Detecting network based attacks in Industrial Control Systems

By Oliver Gunnell

Contents

Abstract

The use of Machine Learning (ML) models in Intrusion Detection Systems (IDS) tailored for Industrial Control Systems (ICS) has increased the level at which malicious activity can be detected. However, newer methods of by-passing IDSs are becoming more prominent. These methods are defined as adversarial machine learning or adversarial attacks which share the common goal of reducing a ML models accuracy. Adversarial attacks have been proven to be effective within the image classification domain. This paper aims to discover the effectiveness of adversarial attacks generated using Fast Gradient Sign Method (FGSM) applied to data obtained from an ICS testbed. This paper will further explore the effectiveness of utilising the adversarial examples generated by FGSM originally intended to lower the performance of a model, but instead are harnessed to improve the model's robustness to adversarial attacks. Random Forest, J48 and Jrip are the three classifiers this paper focuses on. Overall, Random Forest and Jrip adapt better to adversarial training than J48 which is supported by their significant increase in classification accuracy when applied to unseen adversarial examples.

I. Introduction

Industrial control systems (ICS) are a branch of operational technology (OT) networks that govern the monitoring of industrial operations. Features of an ICS include sensors, actuators and commands from an operator. ICS networks are found in critical national infrastructure, for example, water treatment facilities, electrical power system plants as well as other discrete manufacturing plants. The number of targeted cyber-attacks on ICS networks is ever rising [23]. The perceived security of ICS previously focused solely on physical extremities, for example, fences and secure barriers to stop intruders. While this may deter the physical attacker a newer threat which targets the network vulnerabilities of ICS renders this type of security redundant. This branch of cyber-attacks are called adversarial machine learning or adversarial attacks. Adversarial attacks are becoming more common in the field of ICS and this is one of the major motivations behind this project. Intrusion detection systems (IDS) and adversarial attacks have been well researched in the context of Information technology (IT) networks however the latter is only recently becoming a well researched topic. There are some similarities behind the main functions of an IDS for ICS and IT networks however there are some fundamental differences that need to be addressed. The notion that OT networks are completely segmented from IT networks and thus the internet, is a concept that is fading. There was an understanding that this "air gap" placed between the two networks automatically ensured the security of the OT network from the outside world. However, in parallel to the information age, the concept of interconnections between networks and IoT devices is growing within CNI and ICS. This new level of connectivity to the internet has introduced new security threats to ICS, thus highlighting the importance of improving the network-based security for these types of systems.

This project will build on the foundations of established machine learning algorithms to build a robust model that could become the basis of an IDS tailored to ICS networks. Once this model has been finalised adversarial techniques will be used to create adversarial examples that alter both malicious and benign traffic in an attempt to force the model to misclassify datapoints. The model will then be re-tested, and an evaluation of the models performance will be observed and analysed. This project

will further observe the performance of different classifier models by introducing adversarial training with the foresight of improving the classifiers robustness to such adversarial attacks.

A. Aims of the Project

The first aim of this project is to evaluate the performance of state of the art classification models in the context of ICS network data. This evaluation will be used to extract a number of models that perform well and can form the basis of further experiments working towards finding a model suitable for an IDS tailored to ICS networks. To achieve this first aim a dataset comprised of ICS sample data which will contain both benign and malicious datapoints will be used to evaluate and analyse the classifier models. The use of adversarial methods will be applied to the dataset to evaluate the robustness of each classifier. Another aim of this project is to obverse the effectiveness of adversarial training methods used to improve a classifiers robustness to unseen adversarial attacks.

This project's intended audience is anyone who is interested in the security of ICS networks in regards to machine learning algorithms and there use in IDS. This project should also be of interest to those who have experience or wish to gain experience in the use of adversarial techniques, methods and their effect on the performance of classification models. This is especially true for those who work in network positions within CNI or other ICS network-based plants. Another possible area of interest is for those wishing to investigate the potential generalisation of producing adversarial examples generated specifically for one model and the possibility of those examples also affecting other types of classification models.

The scope of this project includes research into suitable machine learning classifiers, IDS methodologies and adversarial techniques used. Firstly to reduce the performance of a machine learning classifier and secondly to be utilised to improve a model's robustness to such attack techniques.

The major outcome of this project is to highlight the robustness of classifier models used in the field of ICS and IDS. Adversarial techniques are becoming common attack methods to force miss-classification and disrupt the function of an IDS. Therefore, choosing a classifier model that is robust to data perturbation is becoming as important as choosing a classifier model with good performance on a particular dataset. Another outcome of this project is to highlight the ability to use adversarial methods that are commonly available to fool well researched classifier algorithms. This in turn will show the need to use adversarial training on these models to improve their performance on adversarial attacks and thus, improving the effectiveness of IDS.

The effectiveness and suitability of adversarial training in an ICS context will be evaluated with the intention of improving a model's robustness to adversarial attacks. This project aims to provide evidence that supports the need for adversarial training whist evaluating the effectiveness of adversarial training on different types of classifier models.

II. Related Work and Background

Intrusion detection systems are at the fore-front of detecting and alerting security engineers of a potential network breach. A fundamental aspect of IDS is the machine learning algorithm that needs to establish whether the traffic it is analysing is normal everyday network traffic or malicious activity that could be a cyber-attack. To be able to make these predictions the machine learning model must first understand the different types of traffic and then identify the features in the network that change when a cyber-attack is occurring or has occurred. To achieve this, datasets that contain a mixture of normal (benign) network traffic and attack (malicious) network traffic are needed to train a model and then an unseen dataset is needed to evaluate its performance and transferability to practical implementations. Training a model too closely to training data will produce a very good model for that training set but will produce poor predictions for any unseen dataset. This is the notion of overfitting. 10-fold cross validation can be used to reduce the likelihood of overfitting. Conventionally the dataset is split into training, validation and then testing sets. There are 3 main performance metrics used to evaluate the performance of a classifier model namely, Precision, Recall and F-measure or score. These matrices and their use in this project's context are discussed in more detail in later sections of this paper.

The problem with such models is that the field of adversarial machine learning has emerged which highlights the opportunity to by-pass such IDS by carefully modifying data points in a fashion that causes miss-classification. If adversarial techniques are successfully used in an ICS network this could cause major damage to equipment and have a large impact on a multitude of people. ICS networks are controlling large scale equipment with extreme precision where any error made by the IDS could cause massive concern.

A. Adversarial attack types

Adversarial attacks can be categorised depending on how they are designed to impact the machine learning algorithm and what elements of the model they exploit.

Huang et al (2011) has created a taxonomy for classifying the different types of adversarial attacks based on what stages of a machine learning model the attack targets and the underlying motive behind the attack. The resulting taxonomy is divided into the following segments.

- **Influence:** Influence is further subcategorised into two sections; **Causative attacks** rely on manipulating the training process by possessing influence over the training data. Using the above metric causative attacks would be classed as a poison attack. **Exploratory attacks** have no direct influence over the training data and thus, the training process is not altered during this attack. This attack focuses on extracting information about the detector/classifier or the data used to train the model.
- **Security Violation:** These attacks target the integrity and availability of a model by producing adversarial examples that cause a model to generate False Negatives or by severely reducing classification performance. This is achieved by producing a large number of classification errors which render the model unusable. If an attack focuses on the integrity of the model this is an example of an evasion attack. Another subsection of security violation attacks focuses on extracting information from the learner to gain private information about the users of a system. These are further subcategorised under security violations as privacy attacks.

Huang et al (2011) further discusses the level of knowledge an adversarial may have about the IDS and the underlying model used for classification and detection. They describe the different elements of the classification system the attacker may or may not have obtained knowledge about. They outline that much like an attack on an encrypted system where the encryption method used will be known by the attacker and could even be open source, the classification algorithm used by the IDS needs to be assumed that this knowledge is also known by an attacker. It is also highlighted that knowledge about the features and training data could also be known by an attacker. Whilst it is easier to reduce the likelihood of an attacker knowing the exact format of the training data used to train a model, as with many ICS applications they follow similar methodologies and frameworks furthermore, some will use the same equipment. One example of this is the majority of water treatment plants which use similar protocols, sensors and actuators controlled and monitored in similar ways resulting in comparable frameworks. Even if the system designer has kept one particular feature space hidden, an attacker is likely to have gained knowledge from similar systems with less discrete feature choices that could indicate the type of training data used which could be transferable.

Yuan et al (2017) introduced a treat model for adversarial attacks that is devised of 4 subsections

- **Adversarial falsification**
    - This describes the method of using adversarial examples to produce False Positives or False Negatives (evasion)
- **Adversarial's knowledge:** This has been described in the above section of this paper however, this paper introduces the notion of white-box and black-box attacks.
    - Whitebox attacks rely on the adversaries having extensive overall knowledge of the machine learning model. This covers all areas of the learning process including knowledge of the training set and the learner algorithm.
    - Black-box attacks assume the adversarial has no knowledge of the model or the ability to gain information about the training data. They only know the output of the confidence score of the model but not the exact model itself.
- **Adversarial specificity**
    - Targeted attacks focus on misclassifying data points to a specific target class. This type of attack is usually designed for multiclass applications.
    - Non-targeted attacks do not specify which class a datapoint should be misclassified as providing the adversarial class is not the original class of that datapoint.

In the context of binary class datasets targeted and non-targeted are regarded as equal and there is only one target class available for the adversarial class that isn't the original.

- **Adversarial frequency**
    - **one-time** attacks offer the notion of producing an adversarial example from one run through of the method
    - **iterative** attacks require iterative methods for optimising an adversarial example

Adversarial attacks on ICS networks can also be categorised depending on how they are designed to impact the IDS monitoring the ICS network. Many types of attack designed for IDS exist. Attacks that are designed to go unnoticed by the IDS or to overload a piece of equipment on the ICS causing this equipment to be unusable are just two examples of attacks that can be leveraged to cause damage to ICS. Corona et al (2013) created a taxonomy of the different types of adversarial attacks designed for ICS networks. This taxonomy outlined 6 types of attacks which have been summarised below.

- **Evasion Attacks:** The intrusion pattern intended to disturb the system aims to go undetected and therefore creating no alerts to operators of its presence.
- **Overstimulation:** This attack creates a large number of false alerts with the aim to confuse the system and the operators monitoring the system.
- **Poisoning:** Well-designed patterns are introduced to the IDS learning algorithm with the intension to lower detection accuracy and classification performance.
- **Denial of Service:** Overloading a sensor in an IDS causing the sensor to shut down decreasing the IDS detection performance.
- **Response Hijacking:** Forces the IDS to produce incorrect alert descriptions which mislead the IDS response protocols.
- **Reverse Engineering:** This involves the attacker gaining knowledge about the IDS and more specifically the features used within the IDS and even what detection algorithm is being used. This attack improves the likelihood of the above methods as they can be tailored to specific IDS.

B. Adversarial Methods & Defence Techniques

Adversarial machine learning in its simplest form is the practice of adding noise to datapoints in a sufficient way to cause a classifier model to misclassify a datapoint. The art of calculating the optimal amount of noise needed to achieve this goal whilst minimising this noise level is a well researched area.

There are many adversarial methods available that are widely researched, the leading research in this area focuses on the image classification domain. Goodfellow et al (2015) outlined and implemented the Fast Gradient Sign Method (FGSM) on different image classification tasks. Using the ImageNet dataset, they show that FGSM can create adversarial examples for individual pixels in an image that force a convolutional neural network (GoogeLeNet) to misclassify images. They go further and show that this method can be used to make the classifier produce a 99.3% confidence level that an image is of an incorrect class. This shows the effectiveness of this adversarial generation method.

The formula for FGSM method is found below.

$$\eta = \varepsilon \operatorname{sign}(\nabla_x J(\theta, x, y))$$

where:

- η = Adversarial example
- θ = Model parameters
- x = The input to the model
- y = Input labels associated with x
- J = Cost
- ε = Multiplier to ensure perturbation is within controlled bounds

This method uses gradients of the neural network to create an adversarial example. More specifically it adds perturbations based on the gradient of the cost function with respect to the input data. Different values of ε controls the size of the perturbation. This value must be large enough to create an effective adversarial example but not too large that the example is obviously detected as being malicious.

There has been related work in the area of adversarial attacks in an ICS context, a particular example of this is Zizzo et al (2019a). They construct a simple adversarial attack on a long short-term memory (LSTM) classifier used on the SWAT dataset. For this attack to be successful the attacker needed a vast knowledge of the IDS and required information regarding the features used for classification. This attack may not be transferable to many real-world applications, but it re-enforces the opportunity to by-pass machine learning based IDS within the ICS field. Zizzo et al (2019b) outlines a method to generate adversarial attacks on a time series IDS. This study uses a more sophisticated method of adversarial generation to fool the LSTM.

C. Adversarial Training and Defence mechanisms

The goal of defending against adversarial attacks is to improve a ML models robustness to adversarial attacks. Two methods of achieving this will be discussed in this section, the first uses adversarial training and the second uses the concept of defence distillation.

Adversarial training has been introduced within the computer vision field of machine learning. Goodfellow et al (2015) observed the performance change of a neural network when applied to unseen adversarial examples when re-trained on a training set that contained a sample of adversarial examples and original training data. The conclusion of that experiment shows that adversarial training can improve the robustness of a model against adversarial examples. Szegedy et al (2014b) experiment using the MNIST dataset and a non-convolutional neural network. The model again, is trained using a sample of adversarial examples mixed into the original training data. The resulting test error rate is extremely low.

Szegedy et al (2014b), discuss the transferability between the generation of adversarial examples for one type of model and the effectiveness of these examples when applied to other types of ML model [10]. They also discuss the generalisation of training sets, showing that a large portion of adversarial examples generated specifically from one training set will be misclassified by a model trained on a mutually exclusive training set. Again, these papers focus on the image classification domain, nevertheless, showing the potential of adversarial training and suggesting more universal characteristics of adversarial machine learning.

Another approach to adversarial training outlined by Papernot el at (2015) uses defence distillation. This research shows the effectiveness of this type of adversarial training on the MNIST dataset and CIFAR10. While this method seems very effective at improving the robustness of the model, there is little research in using defence distillation on larger datasets. For this reason, this method of adversarial training will not be employed in this paper as the classifier used for an IDS needs to show effectiveness on a range of dataset sizes.

As there is further evidence to support the effectiveness of adversarial machine learning this paper will focus on this method of improving a model's robustness to adversarial attacks.

III. Dataset

This section of the paper will outline in detail the power system testbed used for all experiments constructed for this project. The data set used for this project was made publicly available by Mississippi State University and Oak Ridge National Laboratory - 4/15/2014. They have devised a small overview of a power system framework. See Fig. 1 for a more complete structure of the power system used to produce the data sets.

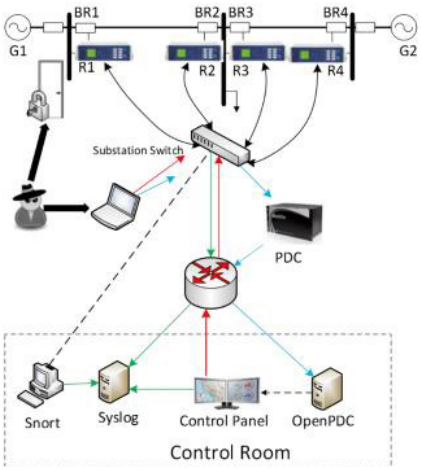

Fig. 1: Power System Testbed [19]

Looking at Fig.1, G1 and G2 are power generators whilst R1-R4 are Intelligent Eectronic Devices (IED's) which control the operation of the breakers (BR1-BR4). The IEDs use a distance protection scheme to identify faults and trip the required breaker. Each IED controls one breaker. Distance protection schemes use voltage, current phase angles and other metrics to ensure that if a fault occurs only an isolated section of the power system is shut off. There is no internal method to validate whether a fault is faked or legitimate. Operators can also manually issue commands to the IEDS to trip the breakers. This usually occurs during maintenance on the line.

There are 4 PMU's in this framework of which 29 are synchrophasor measurements are taken for each giving a total of 116 measurement columns, 12 columns for control panel logs, Snort alerts and relay logs totalling 128 features. A more in-depth outline of the features included in this dataset are displayed in Table. 1. The index of each column is in the form of "R#-Signal Reference" that indicates a type of measurement from a PMU specified by "R#". The signal references and corresponding descriptions are listed below. For example, R1-PA1:VH means Phase A voltage phase angle measured by PMU R1

Feature	Description
PA1:VH – PA3:VH	Phase A - C Voltage Phase Angle
PM1: V – PM3: V	Phase A - C Voltage Phase Magnitude
PA4:IH – PA6:IH	Phase A - C Current Phase Angle
PM4: I – PM6: I	Phase A - C Current Phase Magnitude
PA7:VH – PA9:VH	Pos. – Neg. – Zero Voltage Phase Angle
PM7: V – PM9: V	Pos. – Neg. – Zero Voltage Phase Magnitude
PA10:VH - PA12:VH	Pos. – Neg. – Zero Current Phase Angle
PM10: V - PM12: V	Pos. – Neg. – Zero Current Phase Magnitude
F	Frequency for relays
DF	Frequency Delta (dF/dt) for relays
PA:Z	Appearance Impedance for relays
PA:ZH	Appearance Impedance Angle for relays
S	Status Flag for relays

Table. 1: List of Features with a brief description used in the power system testbed.

There is a total of 15 data sets produced from this testbed that contain both benign and malicious data points. These data points were originally categorised into three classes, “Natural events”, “No events” and “Attack events”. Natural and No events have been grouped together and classed as benign activity leaving the Attack events as malicious data points and thus creating a binary classification set.

There are four types of scenario manufactured on this power system framework.

1. **Short-circuit fault**- Short in the power line that can occur at multiple locations along the line.
2. **Line maintenance**- One or more relays are disabled on a specific line to allow maintenance on that line
3. **Remote tripping command injection (Attack)**- This is an attack that sends commands to a relay which causes a breaker to open.
4. **Relay setting change (Attack)-** Relays are configured with a distance protection scheme and the attacker modifies the relays settings to disable the relays function. This causes it not to trip on a valid fault or valid command sent by an operator.
5. **Data injection (Attack)**- Valid fault is imitated by changing values to parameters such as current, voltage sequence components etc. this attack aims to blind the operator and causes a black out.

The finalised output of this testbed data in regards to this project is a binary class set that uses all 15 datasets that includes malicious or benign datapoints.

IV. Training and Evaluating the Models

This section of the paper will outline the process used to train and evaluate each classification model chosen to be evaluated in this project. This section will also include the evaluation matrices used to record and analyse each classifiers performance

To evaluate the effectiveness of machine learning algorithms in the context of ICS, the dataset discussed in the previous section was used to analyse commonly used classifiers in IDS. The metric used to evaluate the accuracy of a classifier is discussed below. Weka outputs a number of important results when evaluating any classifier which help to give an understanding of that classifiers performance.

-**True Positive**: Where the classifier has correctly predicted a data point as being malicious and it is confirmed to be of that class.

- **True Negative**: Classifier has correctly predicted that a data point is benign, and it is confirmed to be of that class

- **False Positive**: Classifier has predicted a data point is malicious, but it is confirmed to be of the benign class

- **False Negative**: Classifier has predicted that a data point is benign, but it is confirmed to be of the malicious class.

Once the above metrics are calculated a further 3 metrics that give an overall good understanding of the performance of a classifier can be calculated namely, Precision (P), Recall (R) and F-score (F). The equations to achieve these metrics are found below.

$$P = TP/\ TP + FP$$

$$R = TP/TP + FN$$

$$F = 2 \cdot P \cdot R/\ P + R$$

All classifiers used in this paper were from those available within the Weka platform and thius, used this implementation. Each classifier was initially evaluated using 10 -fold cross validation to reduce overfitting. A variety of classifiers were chosen based on their performance in ICS scenarios discussed in previous studies of IDS, namely Random Forest, J48, Naive Bayes ZeroR and finally Jrip. The classifiers evaluated are described in more detail below.

-Random Forest: Uses an ensemble learning method, which creates a number of decision trees where each tree outputs a class prediction. These predictions are aggregated and the class with the highest number of individual predictions is chosen as the overall output prediction.

-J48: An implementation of C4.5 which itself is an extension of the ID3 algorithm. Trees are built using information entropy, each node is split using information gain. This classifier uses single pass pruning to reduce the risk of overfitting.

- Naïve Bayes: Generative learning model, this model is based on Bayes theorem **cite**. Where the model assumes that the presence of a particular feature in a class is irrelevant to the presence of any other feature.

-ZeroR: Simply predicts the majority class for every datapoint. For this reason, this classifier serves as a good baseline. If a classifier does not out-perform this classifier the classifier is not suitable for the application.

-Jrip: Rule based classifier that uses Repeated Incremental Pruning to Produce Error Reduction (RIPPER) [12].

Previous studies evaluating power system data have used Jrip + Ada boost on a reduced sample size. This produced promising results, however, this classifier is not feasible for large datasets as the time taken to train and evaluate the classifier is extremely large. Therefore, this model has not been evaluated in this project.

A. Data Pre-processing

Data pre-processing is an important step in machine learning. Carefully analysing the data points being used to evaluate a classifiers performance is paramount to the reliability of the results obtained.

Python using scikit and pandas were used to handle the pre-processing needs for this project [22]. Infinite values were removed, and the data set was checked for missing values or outliers before any classification tests were executed. Due to the datasets being in csv format when loaded into Weka the feature used to indicate the class of a datapoint was originally of type numeric. This features data type was then changed to nominal as this is supported by the majority of classifiers on the Weka platform. Weka also provides an ARFF viewer application, this can be used to analyse the features in more detail and simplifies the conversion from csv files to arff. These can then be Imported into Weka explorer keeping all the predefined feature types and finally be used to train and then test models.

Fig.2: Distribution of Benign and Malicious datapoints across both the training and test sets

B. Evaluating the Models

The 15 dataset files discussed in section III were combined into one csv. A python program using scikit learn and pandas were used to split the dataset into approximately 60% training and 40% testing sets. Once split the sets were converted to .arff which is a recognised machine learning file format for Weka. Each classifier model was first trained using the training set and 10-fold cross validation. Cross validation was used to reduce overfitting. This process involves partitioning the dataset into a number of subsets and holding one set back as the testing set. For K- fold cross validation the dataset is split into K subsets with 1 set held for testing the others are used to train the model, this process is repeated K times holding out a different subset each time. For each fold the evaluation score is recorded and thus giving one overall score for the K folds. Each classifier was then evaluated on the unseen test set where the results were used as the baseline of that model's performance. Results for the baseline accuracy is displayed in table Fig. 3.

Fig.3: Accuracy Score as a percentage for; ZeroR, J48, Random Forest, Naïve Bayes and Jrip applied to the unseen test set.

Once this baseline was achieved for all classifiers the next step was to investigate the features themselves. Understanding which features rank highest and therefore have the greatest influence over the classification of a datapoint is very important. Modifying sample values in features that have little or no overall impact on that datapoints overall classification is redundant. Weka includes a filter called InfoGainAttributeEval, this filter evaluates the entropy value for each feature which displays each features contribution to the overall class decision. Using Weka's infogain filter, the 128 features were ranked in order of influence. This list was then used to identify which features require the most attention from an adversarial perspective. The top 3 ranking features are shown in table. 2.

Feature	Rank
R4-PM2:V	1
R1-PA2:VH	2
R1-PA3:VH	3

Table.2: Top 3 ranked features extracting using Weka's Infogain Filter

The trained models using the method described in this section of the paper will be used for all future experiments in the context of this report. The only exception of this is in the adversarial training section of this paper.

V. Adversarial Example Generation

This section of the paper will outline and discuss two methods of generating adversarial examples. The first approach was a simplistic approach that was designed for this project with the idea of producing a baseline for potential levels of misclassification within the test set used without being as strict on the amount of perturbation needed to be applied to achieve such misclassifications. The second method described in this paper involves the Fast Gradient Sign Method (FGSM) introduced by Goodfellow [7]. This method will aim to use much stricter perturbation levels to achieve high levels of misclassification. FGSM has been applied in two concepts, the first focusing on increasing the number of false negatives and the second increasing the number of false positives. Both concepts will be used to further analyse the robustness of ML classifiers.

A. Outline of Manual Approach for Adversarial Attacks

The first approach adopted was to manually introduce noise to the top 10 ranked features values to force the classification models to miss-classify malicious datapoints as benign. Using the InfoGainAttributeEval filter within the Weka environment meant extracting the features that influence the class decision the most was a trivial task. Initial steps in implemented this method involved extracting the malicious datapoints from the test set to begin modifications. Once a malicious set was constructed the next step was introducing a method to modify these values to explore the difference in classifier performance. Using python pandas and data frame objects made handling and manipulating csv files relatively trivial, using the feature labels it was simple to modify sample data points independently. Focusing on one feature at a time resulted in the ability to calculate a min-max range for each filter, then a percentage of that range, for example, 1% and finally increasing each sample value by the chosen percentage level. For each percentage value a new csv file was produced which included the adversarial examples generated for all malicious datapoints in the test set. These adversarial examples where then reintroduced to the benign datapoints creating a new test set for each percentage value used in this experiment. These newly created test sets were then used to re-evaluate the models in Weka. To explore whether decreasing certain values had a more detrimental effect to the overall classification performance of the chosen models the experiments were repeated but this time applying the method in a negative direction for each feature.

B. Outline of FGSM Approach for Adversarial Attacks

The second approach of producing adversarial examples for this project was using FGSM. Szegedy et al (2014b) shows that generating adversarial examples for one type of classifier model can be transferable and have an effect on other models, it is for this reason that I chose to use Sklearn's Support Vector Machine as the model to train and use as the basis of generating adversarial examples using FGSM. The SVM kernel configuration used for these experiments was the Gaussian orientation as this is a common kernel to use for non-linear classification problems. SVM is also a classifier used

in previous studies [13] and it has shown to be suitable for ICS data and thus can be used as the base for an IDS. This again supports the decision to use this classifier as the base for generating adversarial examples.

The approach of generating adversarial examples using FGSM was to first train a model on 4% of the entire dataset. This percentage was chosen as this was approximately 10% of the testing set used in all previous experiments. This reduced sample set was used due to performance conditions. The time taken to execute FGSM attacks on a large dataset was excessive for this project. This could be due to hardware limitations on the computer used to execute the experiments. To generate an adversarial example for each malicious data point in the test set, all the malicious data points were extracted from the test set into a single csv file. Once this file had been produced these malicious datapoints were parsed to the FGSM along with the parameter Epsilon. The implementation of FGSM used in this paper was supplied by Adversarial Robustness Toolbox using their Fast Gradient Method implementation [15]. Different values of FGSM's parameter epsilon were used to control the level of perturbation applied to each malicious datapoint. Larger values of epsilon introduce larger data perturbation levels. Table. 3 shows an example of two features that have had FGSM applied with different values of epsilon. This project has focused on re-evaluating the adversarial examples on Random Forest, Jrip and J48. This is due to the nature of ZeroR which predicts the majority class for every datapoint, thus rendering adding noise to features and generating adversarial examples for this classifier to be redundant. Naive Bayes performed significantly worse than the other 4 classifiers used in these experiments and therefore displayed its lack of suitability to this form of classification. It will therefore not be included in the further evaluations discussed in this paper.

Dataset	**R1-PA1:VH**	**R1-PM5:I**
Original Test set	85.16444667	297.7369
Epsilon = 0.1	85.26444244	297.6369
Epsilon = 1.0	86.16444397	296.7368

Table. 3: Adversarial examples generated by FGSM with 0.1 and 1.0 epsilon

Another type of adversarial attack which is described by Corona et al (2013) is overstimulation. This attack focuses on creating false alerts that aim to overwhelm the operator. This attack essentially causes the classifier model to produce a large number of false positives. False positives are when a benign data point gets classified as being malicious. An IDS that is constantly producing false alerts that bombards the operators can have many negligible effects of the monitoring of a network. For example, it can lead to an operator missing true positive datapoints due to the number of false positives clouding causing a cyber-attack to go unnoticed and disrupt the operations of the system.

To analyse the potential impact of such an attack, the same FGSM implementation as discussed in the first part of this section was used to generate adversarial examples on the benign data points in the test set. Again, using the same range of epsilon values, the performance and False positive rate of each classifier was analysed. This attack is focused on raising the False positive rate for the benign class.

Due to the imbalance of classes within the dataset the F-score will not be the only metric used to evaluate the effectiveness of this attack. The confusion matrix the classifier produces will be used

alongside the F-score to give a better understanding on the effect this type of attack is having on the classifier models.

VI. Results

This section of the paper will outline and display the findings produced by implementing the methods discussed in section V.

A. Results for Manual Adversarial Example Generation

This section will discuss the results found by applying the methods outlined in section V.A. The first method explained in this section involves increasing the selected feature sample values by varying percentage values ranging from 1 – 10 %.

Random Forest and Jrip showed improvement in F-scores due to the adversarial examples produced for every percentage value over the F-score produced on the original unseen test set. This could indicate that the noise introducing in a positive direction has aided these classifiers in distinguishing between benign and malicious datapoints. J48 showed a slight decline in F-score most convincingly for 1 and 2 % perturbation which then levelled off for all other perturbation values. Fig. 4 displays all results found by employing this method of adversarial generation for each classifier.

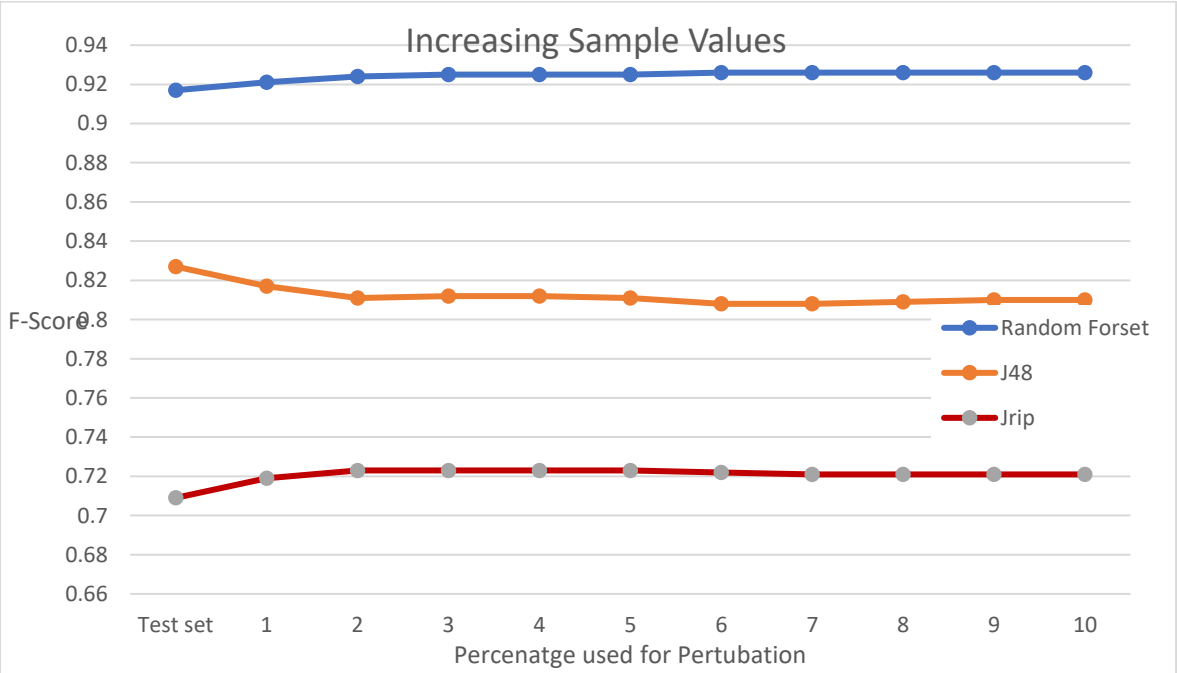

Fig. 4: F-scores for each classifier against the original unseen test set and ranges 1 – 10% perturbation applied to the top 10 ranked features in a positive direction.

The second manual approach taken was to apply the same methodology of perturbation level but to implement this in a negative direction to produce adversarial examples. This approach had significantly more impact on each model's performance over the method applying perturbation in a positive direction. See Fig.5 for all F-scores plotted against the percentage ranges. See Table. 4 for the confusion matrices of each classifier when applied to the unseen test set.

		Predicted	
		0	1
Actual	0	5376	3026
	1	1843	18585

J48

		Predicted	
		0	1
Actual	0	6439	1963
	1	339	20089

Random Forest

		Predicted	
		0	1
Actual	0	2065	6337
	1	692	19736

Jrip

Table. 4: Confusion matrices for J48, Random Forest and Jrip when applied to the unseen test set *(Benign = 0, Malicious = 1)*

J48's F-score ranges from 0.827 which was achieved against the unseen test set with no adversarial examples introduced, to 0.684 which was achieved on adversarial examples generated with 10% perturbation. This is a drop of 0.143. The confusion matrix produced by J48 when applied to the adversarial examples generated with a 10% perturbation reduction is shown in Table. 5. This shows that the percentage of malicious data points that were misclassified as benign (false negatives) in the original test set increases from approximately 9% to 31%. This demonstrates the effect of the adversarial examples generated using this method on the model's classification F-score. This classifier conforms to the hypothesis that the larger the perturbation the lower the classification accuracy. However, looking at Fig. 5 this demonstrates that the biggest difference in F-score between two points in the graph was between the test set and applying the classifier to the 1% adversarial example set. Table. 5 shows the confusion matrix for 1% perturbation. This shows that 4092 malicious data points have been misclassified as benign. When compared to the origin test set this is a rise of 2249 with only a 1% perturbation. The F-score has also dropped from 0.827 to 0.757. This shows a significant drop in F-score alongside a large increase in false negatives for a very small perturbation size.

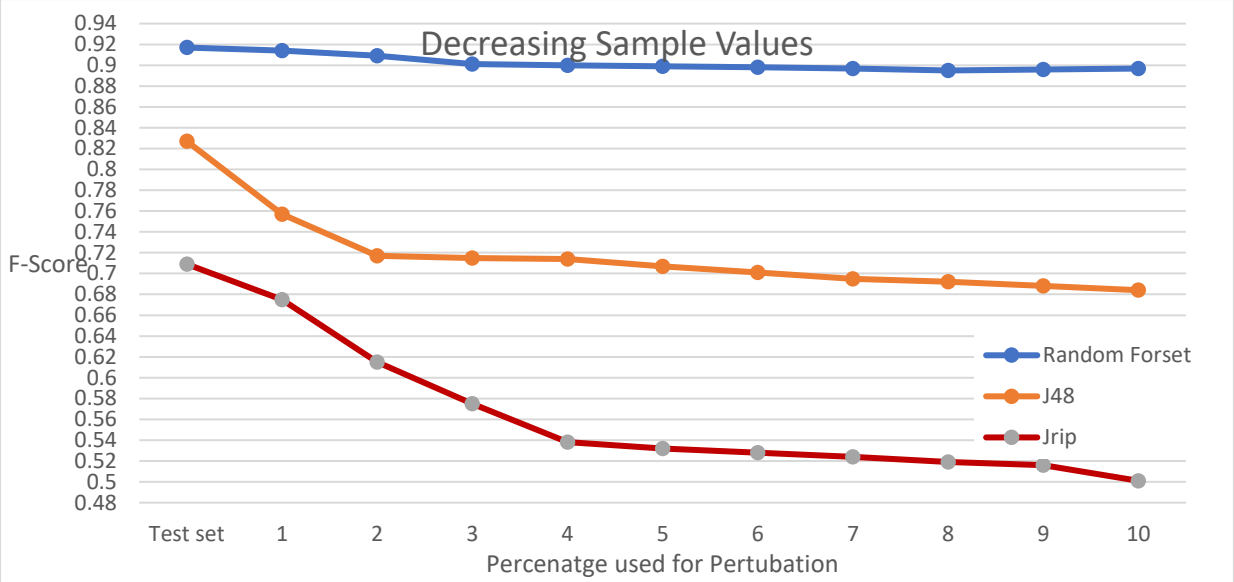

Fig. 5: Perturbation applied in negative direction for range 1 – 10%

Random Forest showed less change is F-score with the same perturbation levels applied when compared to both J48 and Jrip. As seen in Fig. 5 at 8% perturbation level the classifier's F-score is at its lowest of 0.895 this is still relatively high. This is only a 0.022 total drop compared to J48's total drop of 0.143 is not a significant amount. This could indicate that the Random Forest classifier is more robust to this type of adversarial example generation method. Further analysis can be drawn that for 9 and 10% the classifier performs better. This could suggest that when bigger values are applied to the samples it helps Random Forest distinguish between classes and thus, reducing the classification error.

Jrip performed worst out of the three classifiers on the unseen test set, having 0.709 as its F-score. Alongside J48 the larger the perturbation size applied to the sample values the lower the F-score. The total drop in F-score for Jrip was 0.208. This is the biggest change in classification accuracy out of the 3 models. This lowest F-score was produced on the test set with 10% perturbation. The biggest drop from 1 percentage difference is from the 1% and 2% percent example generations. At 1% the F-score is at 0.675 which drops to 0.615 at 2%. Adversarial examples generated from a 2% perturbation causes the F-score of Jrip to drop from a total of 0.094. Which again, is a significant amount that supports the effect this adversarial generation method is achieving.

Observing the behaviours of all three classifiers when applied to adversarial examples produced from decreasing sample values it is noticeable that Jrip had the biggest change drop in F-score, whilst J48 produced lower F-scores directly correlated to increasing perturbation size it is still justifiable to say that Jrip is more susceptible l to this type of adversarial example generation. Random Forest showed the smallest performance drop and overall produced better classification than the other two models suggesting that this model is more robust to this type of adversarial attack.

		Predicted	
		0	1
Actual	0	5376	3026
	1	4092	16336

Adversarial examples generated with 1% decrease applied (Benign = 0, Malicious = 1)

		Predicted	
		0	1
Actual	0	5376	3026
	1	6467	13961

Adversarial examples generated with 10% decrease applied (Benign = 0, Malicious = 1)

Table. 5: J48 confusion matrix produced when applied to adversarial examples

Dataset	**R1-PA2:VH**	**R4-PM:V**
Original test set	97.44866	132495.2188
1% perturbation	93.84877	131975.041882
10%	61.44972	127293.4

Table. 6: Adversarial examples generated by decreasing varying percentage levels of the selected features range.

B. Results for Adversarial Examples Generated Using FGSM

This section of the paper will display and analyse the results from applying the methodology defined in section V.B. This method involves using FGSM with varying levels of its epsilon parameter to introduce different levels of perturbation to the data set.

The first implementation of FGSM involves generating adversarial examples for all malicious datapoints within the test set. Fig. 6 displays the F-score for Jrip, Random Forest and J48 against epsilon values ranging from 0.1 to 1. The starting value of epsilon used by Goodfellow (2015) was 0.1 and this paper has followed that guideline [7].

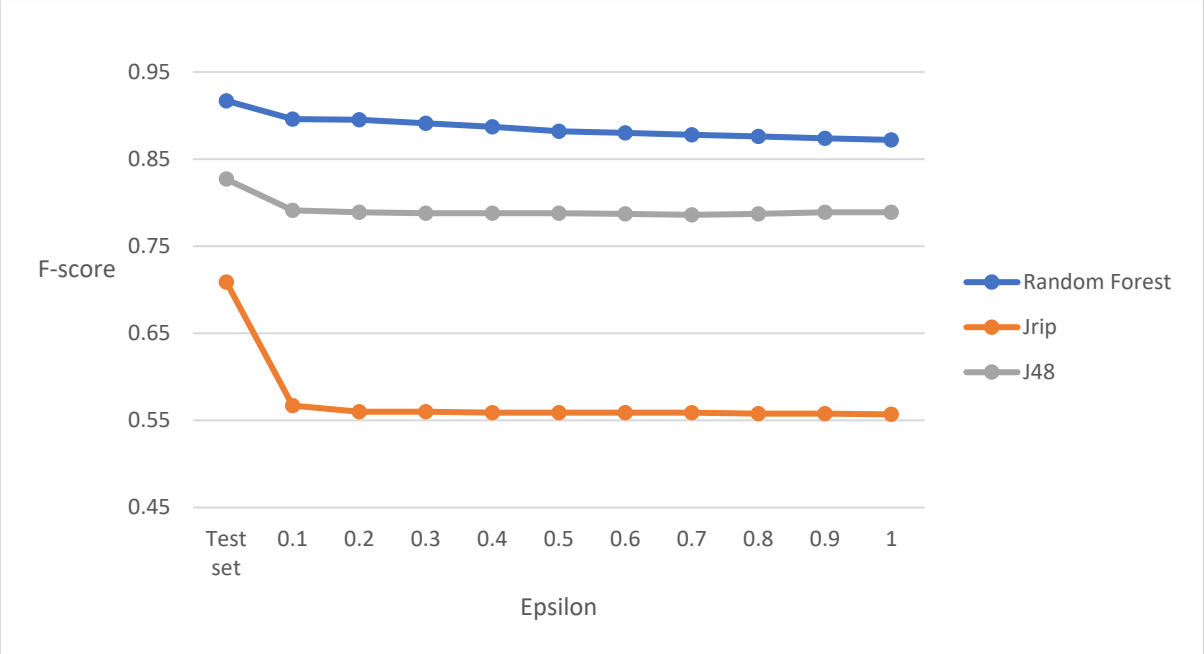

Fig. 6: Random Forest, Jrip and J48's F-score for each adversarial example produced using epsilon values (0.1 - 1).

The results shown in Fig. 6 suggest that Jrip has the lowest F-score out of the 3 classifiers across all adversarial examples generated, the F-score for Jrip on the unseen test set is also the lowest out of the 3 being only 0.709. With FGSM generating adversarial examples with an epsilon value of 0.1 Jrip's F-score drops to 0.567. This is the biggest performance drop of all epsilon values across all 3 classifiers. The confusion matrix for Jrip produced on the unseen test set and for epsilon value 0.1 is shown in Table. 7**.** The number of False Negatives for the unseen test set is 692 which is 3%, compared with 6097 which is approximately 30% for the epsilon value of 0.1 This is a significant increase in False Negatives. Jrip applied to adversarial examples generated with epsilon values larger than 0.1 did show an increase in the number of False Negatives. However, as the increase was minimal this paper has focused on highlighting FGSM with epsilon 0.1. The reason for this is that finding the smallest perturbation needed for maximal performance decrease is the aim for all adversarial attacks.

		Predicted	
		0	1
Actual	0	2056	6337
	1	692	19736

Unseen Test set Epsilon = 0

		Predicted	
		0	1
Actual	0	2056	6337
	1	6097	14331

Epsilon = 0.1

Table. 7: Confusion matrices produced by Jrip for (clean) test set and adversarial generated examples (Benign = 0, Malicious = 1)

J48 and Random Forest performed better than Jrip on the unseen test set also performing better than Jrip over all adversarial examples generated. J48's F-score dropped by 0.041 when epsilon equalled 0.7 which was its lowest score, see Table 8. for the confusion matrix for J48 applied to adversarial examples generated using epsilon 0.7. More interestingly, the J48 classifier showed very slight classification improvement after epsilon = 0.7 and continued to rise slowly for the remainder of the epsilon values used in this experiment. One possible reason for this is that the larger perturbations added to the datapoints helped distinguish the malicious points from the benign and therefore slightly improving the model's performance. J48 followed a similar pattern to Jrip regarding the biggest change in performance F-scores relative to epsilon values. Again, epsilon equal to 0.1 had the greatest impact on J48s F-score. Adversarial examples generated from this epsilon value would be the optimum level of perturbation used to achieve the highest relative performance impact.

		Predicted	
		0	1
Actual	0	5376	3026
	1	3142	17286

Table. 8: *J48 applied to Malicious datapoints using epsilon = 0.7 (Benign = 0, Malicious = 1)*

		Predicted	
		0	1
Actual	0	6439	1963
	1	1708	18720

Table. 9: *Random Forest applied to Malicious datapoints using epsilon = 1 (Benign = 0, Malicious = 1)*

Random forest performed the best on the unseen test set and also was the classifier that showed the smallest decrease in performance over all adversarial examples. This is shown by the total drop of 0.045 in Random Forest's F-score from unseen test set to epsilon equal to 1. See Table. 9 for confusion matrix where epsilon = 1.

These results show that adversarial examples generated using FGSM has affected the performance accuracy of all three models. For all models expect J48 as the epsilon value increased the classifiers F-score decreased. The original model used for adversarial generation was a SVM. The results show that examples produced for SVM are generalisable and therefore have affected the performance of Random Forest, Jrip and J48 models. This also is at harmony with Szegedy et al (2014b) theory that adversarial examples are generalisable between models. Further demonstrating that without knowing the exact model being used for an IDS, FGSM can be used to create adversarial examples that could

reduce the accuracy of a number of classifiers that are used in the field of ICS. With small values of epsilon classification accuracies are shown to drop. Fig. 6 shows the extent of miss-classifications if larger epsilon values and thus larger perturbations are applied to the dataset. While these adversarial examples are more likely to be spotted due to the large level of noise added to the datapoints the effect on the model's performance should these examples be undiscovered is considerable. For both J48 and Jrip the difference in F-scores for epsilon values greater than 0.1 is fractional. This could indicate that these models are more sensitive to smaller data perturbations but when handling larger perturbations, the accuracy of the model does not decrease in parallel as the model can distinguish between datapoints. On the other hand, Random Forest behaved as predicted, as epsilon increased the model's accuracy decreased. The overall accuracy of Random Forest was better than the two other models furthermore, Random Forest had the lowest drop in F-score over all adversarial examples used in this experiment. Indicating that this model is the most robust to this adversarial generation method outlined in section V.B.

The remainder of this section will analyse the results of an overstimulation attack type. This is the second method of utilising FGSM to create adversarial examples discussed in this paper. This attack focuses on raising the false positive rate produced by a classifier. This is achieved by increasing the number of benign datapoints classified as being malicious.

The adversarial examples generated using this method had drastic effects on J48's false positive rate. Table. 4 and 10 show the confusion matrices for J48 on the unseen test set and adversarial examples generates with epsilon 0.1 respectively. J48 applied to the unseen test set has a total of 3026 False Positives. This is approximately 36% misclassification for the benign class. Adversarial examples generated using epsilon value of 0.1 forced J48 to produce 6630 False Positives, a misclassification percentage of 79% for the benign class, this is a 43 percentage point increase in the number of false positives. The number of False Positives increases very slightly for the rest of the epsilon values used in this experiment. This indicates that smaller epsilon values and thus, smaller perturbations to data points may have greater impact on the model's classification performance. This should be further evaluated as the main goal of any adversarial attack is to be undetected by the classifier model, this is more likely achieved using the smallest perturbation levels possible that still effect the classifiers performance.

		Predicted	
		0	1
Actual	0	1772	6630
	1	1843	18585

J48

		Predicted	
		0	1
Actual	0	5864	2538
	1	339	20089

Random Forest

		Predicted	
		0	1
Actual	0	1554	6858
	1	692	19736

Jrip

Table. 10: *Confusion matrix for J48, Random Forest and Jrip. Epsilon = 0.1 applied to Benign data points (Benign = 0, Malicious = 1)*

Jrip has less significant performance change when applied to the range of adversarial examples introduced by this attack method. One possible reason for this is that Jrip has a much higher number of False Positives when applied to the original unseen test set meaning there is a much smaller number of True Negatives that the adversarial examples can leverage and force the model to misclassify. The number of False Positives Jrip displays when applied to the unseen test set is 6337, a misclassification percentage of approximately 75% for the benign class. Comparing this starting False Positive percentage to that produced by J48 on the same test set could explain the difference in performed drops between the model when applied to adversarial examples. See Table. 10 for the confusion matrix produced by Jrip when applied to FGSM with epsilon equal to 0.1.

Similar to the first adversarial attack mentioned in this subsection, Random Forest behaved as expected. As the perturbations applied increased alongside epsilon, the classification accuracy decreased. Unlike Jrip, Random Forest when applied to the test set only misclassified a small fraction approximately 23% of the benign datapoints as being malicious, a much better initial False Positive rate than Jrip. This percentage rose to approximately 58% when applied to adversarial examples with epsilon equal to 1. See Table. 11 for the confusion matrix for epsilon equal to 1.

		Predicted	
		0	1
Actual	0	4876	3526
	1	339	20089

Table. 11: *Confusion matrix for Random Forest with FGSM applied to benign datapoints with epsilon = 1.*

(Benign = 0, Malicious = 1)

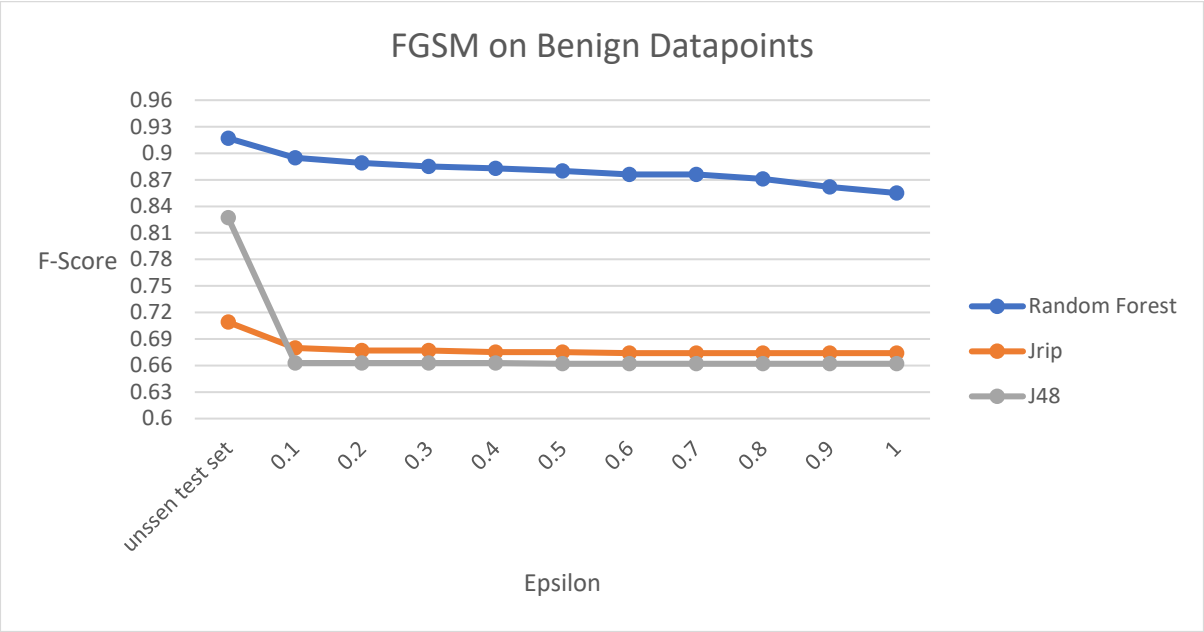

Fig. 7: Random Forest, Jrip and J48 applied to the test set and all ranges (0.1 – 1) of epsilon values generated for all benign datapoints.

The observation of the behaviour for Random Forest, Jrip and J48 when introduced to this overstimulation attack can conclude that all 3 models produced increased levels of False Positives alongside decreased F-scores indicating the effectiveness of this attack and thus the methodology outline in section V.B. This attack has produced almost parallel F-score trends when compared to the first attack implemented using FGSM. The results produced from this attack also highlight that in all 3 cases, the most efficient epsilon value to use for adversarial generation is 0.1. This again opens the door to further research in smaller epsilon permutations to identify the optimum noise level needed to create the most relative impact of the classifier algorithm. Moreover, this attack has again shown the generalisation between models and has also highlighted that FGSM can be used in an ICS context for adversarial machine learning.

VII. Adversarial Training

This section of the project will discuss and apply adversarial training to the models evaluated in section VI leading to further analysing the model's robustness to the adversarial methods discussed in VI.B

Adversarial training focuses on improving a classifiers performance at recognising perturbations added to datapoints intended to force a classifier to misclassify datapoints and therefore, reduce the classifiers accuracy. Szegedy et al (2014b) suggest that by adding a portion of the adversarial examples to the training set will improve a model's robustness to adversarial attacks. The first attack method outlined in section V.B will be used to evaluate the effectiveness of adversarial training on the robustness of a classifiers accuracy when identifying malicious datapoints.

Both Jrip and Random Forest models achieved their lowest F-score with epsilon value of 1.0. For this reason, 25% of the adversarial examples generated with this epsilon value have been added to the original training set. Both models were retrained using this new test set and then re-evaluated on all unseen adversarial examples. J48 performance was at its lowest with epsilon value 0.7 again, 25% of the adversarial examples generated with epsilon 0.7 were added to original test set to create a new test set and the model re-trained and re-evaluated on all adversarial examples.

Random Forest and Jrip both performed significantly better with adversarial training over the entire array of adversarial examples compared to the same classifier model without adversarial training. Random Forest with adversarial training had its lowest F-score at 0.901 this compared to 0.882 from the model without adversarial training is a noticeable difference, an increase of 2 percentage points. What's more interesting is that for epsilon values above 0.5 the classifier with adversarial training produced 0 False Negatives when classifying malicious datapoints. This means that all malicious datapoints were predicted to be malicious with no errors. Random Forest with no adversarial training performed slightly worse as the epsilon value increased however, as already discussed for the classifier with adversarial training improved with the increase of epsilon for 80% of the epsilon values used. Fig. 8 shows Random Forest F-scores across all adversarial examples for the model adversarial training and without.

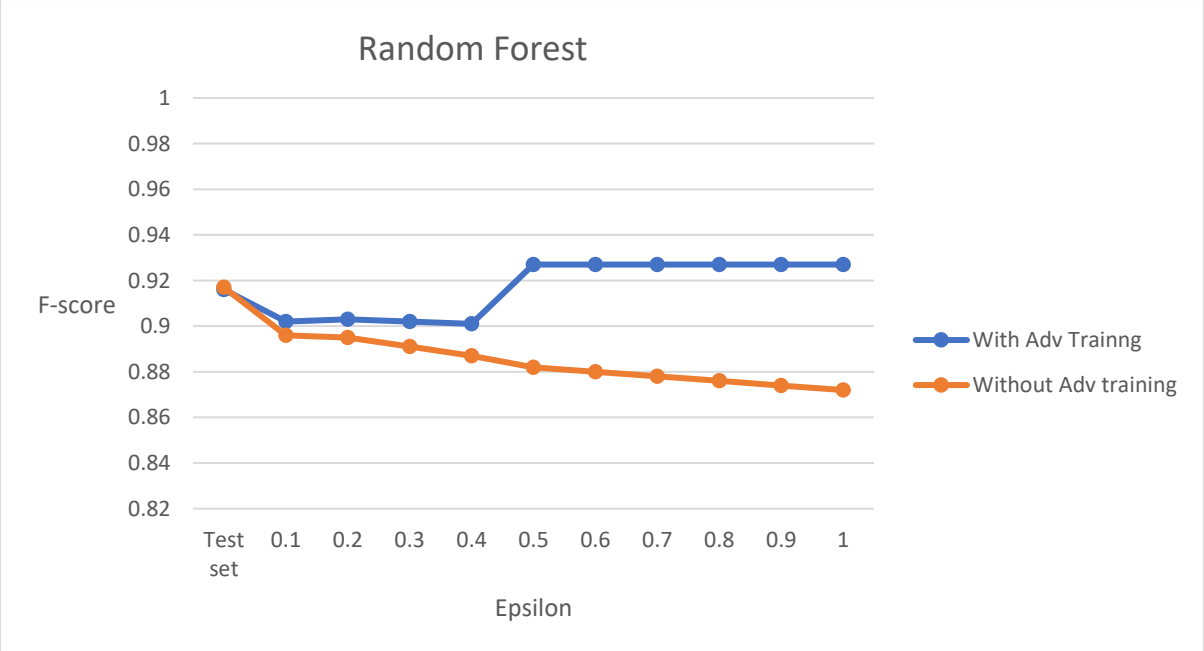

Fig. 8: Random Forest F-score for each epsilon value (0.1 – 1) with and without adversarial training

Jrip in every case saw the accuracy of predicting malicious datapoints improve with adversarial training over all unseen adversarial examples. For epsilon values 0.3 - 1 Jrip had an F-score of 0.718. In comparison the highest F-score recorded for Jrip without adversarial training was 0.567, furthermore, this highest F-score was achieved with the smallest epsilon value used, 0.1. With adversarial training Jrip produced an F-score of 0.71 or greater across the whole range of adversarial examples. Jrip with adversarial training produced 0 False negatives when classifying malicious datapoints for epsilon values 0.8 to 1, this could be due to the effectiveness of this adversarial training method or slightly due to overfitting as the adversarial examples added to the original test set were generated using epsilon equal to 1. Fig. 9 shows the difference is accuracy performance between adversarial training applied to Jrip and non-adversarial training models. Although overfitting was mentioned as a possible reason for certain epsilon values achieving better F-scores the overwhelming increase in accuracy performance across all adversarial examples reinforces the impact adversarial training has had on the classifier.

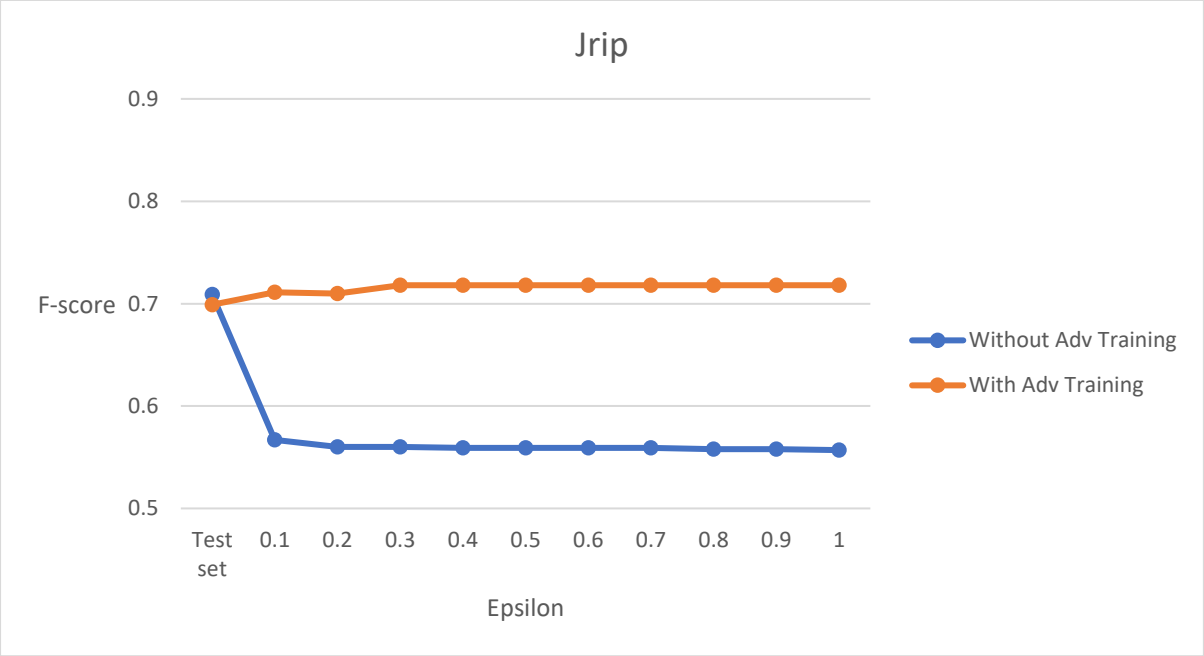

Fig. 9: Jrip classifier F-score for score for each epsilon value (0.1 – 1) with and without adversarial training.

J48 without adversarial training had very consistent F-scores across the entire epsilon range. J48 showed less improvement with adversarial training compared with both Jrip and Random Forest but still slight improvement across all combinations of adversarial examples. Fig. 10 displays the F-score for every adversarial combination generated using different values of FGSM's epsilon value against the model with and without adversarial training.

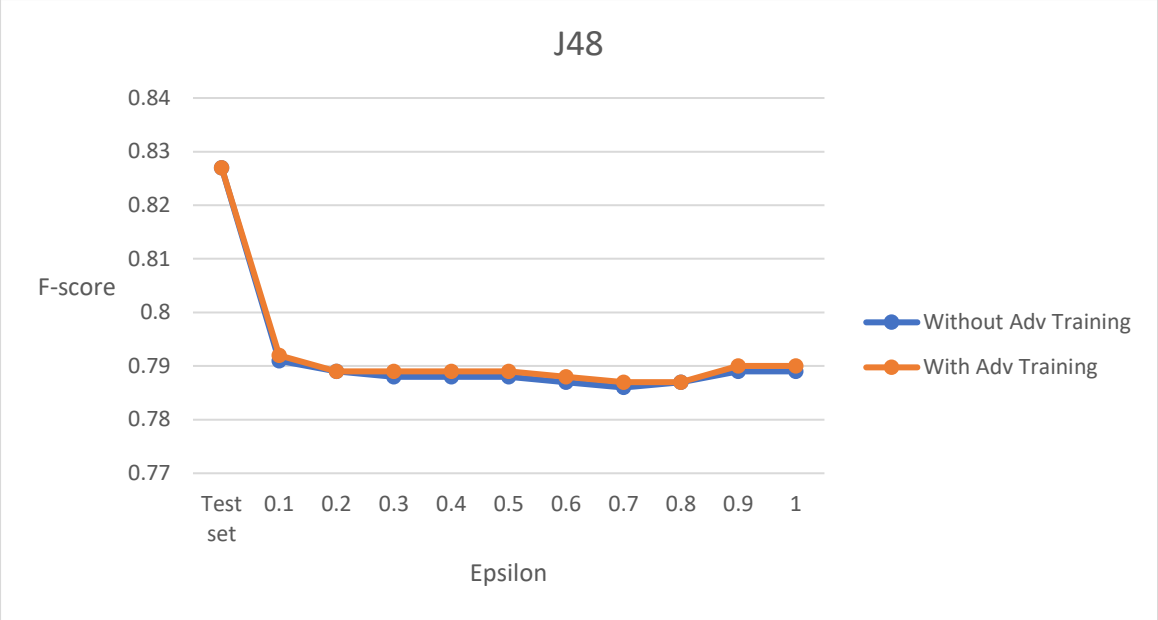

Fig. 10: Jrip classifier with and without adversarial training applied to all epsilon values (0.1 – 1)

See appendix 1 and 2 for the comparison of confusion matrices produced for epsilon values 0.1 and 1 between models with and without adversarial training.

Jrip displays the most improvement as a result of adversarial training. Followed by Random Forest. J48 showed the least performance increase due to adversarial training.

VIII. Conclusion

Machine learning classifiers are a fundamental element of IDS. The advancements of the internet have introduced interconnectivity between the once secluded ICS and the external world. Whilst this new level of connectivity has greatly improved some aspects of monitoring and performing operations on an ICS network it has also introduced a new attack vector to ICS. Adversarial attacks are known to hinder the performance of such classifiers, furthermore, disrupting the ICS operations which can cause massive problems, as CNI use ICS if an adversarial successfully bypasses an IDS monitoring the ICS using adversarial machine learning the consequences could be astronomical. Not only can cyber-attacks go unnoticed causing copious amounts of problems. The magnitude of alerts a security engineer is faced with on a day to day basis whilst monitoring an ICS is cumbersome without the added danger of an adversarial attack causing extra false positive alerts.

This paper has discussed the effect of generating adversarial examples is a number of ways with the intention of decreasing the accuracy performance of a number of classifiers that show promising performance measures in the context of classifying data found in ICS. Two methods of generating adversarial examples were outlined in this paper. One method consisted of manually adding various levels of noise to the dataset and observing their impact of the classification models and the second method utilised FGSM to generate optimised noise levels within varying bounds, controlled by the parameter epsilon. The manual method of data perturbation did produce promising results. However,

the level of noise introduced to achieve significant performance reduction for each classifier was much larger than those examples generated using FGSM.

This paper has evaluated the effectiveness of FGSM when generating adversarial examples on a power system testbed varying the level of perturbation applied to the features. The analysis reported in this paper shows that J48, Jrip and Random Forest all display a decrease in F-score performance when applied to adversarial examples generated using FGSM. Random Forest performed the best when applied to the adversarial examples generating using FGSM. This in turn indicates the higher robustness level of this model furthermore indicating its potential use for an IDS for ICS.

The results from this paper also suggest that using adversarial training and introducing 25% of adversarial examples generated using FGSM to the original training set and then re-training the models improves the robustness against unseen adversarial examples significantly for both Random Forest and Jrip.

This model has also reinforced the theory put forward by Szegedy et al (2014b) suggesting the generalisation of adversarial examples generated by one model effecting the performance of other types of model. This has been shown by using a SVM to generate adversarial examples using FGSM and observing the behaviour of Random Forest, Jrip and J48.

IX. Future Work

An aspect of this project that requires further attention is the scope of classifiers used in the experiments. Expanding this list to include more models would help fortify a good model to use for and IDS tailored to ICS. This project used Weka as the implementation of each classifier model. The robustness of SVM was not evaluated in this project but has been suggested to be effective in IDS. Further research into the feasibility of using such a classifier in this context is needed.

Future work could also include the implementation of other adversarial methods like Carlini and Wagner (2016) and Newton-fool (2017). The scope of this project focuses on FGSM which is a popular adversarial method within the image classification domain however, more research into a more suitable method for ICS applications may be needed to extensively explore the ability of adversarial machine learning within this context. To test the effectiveness of FGSM this project focuses on epsilon values below 1. Future work could provide an insight into using larger epsilon values and their effect on the robustness of classifiers against the size of perturbations.

Most IDS using machine learning classifiers will be assigned the task of not only identifying a data point as malicious but also what type of attack is occurring within in the system. Or this type of classification a multiclass dataset Is required. As this project focuses on binary class datasets future work could involve working with multiclass datasets and evaluating the impact adversarial examples produced using methods like FGSM have on multiclass accuracy. The overall goal of the adversarial example will still be to fool the classifier to misclassify a malicious datapoint as being benign, but the extra level of sensitiveness needed to train a model to classify data points correctly in a multiclass set is higher than that of a binary set. Discovering whether this more sophisticated training paradigm will influence a model's susceptibility to adversarial examples could be included in future work now this project can be used as a baseline.

Another area of future work would focus on finding an optimal level of adversarial examples reintroduced to a training set that both improves a model's robustness but does not lend itself to overfitting.

This project did not look into the effects of defence distillation used as an adversarial training method in an ICS context, this method has showed promising results for smaller datasets [12] and therefore requires further research into its capabilities in larger scenarios. Future work could identify the applicability of such defence methods in larger contexts providing critical research into the use of adversarial training for IDS.

X. Reflection

On reflection deciding on a machine learning project with no background knowledge in this area was a steeper learning curve than initially thought. Whilst knowledge of ICS and protocols was beneficial to this project as it helped to identify the application of this work and the motivation of the research area it did not serve as any technical help. However, my new understanding of machine learning as a broad concept has increased exponentially. Concepts like, decision trees, adversarial attacks and F-measures were all alien to me prior to this project. I know feel comfortable talking about such concepts in an academic environment.

My methodology was to first try to understand the concept of machine learning which I still believe was the intuitive starting point. However, now I see that practice with documented and well researched datasets would have given me a much better understanding of the stages of machine learning i.e. pre-processing, training, validation, testing and understanding the results output by the Weka platform. This would have also aided my overall knowledge of the Weka platform and all the features that was at my disposal. I decided to by-pass this stage and focus straight on the power system dataset which had minimal work related to it and a small document which described the dataset in technical terms that I was not accustom to. This made analysing results and attempting create adversarial examples for this dataset difficult and time consuming.

Research is a very different concept to anything I have approached in the past and adapting to the unknown was a challenging concept. This was also the reason I adopted this project as a way of expanding my work methodologies and mental focus.

Adversarial attacks was not a concept I had encountered or worked on before therefore, considerable amounts of time was needed to understand some of the theory behind common adversarial methods. This also meant that implementing these methods even without the need to understanding exactly how they worked was challenging. FGSM is a fairly basic method which can be modified to produce larger perturbations to data by modifying one parameter. Other methods such as JSMA require further knowledge of the foundations of the method to manipulate them in order to produce desired results. Methods more complex like JSMA would have required more time to decipher and implement in an ICS context due to the lack of resources and research focuses on non-image-based adversarial generation.

An aspect of my approach to this project was that slightly lacking the early stages was the extent of the literature reviews surrounding this area. As highlighted, this is a fairly new research area with limited relevant papers especially on non-image focuses adversarial implementations. However, upon further inspection there was a number of papers that discussed related work which would have been beneficial earlier in this project.

The phrase “walk before you can run” is very apt for this is project. Understanding the fundamentals to build upon was the correct approach which I will use again in similar scenarios.

Another unforeseen factor during this project was the global pandemic covid-19. Numerous days/weeks was lost due to the unprecedented times that this pandemic introduced. Moving back home and self-isolating impacted productivity greatly. My mental attitude much like the economy was crashing. Lack of motivation was not a risk that was included in the initial plan risk assessment as this has never been a factor in previous work.

One learning of machine learning that I did not predict was the excessive amount of processing training classifier models and evaluating them took. In the early stages of this project following a literature review of a paper using the same dataset I chose for this project I decided to use the classifier Jrip with ada boost as this classifier performed best on this ICS data. What I failed to notice was that this classifier was only tested on a small fraction of the total dataset due to the lengthy training time. Unfortunately, a number of days was wasted waiting for this model to be trained on a large dataset. Being unpractised and inexperienced with machine learning as a whole I did not realise that this was abnormal and shows how the classifier in question is not suitable to real world applications and was only used to show potential not application.

An element of pre-processing which on reflection may have been slightly overlooked is class balancing, the amount of benign data points is much smaller compared to the amount of malicious datapoints within the dataset I have used for all experiments. Having unbalanced classes could impact the reliability of results.

Focusing more on the technical implementation and experiments, after analysing the results drawn from Fig. 6 perhaps, I should have experimented with smaller values of epsilon to observe the effects on the classifiers performance. All 3 classifiers F-scores dropped significantly when applied to adversarial examples generated with FGSM and an epsilon value of 0.1. Experimenting with epsilon values of below 0.1 could have shown worst performance drops or demonstrated the optimum level of perturbation for greatest adversarial effect. This is definitely an area of this paper that should have been further explored.

XI. References

[1] Teixeira, Marcio et al. “SCADA System Testbed for Cybersecurity Research Using Machine Learning Approach.” Future Internet 10.8 (2018): 76. Crossref. Web.

[2] Baskar, D. and Selvam, P., 2020. Machine Learning Framework For Power System Fault Detection And Classification. 9(02).

[3] A. N. Hasan, P. S. P. Eboule and B. Twala, "The use of machine learning techniques to classify power transmission line fault types and locations," *2017 International Conference on Optimization of Electrical and Electronic Equipment (OPTIM) & 2017 Intl Aegean Conference on Electrical Machines and Power Electronics (ACEMP)*, Brasov, 2017, pp. 221-226, doi: 10.1109/OPTIM.2017.7974974.

[3] R. C. Borges Hink, J. M. Beaver, M. A. Buckner, T. Morris, U. Adhikari and S. Pan, "Machine learning for power system disturbance and cyber-attack discrimination," 2014 7th International Symposium on Resilient Control Systems (ISRCS), Denver, CO, 2014, pp. 1-8.

[4] J. M. Beaver, R. C. Borges-Hink, and M. A. Buckner, “An evaluation of machine learning methods to detect malicious scada communications,” in 2013 12th International Conference on Machine Learning and Applications, vol. 2

[5] https://www.doc.ic.ac.uk/~maffeis/papers/dac19.pdf

[6] Corona, I., Giacinto, G. and Roli, F., 2013. Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues. *Information Sciences*, 239, pp.201-225.

[7] Goodfellow, Ian & Shlens, Jonathon & Szegedy, Christian. (2015). Explaining and harnessing adversarial examples. 1-10.

[8] G. Zizzo, C. Hankin, S. Maffeis, and K. Jones, “Adversarial machine learning beyond the image domain,” in 2019 56th ACM/IEEE Design Automation Conference (DAC). IEEE, 2019a, pp. 1–4.

[9] Zizzo, Giulio, Chris Hankin, Sergio Maffeis and Kevin Jones. “Intrusion Detection for Industrial Control Systems: Evaluation Analysis and Adversarial Attacks.” *ArXiv* abs/1911.04278 (2019b): n. pag.

[10] Szegedy, Christian, Zaremba, Wojciech, Sutskever, Ilya, Bruna, Joan, Erhan, Dumitru, Goodfellow, Ian J., and Fergus, Rob. Intriguing properties of neural networks. ICLR, abs/1312.6199, 2014b. URL http://arxiv.org/abs/1312.6199

[11] Nicolas Papernot, Patrick Drew McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. Distillation as a defense to adversarial perturbations against deep neural networks. CoRR, abs/1511.04508, 2015. URL http://arxiv.org/abs/1511.04508.

[12] William W. Cohen. “Fast Effective Rule Induction” (1995)

[13] A. Robles-Durazno, N. Moradpoor, J. McWhinnie, and G. Russell, “A supervised energy monitoring-based machine learning approach for anomaly detection in a clean water supply system,” in 2018 International Conference on Cyber Security and Protection of Digital Services (Cyber Security). IEEE, 2018

[14] Nicolae, Maria-Irina and Sinn, Mathieu and Tran, Minh~Ngoc and Buesser, Beat and Rawat, Ambrish and Wistuba, Martin and Zantedeschi, Valentina and Baracaldo, Nathalie and Chen, Bryant

and Ludwig, Heiko and Molloy, Ian and Edwards, Ben. “Adversarial Robustness Toolbox v1.2.0”, Journal CoRR, Vol 1807.01069 2018 https://arxiv.org/pdf/1807.01069

[15] Pedregosa, F. , Varoquaux, G.,Gramfort, A.,Michel, V.,Thirion, B. ,Grisel, O.,Blondel, M. ,Prettenhofer, P., Weiss, R.,Dubourg, V. , Vanderplas, J. , Passos, A. , Cournapeau, D. ,Brucher, M. , Perrot, M. and Duchesnay, E. “Scikit-learn: Machine Learning in Python”. Journal of Machine Learning Research., Vol 12, pages 2825 -2830, 2011.

[16] Nicolas Papernot and Fartash Faghri and Nicholas Carlini and Ian Goodfellow and Reuben Feinman and Alexey Kurakin and Cihang Xie and Yash Sharma and Tom Brown and Aurko Roy and Alexander Matyasko and Vahid Behzadan and Karen Hambardzumyan and Zhishuai Zhang and

Yi-Lin Juang and Zhi Li and Ryan Sheatsley and Abhibhav Garg and Jonathan Uesato and Willi Gierke and Yinpeng Dong and David Berthelot and Paul Hendricks and Jonas Rauber and Rujun Long. “Technical Report on the CleverHans v2.1.0 Adversarial Examples Library”. arXiv preprint arXiv:1610.00768. 2018.

[17] L. Huang, A. D. Joseph, B. Nelson, B. I. Rubinstein, and J. D. Tygar, “Adversarial machine learning,” in Proceedings of the 4th ACM workshop on Security and artificial intelligence. ACM, 2011, pp. 43– 58.

[18] Xiaoyong Yuan and Pan He and Qile Zhu and Xiaolin Li “Adversarial Examples: Attacks and Defenses for Deep Learning”. 2017.

[19] “Powersystem dataset readme.pdf,”

[20] Nicholas Carlini and David Wagner, “Towards Evaluating the Robustness of Neural Networks”.. arXiv1608.04644 2016

[21] Uyeong Jang, Xi Wu, and Somesh Jha. 2017. Objective Metrics and Gradient Descent Algorithms for Adversarial Examples in Machine Learning. In Proceedings of the 33rd Annual Computer Security Applications Conference (ACSAC 2017). Association for Computing Machinery, New York, NY, USA, 262–277.

[22] Wes McKinney**.** Data Structures for Statistical Computing in Python, Proceedings of the 9th Python in Science Conference, 51-56 (2010)

[23] https://ics-cert.kaspersky.com/reports/2019/03/27/threat-landscape-for-industrial-automation-systems-h2-2018/

XII. Appendix

1) Confusion matrices for Random Forest, Jrip and J48 for epsilon value 0.1. Left hand column shows results with adversarial training and right hand column shows original training method.

		Predicted	
		0	1
Actual	0	6400	2002
	1	767	19661

Random Forest with Adversarial Training

		Predicted	
		0	1
Actual	0	1888	6514
	1	261	20167

Random Forest Without Adversarial Training

		Predicted	
		0	1
Actual	0	1888	6514
	1	261	20167

Jrip

		Predicted	
		0	1
Actual	0	2065	6337
	1	6097	14331

Jrip

		Predicted	
		0	1
Actual	0	5372	3030
	1	2969	17459

J48

		Predicted	
		0	1
Actual	0	5376	3036
	1	3006	17422

J48

2) Confusion matrices for Random Forest, Jrip and J48 for epsilon value 0.1. Left hand column shows results with adversarial training and right hand column shows original training method.

		Predicted	
		0	1
Actual	0	6400	2002
	1	0	20428

Random Forest

		Predicted	
		0	1
Actual	0	6439	1963
	1	1708	18720

Random Forest

		Predicted	
		0	1
Actual	0	1888	6514
	1	0	20428

Jrip

		Predicted	
		0	1
Actual	0	2065	6337
	1	6448	13980

Jrip

		Predicted	
		0	1
Actual	0	5372	3030
	1	3039	17389

J48

		Predicted	
		0	1
Actual	0	5376	3026
	1	3065	17363

J48