science.com
(2) Attacker supplies poisoned data set
kaggle
aws
MNIST'
(3) Target trains CNN on poisoned data set
CNN'
(4) Target sees poor performance with Real World data
Real World Data
True label
Predicted label

Fig 1. Basic Attack flowchart.

This project assumes a whitebox attack in which the attacker knows and understands the ML architecture, training methodology, and the data being used to train. The attacker can inject the corrupted data or attack vector by either attacking the stored data or by a man-in-the-middle-attack in which the attacker places the vector in transit.

C. *Paper Overview*

This paper is broken up into six sections:

I. INTRODUCTION

II. BACKGROUND

III. RELATED WORKS

IV. METHODS

V. RESULTS

VI. CONCLUSION

The introduction contains a description of the motivation for the project and a general overview. The background contains information about Machine Learning, Poisoning Attacks, and other relevant topics. The related works section discusses the research environment, relevant papers and this work's contribution. Methods describe the methods used to test basic ML, evaluate poisoning attacks, and the eventual approaches to implement attack vectors in the simulated environment. The results enclose the main findings of this research concerning poisoning attacks on ML developed for robotic control. The conclusion describes the significance of the findings and sets a way forward for more tests and potential research in physical implementation.

II. BACKGROUND

A. *Machine Learning*

ML has constantly increased its role within the world as technologies adapt to this emerging technology. It has been taken up by nearly every profession and industry to accomplish tasks ranging from sorting algorithms to identifying handwritten numbers. The term ML covers a broad array of algorithmic models developed to learn nonlinear patterns. Since its origin, several different types of ML have been developed and adopted. Support Vector Machines (SVM) were developed to classify a broad range of data using hyperplanes to distinguish between classes of data. Neural Networks (NN) are a more recent innovation that essentially model the human brain through the use of interconnected nodes. Layers, which are made of an organization of nodes, take in an input and activate some of the nodes which pass on information to the next layer until a final classification or decision is made. NNs are so popular that different types have evolved to accomplish a wide variety of tasks. For example, Deep Neural Networks (DNN) have additional layers of nodes, referred to as the hidden layer, that can process more complex datasets because there are more nodes that make up the network. Convolutional Neural Networks (CNN) use some of the layers to condense inputs in order to simplify the decision criteria and boost efficiency. The research used basic SVM, NN, CNN, and DNN to test different ML models efficiency and success.

ML has been used for classification algorithms and is commonly applied to tasks such as facial recognition, scanning, or object categorization. Industries have started adopting ML to help with controls such as cars, manufacturing robots and more. Because such industries are applying ML to physical controls, this research can be considered as a response to the inherent risks associated with this combination of technologies.

B. *Poisoning Attacks*

Using Machine Learning does not come without a cost. ML needs an immense amount of data to train on in order for it to become efficient and accurate. It may take hundreds of epochs or training cycles on a dataset with thousands of instances for the model to become effective. This training data and training cycle creates a large attack surface for potential hackers to inject an attack vector. One of the most widely studied attacks in the ML field is the poisoning attack.

A poisoning attack injects malicious data into the training data set to establish an attacker's desired bias in the ML or to cause unwanted behavior from the trainer. Delivery methods such as Man-in-the-Middle attacks, hosting corrupted files on websites, or affecting the collection of data can all be used to inject the vector, as the payload, into a training data set. Depending on the type of training used, attackers can also attack different sections of the dataset. For example, an attack's outcome can be determined by whether or not it supervised the attacking of the dataset's labels so that it mislabels information. A different approach attacks the data by changing aspects of the instance to create a bias within the ML model. Poisoning attacks can be used to diminish the availability of a model by causing it to fail in its ability to perform its intended function or to influence it into misidentifying a chosen instance in order to reduce its integrity.

III. RELATED WORKS

A. Research of Poisoning Attacks

There is currently an ongoing race for adversaries to seek knowledge on compromise machine learning systems and for defenders to protect against these machine learning attacks [4]. Substantial progress has been made on the defensive end and the National Institute of Standards and Technology has released an article articulating different types of attacks and possible mitigation strategies [5]. This work aims to delve further into the realm of possible attack strategies and how they could impact a machine learning system.

Machine Learning systems require large amounts of data and training to work effectively. The research uses the Cornell dataset to train for the UR5 robotic arm simulation [6]. It uses hundreds of images of everyday objects and labels how to properly grip it with coordinates. The Cornell Dataset is one of the most commonly used datasets for ML grasping problems.

Fig 2. Image of a UR5 Robotic Arm
Source: Adapted from [7]

Traditional poisoning attacks on Machine Learning data will ultimately aim to degrade the accuracy of the entire system which can make them easily identifiable [8]. Traditional attacks sometimes assume an attacker is altering up to 100% of the training data. This is useful if the ultimate goal is to degrade the system so much that it is not functional at all, but it is very obvious and detected easily.

As previously stated, different types of poisoning attacks already exist and are actively being researched and attackers and defenders.. A well-implemented poisoning attack will be undetected by the system and undetectable by a person. Clean data and poisoned data can look indistinguishable from one another to a human being but the poisoned data will ultimately cause the system to malfunction [2].

Poisoning attacks on classification algorithms have been proven to be very effective [8]. They've proven so effective that such attacks can cause up to a 12%-23% reduction in accuracy when only 3% of the training data is modified [8]. This significant margin of error can have serious economic and safety implications, especially in industrial environments. The longer the poisoning attack goes undetected, the more economic damage it will cause and the greater chance it becomes a safety risk.

IV. METHODS

A. Introductory Individual Neural Nets and Poisoning Attacks

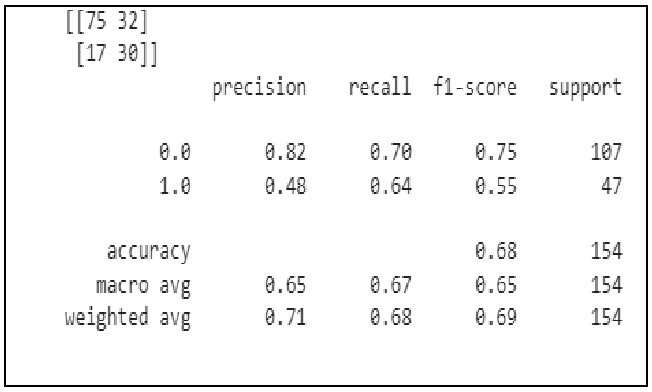
This project can be accurately assessed in three stages: introduction to machine learning and poisoning attacks on basic neural networks, simulating an attack on the UR5, and a physical attack on the UR3 arm. Before attempting to simulate and physically attack the UR3 and UR5 robotic arms respectively, this project developed an entry-level understanding of how neural networks are created and how basic poisoning attacks influence the performance of these neural networks. To accomplish this, this project used machine learning tutorials to create functioning neural networks on Google Colab. By using different neural networks and python libraries, this project gained the greatest breadth of experience in different forms of neural networks with the additional hopes of attempting different types of poisoning attacks in the simulation and physical attack phases of the project. The two types of neural networks investigated were Convolutional and Deep Neural Networks (CNNs and DNNs).

One example of a neural network tutorial this project implemented was a Deep Neural Network tutorial that used the Keras python library to build the neural network. The neural network trained on a dataset that predicted whether or not a patient was diabetic positive, using eight health indicators (i.e. blood pressure, BMI, number of pregnancies) associated with each patient. The dataset, called the Pima Indians Diabetes Dataset, is commonly used in machine learning because it allows the neural network to train on the eight indicators to predict the boolean value of a diabetes patient (1 for positive, 0 for negative).

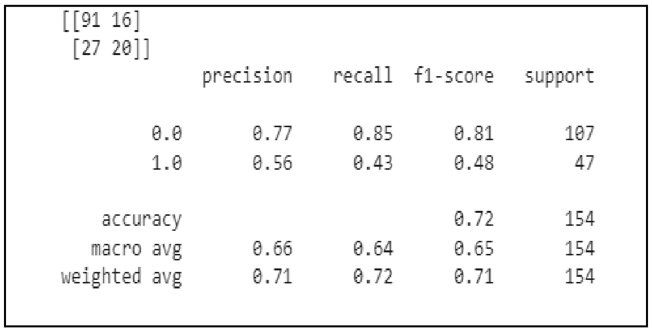
The neural network tutorial built out a neural network with three layers, designed to optimize the accuracy of the neural network on this particular dataset. The neural network was given two sets of data, a training set and a testing set. This model trained, or learned, using the training set by cycling through a set number of epochs to predict each patient's diabetic diagnosis in an output dataset. This output dataset was then compared against the actual figures in the testing set to gauge the performance of the neural network.

This project also implemented a confusion matrix in each neural network designed. A confusion matrix is a common performance metric in machine learning. A confusion matrix compares the predicted values that the neural network generated against the test values and generates a matrix of how many of the predicted values were true positives, true negatives, false positives, and which were false negatives. The neural network performed well in this particular tutorial by giving an outcome with a high number of true positives and true negatives and a low number of false positives and false negatives, which generally indicate a high performing model.

The poisoning attack on this neural network was relatively basic; it targeted specific health indicators that were shown to have higher impacts on the behavior of the neural network and tweaked them only slightly. In one particular experiment, increasing the blood pressure (one of the health indicators) by 50% for every third patient in the dataset increased the number of false negatives in the confusion matrix by over 58%. False negatives are the most damaging statistic when it comes to diagnosing patients, as a false negative indicates that a patient is negative when they are actually positive. This basic poisoning attack has the potential to heavily impact the performance of a neural network designed to predict whether or not a patient is diabetic based on their health assessments. Such an attack would certainly lead to medical errors and cause the health and lives of many patients to be put at risk.



```
[[75 32]
 [17 30]]
              precision    recall  f1-score   support

         0.0       0.82      0.70      0.75       107
         1.0       0.48      0.64      0.55        47

    accuracy                           0.68       154
   macro avg       0.65      0.67      0.65       154
weighted avg       0.71      0.68      0.69       154
```



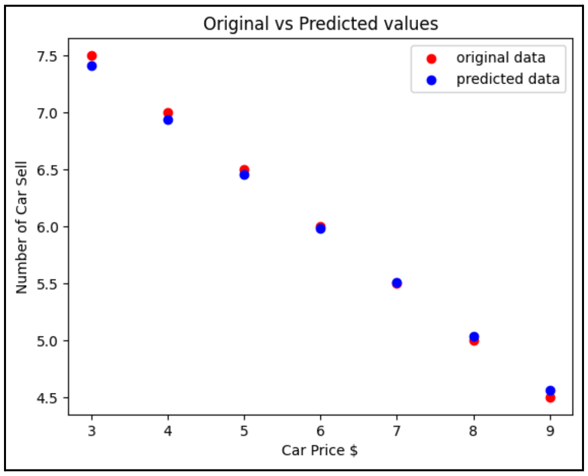
```
[[91 16]
 [27 20]]
              precision    recall  f1-score   support

         0.0       0.77      0.85      0.81       107
         1.0       0.56      0.43      0.48        47

    accuracy                           0.72       154
   macro avg       0.66      0.64      0.65       154
weighted avg       0.71      0.72      0.71       154
```

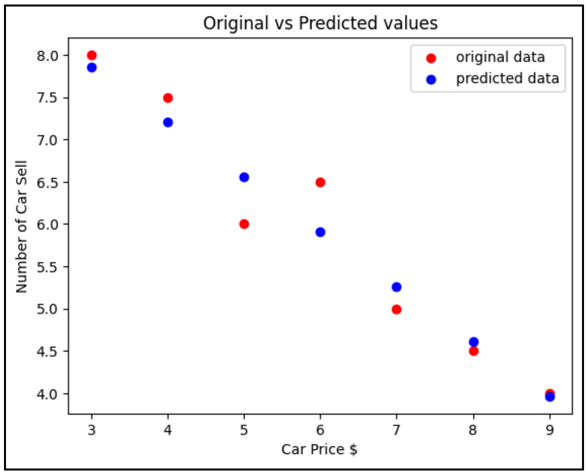
Figs 3 & 4. The confusion matrix and classification matrix from before and after the attack.

Figures 3 and 4 demonstrate the effectiveness of this poisoning attack. The statistic mentioned above is found by looking at the lower left figures in the confusion matrices, an increase of 58% between the two.

The project also used an SVC to demonstrate ML capabilities. The SVC used a linear kernel with sklearn framework and trained the data on MNIST data set commonly used for classification problems. The MNIST data set has 60,000 28x28 images of handwritten digits on the black and white scale. The SVM trained to identify what number the handwritten digits represented using supervised learning. The SVM had a 90% success rate with classifying the numbers when trained on a subset of the training data.

Linear regression is used to predict values based on the value of something else. This is especially useful for calculating likely car prices. The graphs below show an example of linear regression being used for car prices and the predicted data almost exactly matches the actual car data. The second linear regression graph shows the same information but with poisoned data. With only slight alterations to the input data, the accuracy of the predictions is significantly altered.





Figs 5 & 6. Unpoisoned data (top) vs poisoned data (bottom)

B. Simulation Testing

Following the experiments with individual poisoning attacks, this project simulated poisoning attacks on a simulation software of the UR5 Robotic Arm. The simulation software used is an open source Github repository and is based off of Kumra et. al's project on developing a convolutional neural network for robotic grasping [9][10]. The simulation can train on both the Cornell and Jacquard datasets, and it comes prepopulated with training models from both datasets, making it convenient to evaluate the success of subsequent poisoning attacks. The poisoning attacks presented in this paper all target the Cornell dataset [6]. This paper presents three methods of poisoning attacks attempted on the UR5 simulation software, with varying degrees of success in poisoning the simulation (see appendix).

The first method of attack targets the point cloud data of the Cornell dataset. In the Cornell dataset, each training image contains a set of corresponding text files that provide orientation and training details for the model. One of these text files is the point cloud data, which provides coordinates, an RGB value, and an index for every pixel in an image. In an effort to produce a visible effect on the simulation, the first attack distributes a universal 50% increase to every x, y, and z coordinate for every image in the Cornell dataset. Hypotheses for a dramatic effect in the performance of the simulation after a dramatic change to the dataset were incorrect, as this attempted attack demonstrated a negligible impact on the performance of the simulation. The highest performing epoch of the attacked dataset showed a 98% grasp success rate which was consistent with models trained on correct data.

This project's second attack on the simulation targeted a different aspect of the Cornell dataset: the "cpos" file associated with each image in the dataset. The cpos files contain coordinates of pixels in the images where the object exists, giving the training model a way to differentiate between the background of an image and the object it is required to pick up. The Cornell dataset also contains "cneg" files, which contain coordinates of pixels where the object does not exist, or negative values of the object. Making every effort to produce an effect on the simulation, this project again increases each coordinate in the cpos file by 50%. The results show immediate differences in the quality of training, the best performing epoch in the poisoned data performs at a 8% grasp success rate, comparatively dismal to the 98% success rate in a normally trained model. Given this trained network, the simulated arm is unable to pick up any object, and it can be reasonably concluded that targeting the cpos files has a considerable effect on the performance of a simulated UR5 arm trained on the Cornell dataset.

The final poisoning attack attempted to target the simulation more precisely with the aim of affecting the simulated arm's ability to pick up spherical objects. In a search through the Cornell dataset's 885 images, only 58 images are spherical in appearance, constituting about 6.5% of the dataset. The spherical objects' dataset index numbers were placed into a text file, which was read into the attack program. The program filtered the cpos files based on the spherical images' index numbers, and increased the coordinates of only those files by 20%, 40%, and 50%. The training with this subtly poisoned data at the 50% trial appeared successful, with the highest training epoch having a 97% grasp success rate. This grasp success rate is consistent with unpoisoned networks. However, when this network was run on the simulation, there was a noticeable effect on the performance of the arm. The arm picked up larger objects like cracker boxes and hammers with ease, but it was unable to pick up spherical-like objects like strawberries or tennis balls. The augmentation of the coordinates within the cpos files of the spherical images skewed the simulation's ability to locate the correct grasping position. The results section of this paper more clearly shows the effects that this attack had on the simulation. Having produced a subtle but significant effect on the simulated UR5 arm, this paper demonstrates that targeted poisoning attacks are possible on a simulated UR5 arm.

C. *Physical Attack with UR3 Implementation*

The final stage of this project was intended to be the implementation of Machine Learning on a functioning UR3 robotic arm (which operates very similarly to the UR5 arm) by using an OAK-D camera as a visual sensor. The arm was meant to use the simulation and the ML in order to act out the simulated UR5's tasks in real life.

The first step of this stage was to secure the hardware needed to run the arm. Luckily, both the UR3 and the OAK-D were provided by the USNA Weapons, Robotics, and Control Engineering Department. It is important to note that the UR3 is capable of moving using either provided instructions (code) or by the use of manual waypoints. Both methods are controlled by the PolyScope controller connected to the arm. The waypoints would be created by physically moving the arm into specific positions and recording them as such. The operator can then execute a "manual" program for the arm to go into each waypoint position in order. Because the arm focuses on preset instructions, the operators can preset grasping movements for the ML to act on depending on what the camera perceived.

A successful attack on the simulated robotic arm is crucial in performing an attack on the physical arm. Given a successful simulated attack on the robotic arm, the poisoned training data is then used to train the physical arm. The physical domain also introduces real world problems that the simulated world does not. For example, in a real scenario, not every object of a specific type will always look exactly the same. The objects trained in the simulated domain might have slight differences to their appearance in the real world. These differences may be attributable to variables such as lighting, distance to the camera, and more. Additionally, a physical arm might also have been degraded overtime which could impact the arms's movements and lead to reduced performance overall. All of these examples are just a few of the scenarios that could lead to a difference between simulated results and physical results that this project aims to identify.

V. RESULTS

A. *Point Cloud Data Attack*

The first poisoning attack on the simulation targeted the point cloud data of each image in the Cornell dataset by increasing each coordinate (x,y, and z) by 50%. This attack was unsuccessful, the highest performing epoch of the network trained on this data achieved a 98% grasp success rate, consistent with the control network which also had a 98% success rate. After working with the network in further attacks, it became clear that the Cornell point cloud data did not affect the simulation's ability to grasp objects and was not a major factor in the training data.

B. CPOS data Attack

The second poisoning attack on the simulation more accurately targeted the cpos data of each image in the Cornell dataset. The cpos text files contain rectangular coordinates of different grasping locations for each image in the dataset. In the attack, each coordinate in every cpos file of the dataset was increased by 50%. This attack was successful, achieving a dismal 8% grasp success rate, and in running the simulation, the arm was unable to identify or pick up a single object. Following this attack, it was evident that cpos data was the most important aspect of the Cornell dataset that had the largest impact on the performance of the network and the simulation. However, this attack was unhelpful in assessing the potential for sophisticated poisoning attacks, and a more subtle poisoning attack was necessary to determine the possibilities of affecting robotic control using poisoning attacks.

C. Targeted CPOS Attack

The final poisoning attack on the simulation targeted a small portion of the dataset, specifically altering the grasp point of spherical objects. This attack was intended to affect the simulation's ability to pick up spherical objects while not affecting the simulation's ability to pick up non-spherical objects. Compared to the first two attacks, this attack was subtle and based on a specific type of object. Figure 7 demonstrates the results gained from this attack, where the simulation was given trained networks with varying levels of data poisoning.

In this paper, level of data poisoning relates to the percentage by which the spherical cpos data coordinates were increased. The simulation was trained at each level of data poisoning and tested on each chosen object ten times. The simulation's score in Figure 7 was the number of times that it successfully picked and placed the object. The tests used five of the fifteen objects available in the environment. Two of the objects were mostly spherical objects (Tennis Ball and Strawberry), while the other three were non-spherical objects (Mustard Bottle, Banana, and Power Drill).

The first level of poisoning, which had 0% strength, produced results indicative of a successfully trained simulation. The network used for this simulation was one of the provided networks included in the simulation files. Using this network, the simulation picked up every item 100% of the time except for the mustard bottle, which wasn't picked up consistently at any level of the simulation testing.

UR5 SIMULATION RESULTS				
Item	**0%**	**20%**	**40%**	**50%**
Tennis Ball	10	4	1	0
Strawberry	10	6	0	0
Banana	10	8	5	3
Mustard Bottle	8	4	7	8
Power Drill	10	10	10	10

Fig.7 Testing various items on the simulation with varying poisoning levels

The simulation networks trained on varying levels of spherical object poisoning, at 20%, 40%, and 50% strength. One of the non-spherical objects, the power drill, performed equally at every level, indicating that the attack on spherical objects did not impact the simulation's ability to pick up this object. Similarly, the banana, which does have some round edges, was picked up at a higher rate than the fully spherical objects like the tennis ball and the strawberry. The simulation generally performed worse on spherical objects than the non-spherical objects, decreasing in performance as the level of poisoning increased. For instance, the simulation was unable to pick up the tennis ball at a 50% level of poisoning, but gradually increased to a 40% pickup rate as the poisoning decreased to 20%.

One outlier in the data that can't be accounted for is the simulation's performance in handling the mustard bottle. None of the networks given to the simulation were able to pick up the mustard at a 100% success rate, but what is most puzzling is the simulation's ability in the 20% poisoning level. The simulation only picked up the mustard bottle four out of the ten times it was tested, significantly lower than the other levels. This data remains troubling to the overall results of the simulation. A possible explanation for this exception is that a mustard bottle or similarly shaped object does not appear in the Cornell dataset, making it challenging for the simulation to know where to pick up this object.

As a whole, the simulation results from the spherical object attack suggest that more precise poisoning attacks can be made on certain object categories by targeting the cpos data in the Cornell dataset. Not only did the targeted object category, in this case spherical objects, perform worse than the non-spherical objects, but as the level of attack increased, these objects performed worse. Subsequent attacks on this simulation could target different object categories or draw in more sophisticated vision based attacks.

D. Physical Arm Results

There are many challenges when moving from a simulated network to the physical domain. The arm requires a calibrated camera that can identify different objects. These incompatibilities can lead to obscure bugs that can be difficult to identify. Once these challenges are ironed out, one can then move forward with the machine learning step and apply simple algorithms to ensure the system is working as intended.

During the attempt at implementing the ML with the UR3 robotic arm, there were major issues encountered. Errors from legacy python libraries provided difficult troubleshooting challenges. Networking between software and available hardware proved to be incompatible and led to a breakdown of the system. Because the arm had only used the MATLAB language, the operators had to install a variety of python libraries that had difficulty loading. Some of these may be attributable to the use of a heavily controlled workstation intended for strongly-directed student use - future work should be performed on a setup with less boundaries. The OAK-D, an RGB-D camera capable of "seeing" depth, was unable to be used because the arm itself was unable to perform as intended for this research.

The difference between the simulation and the physical UR3 robotic arm's results would have highlighted the practical effects of applying the ML to a real-life control system. Although inconclusive, this research indicates that there is still much to be learned in this area of study.

VI. CONCLUSION

This project set out to contribute to this undeveloped section of cybersecurity and control systems research by demonstrating different poisoning attacks on a simulated UR5 robotic arm. It intended to do this by drawing attention to the vulnerabilities that Machine Learning presents to control systems when the technologies are applied together. By working with several different MLs, applying them to a simulated UR5 arm, and ultimately executing a series of poisoning attacks to great effect and proving that controllers are at risk to the threat of data poisoning.

By analyzing the results, the project's outcome suggests that the potential effectiveness of applying existing data poisoning attacks designed for vision classification to a robotic arm can be highly successful at achieving desired results after the attack execution. The control data behaved as it should have with no irregularities or disturbances to the simulation's desired performance. The most extreme results demonstrated the capabilities of data poisoning at wiping out a controller's functionality entirely when data is severely augmented. The results also demonstrated the ability to target specific performance metrics by applying more nuance to the scale and execution of the attack.

This research was not inclusive of all the issues facing cyber-physical systems with ML implementation. The attacks can be made more complex to reduce the likelihood of human detection to make the operator think that any mistakes by the system are made by chance instead of as a result of capabilities degraded by data poisoning.

Going forward from the visual classification process applied in the simulation, future research should relate the ability of the arm to physically function while handicapped with poisoned data. Future works that could expand from this project include developing a poisoning attack that targets physical infrastructure like robotic arms directly instead of relying on existing vision classification attacks. These works and others can differ in a variety of ways including by Machine Learning algorithm, data set, as well as the software dictating motion in the arm.

Once more methods of poisoning physical controllers are established, it will be the responsibility of the developers of technologies such as cyber-physical controllers to build in proper security measures as tools of prevention. With the methods known, entities owning such systems and controllers can utilize penetration-testing services to test the security of their own technologies. They can also take responsive action to recover from discovered vulnerabilities/exploits by referencing proper penetration testing reports or, in the worst case, post-incident reports. If such systems are used to support critical infrastructure in any capacity, it might be in order for there to be security oversight and resources from a government entity such as the Cybersecurity and Infrastructure Security Agency (CISA) as an investment in both security and safety.

This project has the potential to make a broad impact in automated manufacturing as well as other sectors where automated machines such as the UR10e arm are becoming increasingly popular. Although it can't currently be said whether such an attack would be more effective than the one envisioned in this project since this attack is yet to occur on a real system (at the time of writing), a more specific poisoning attack could yield more damaging results. It should also be noted that some attacks may be more or less successful based on the hardware and software that is being exploited.

As Machine Learning continues to proliferate throughout all sectors of industry, this project may be able to serve as a warning to maintain security-minded software and hardware whenever possible. Industrial technologies assume inherent risk when they learn and develop through the use of algorithms - especially when Machine Learning is the key component. Hopefully the gap in knowledge regarding the relationship and impact of poisoning attacks and physical systems can be closed to ensure safe cyber-physical systems in the future.

APPENDIX

Github Repository

All of the code for this project was compiled into a Github repository. This repository contains the attack programs, the text files of images targeted in the spherical object attack from the Cornell dataset, and the trained networks that were inputted into the simulation to test for results. The repository is linked here: https://github.com/gabeslind1/The-Application-of-Poison-Attacks-on-Robotic-Controls

ACKNOWLEDGEMENTS

Our team would like to thank CDR Brien Croteau, Ph. D, for his constant efforts throughout the year as our advisor for this project. Additionally, we would like to thank Profs. Mike Kutzer from the USNA Weapons, Robotics, and Control Engineering Department, and finally Gavin Taylor from the USNA Computer Science Department. This project was made possible by the USNA Cyber Science Department, and we would like to thank the entire department for these past three years in the Cyber Operations major.

REFERENCES

[1] E. Atkinson. “Man crushed to death by robot in South Korea” BBC. https://www.bbc.com/news/world-asia-67354709 (accessed December 3, 2023)

[2] Oladimeji D, Rasheed A, Varol C, Baza M, Alshahrani H, Baz A. “CANAttack: Assessing Vulnerabilities within Controller Area Network” Sensors (Basel). 2023 Oct 2;23(19):8223. doi: 10.3390/s23198223. PMID: 37837053; PMCID: PMC10575265. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10575265/

[3] Geiping, Jonas, Liam Fowl, W. Ronny Huang, Wojciech Czaja, Gavin Taylor, Michael Moeller, and Tom Goldstein. "Witches' brew: Industrial scale data poisoning via gradient matching." arXiv preprint arXiv:2009.02276 (2020).

[4] M. Mozaffari-Kermani, S. Sur-Kolay, A. Raghunathan, and N. K. Jha, “Systematic Poisoning Attacks on and Defenses for Machine Learning in Healthcare,” *IEEE Journal of Biomedical and Health Informatics*, vol. 19, no. 6, pp. 1893–1905, Nov. 2015, doi: https://doi.org/10.1109/jbhi.2014.2344095.

[5] A. Vassilev, “Adversarial Machine Learning”:, 2024, doi: https://doi.org/10.6028/nist.ai.100-2e2023.

[6] Y.Jiang, S.Moseson, A.Saxena, “Efficient Grasping from RGBD Images: Learning using a new Rectangle Representation,”*in IEEE International Conference on Robotics and Automation*, Shanghai, China, May 9-13, 2021, 3304-3311

[7] CLEARPATH Robotics. *UR5.* Accessed: Nov. 29. 2023. Available: https://store.clearpathrobotics.com/products/universal-robots-ur5

[8] J. Steinhardt, P. W. Koh, and P. Liang, “Certified Defenses for Data Poisoning Attacks,” arXiv:1706.03691 (2017)

[9] Vrielink, J (2021) Learning to Grasp Objects in Highly Cluttered Environments using Deep Convolutional Neural Networks [Source code]. https://github.com/JeroenOudeVrielink/ur5-robotic-grasping

[10] S. Kumra, S.Joshi, F.Sahin, “Antipodal Robotic Grasping using Generative Residual Convolutional Neural Network,” arXiv:1909.04810 (2019)