Machine Learning Methods for Anomaly Detection in Industrial Control Systems

Johnathan Tai
College of Engineering &
Computer Science
Syracuse University
Syracuse, NY
jtai02@syr.edu

Izzat Alsmadi, Ph.D.
Department of Computing
and Cyber Security
Texas A&M, San Antonio
San Antonio, Texas
ialsmadi@tamusa.edu

Yunpeng Zhang, Ph.D.
Department of Information
and Logistics Technology
University of Houston
Houston, Texas
yzhan226@central.uh.edu

Fengxiang Qiao, Ph.D.
College of Science,
Engineering & Technology
Texas Southern University
Houston, Texas
fengxiang.qiao@tsu.edu

***Abstract*—This paper examines multiple machine learning models to find the model that best indicates anomalous activity in an industrial control system that is under a software-based attack. The researched machine learning models are Random Forest, Gradient Boosting Machine, Artificial Neural Network, and Recurrent Neural Network classifiers built-in Python and tested against the HIL-based Augmented ICS dataset. Although the results showed that Random Forest, Gradient Boosting Machine, Artificial Neural Network, and Long Short-Term Memory classification models have great potential for anomaly detection in industrial control systems, we found that Random Forest with tuned hyperparameters slightly outperformed the other models.**

Keywords—Machine Learning, CPS, ICS, Industrial Control System Security, Cyber-Physical System Security, Neural Networks, Anomaly Detection, Deep Learning

I. Introduction

Conflicts between nation-states have evolved in the 21st century. Where bombs and bullets were exchanged between competing entities in the past, now there are cyber weapons and tariffs. Nowhere is this truer than in the conflict between the competing interests in the Middle East. Objectively, one such recipient of this exchange was Iran. Sanctions were deployed to cripple the nation's economy, and cyber weapons were deployed to impede the progress of Iran's nuclear program [10, 19].

Specifically, in 2010, the Stuxnet computer virus was deployed to destroy the nuclear centrifuges used to enrich uranium at Iran's nuclear facility. The computer virus worked by displaying normally functioning nuclear centrifuges in the user interface to technicians while the nuclear centrifuges were induced to spin quickly and slowly in a cyclical fashion, which led to the destruction of these devices [19]. Never before had a nation-state inflicted upon another a cyber weapon with the degree of sophistication and number of zero-days contained in Stuxnet against the industrial control system of a competing nation-state [19]. Thus, Stuxnet marked an escalation in the sophistication of the software-based tools upon which one nation is willing to inflict upon another. Arguably, this was an improvement over the kinetic means of warfare, which almost always led to the loss of life. However, it nonetheless represented an escalation in a space for which the rules of engagement have yet to be defined.

Consequently, the successful deployment of a remote cyber weapon by one nation-state upon another sets a new precedent, which makes industrial control systems fair game for both attack and defense and the search for improved defense mechanisms a worthy endeavor. In other words, a point of no return in the cyber conflict has been reached, and there is no telling how nation-states or rogue adversaries will use this capability. Therefore, defense mechanisms should be built into one nation's own cyber-physical systems, whether they are electrical, sanitation, public use of water, or as a separate public utility essential to everyday living.

II. Problem Statement

Our paper shares the sentiment of industrial cybersecurity researchers and firms such as Dragos, which aim to improve the current security status of Industrial Control Systems (ICS). Ultimately, the goal is to find machine-learning based solutions for detecting anomalies in those systems and to find methods that prevent exploiting such anomalies.

However, a complete review of an industrial control system's defenses would involve an audit of every software, electrical, and mechanical component built into the industrial control system combined with a review of the industrial control system's physical site perimeter. Thus, if this discussion of defending an industrial control system could be metaphorically compared to a real-world scenario where a guard needs to defend his or her building, then this paper focuses on building a better camera. More specifically, a guard would have many cameras to detect intruders at different points of interest in a building that typically includes entry and hallways. Detecting an anomaly for a guard would mean seeing an individual roaming the hallway from the feed of a closed-circuit camera during a time of day in which the presence of a person is atypical – such as late at night.

Intelligent Agent	Detection Mechanism	Normal Conditions	Abnormal Conditions
Guard	Camera	Empty hallway at 1:00 AM	Intruder roaming the hallway at 1:00 AM
System Technician	Machine Learning System	Water level never exceeds 70% between the hours of 9:00 PM to 12:00 PM every day of the week	Water level exceeds 95% between 9:00 PM to 12:00 PM for two days

In this case, the guard is trained to accept one input – the camera feed – to detect anomalies. However, the guard must combine the input from the camera feed with his or her *prior* experience gained observing camera feeds during his or her time on the job. The goal is to understand that a person should not be present at a certain location at a particular moment in time to conclude that there is an intruder in the building. Whereas, the machine learning systems trained in this paper are trained on 59 inputs (water level, power level, temperature, etc.) of time series data in which the model gains experience understanding how data points should behave under *normal* operating conditions. Next, the model is tested against the *abnormal* dataset (the dataset with abnormal data points) to test how well the machine learning model can identify whether the industrial control system is under attack. In this case, the target column can have a value of 0 or 1 – not under attack or under attack. In other words, the model is trained to understand how data points should behave under *normal* operating conditions (the attack column is 0 in the normal dataset) while measuring how well a model is trained depends on the model's ability to accurately predict the attack column during *abnormal* operating conditions (the attack column switches between 0 and 1 for fixed periods).

III. Anomaly Detection in Industrial control systems through machine learning

First, detection of an anomaly requires processing of data points from the system by a human or non-human agent. In the case of Stuxnet, which was found in the Siemens Step7 software that controlled programmable logic controllers at Natanz, the nuclear centrifuges used to enrich uranium appeared to work correctly. However, physically, the centrifuges were induced to spin in a way that led to their destruction [19]. Thus, the first requirement is to ensure that data showing the industrial control system's activity is accurate. Second, if the data is accurate, then there is an opportunity to detect anomalous activity in the system through machine learning.

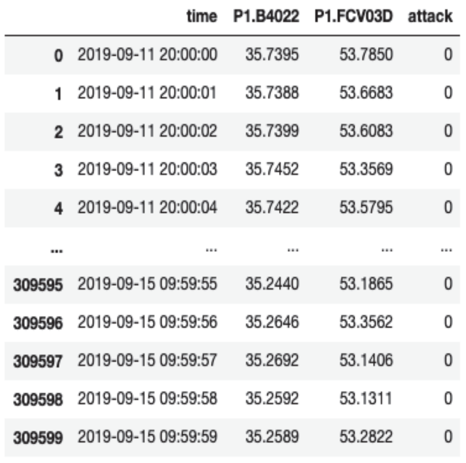
IV. HIL-based Augmented ICS Security Dataset

The dataset discussed in this paper that allows machine learning researchers an opportunity to approach the detection of anomalous activity as a binary classification problem is the HAI Dataset created by researchers at the Electronics and Telecommunications Research Institute in South Korea. The dataset consists of 59 datapoints collected every second from a realistic industrial control system testbed that simulates stream-turbine power generation and hydropower generation. The 59 data points come from the 4 major processes of the testbed: the boiler (P1), turbine (P2), water-treatment component (P3), and HIL simulator (P4), where a data point represents a signal at a moment in time. Table 1 lists a sample dataset from HAI, indicating whether or not the industrial control system is under attack.

Table 1 shows four columns from the 1st Column of the Normal Dataset to the 59th Column of the original dataset. The dataset in Table 1 was before the preprocessing phase, including (1) the time column, (2) the P1.B4022 sensor column, (3) the P1.FCV03D column, and (4) the binary attack column, which indicates whether or not the industrial control system is under attack. In the normal dataset used for training, all of the column "Attack" values are zero.

The data is divided into a normal dataset and an abnormal dataset. In the normal dataset, column "Attack" contains 0's for all rows, and the collected data represents the testbed (e.g., turbine, boiler, etc.) operating under normal conditions, whereas the abnormal dataset represents data points collected during the period under which the industrial control system is under a software-based attack. In other words, in the abnormal dataset, a signal may be much higher or much lower than normal, or the water level may be higher or lower than normal, which might indicate compromise.

Table 1. Sample HAI Dataset Indicating whether or not the Industrial Control System is under Attack



	time	P1.B4022	P1.FCV03D	attack
0	2019-09-11 20:00:00	35.7395	53.7850	0
1	2019-09-11 20:00:01	35.7388	53.6683	0
2	2019-09-11 20:00:02	35.7399	53.6083	0
3	2019-09-11 20:00:03	35.7452	53.3569	0
4	2019-09-11 20:00:04	35.7422	53.5795	0
...	...	...	...	...
309595	2019-09-15 09:59:55	35.2440	53.1865	0
309596	2019-09-15 09:59:56	35.2646	53.3562	0
309597	2019-09-15 09:59:57	35.2692	53.1406	0
309598	2019-09-15 09:59:58	35.2592	53.1311	0
309599	2019-09-15 09:59:59	35.2589	53.2822	0

Note: The data in this table is from the project's .ipynb notebook [16]

V. Time Series Anomaly Detection

Given that normal data points are arranged in chronological order and that the industrial control system operates in fixed, periodic cycles, that the target column is binary, then there is an opportunity to approach the problem as a sequence classification problem [6, 7, 21]. Research has been conducted in this area to find models that can predict anomalous activity in datasets that possess these characteristics – some of which are more suited to account for time and some that are not, including:

- Gaussian Naïve Bayes – "probabilistic classification models that are able to quantify the uncertainty in predictions by providing posterior probability estimates" using Bayes Theorem ($P(y|x) = (P(x|y)P(y) / P(x))$) [17].
- Random Forest – "Random forests attempt to improve the generalization performance by constructing an ensemble of *decorrelated* decision trees. Random forests build on the idea of bagging to use a different bootstrap sample of the training data for learning decision trees. However, a key distinguishing feature of random forests from bagging is that, at every internal node of a tree, the best splitting criterion is chosen among a small set of randomly selected attributes" [17].
- Random Forest (GridSearchCV): Same as above, but the best hyperparameters were found with GridSearchCV, which algorithmically finds the best estimator.
- Gradient Boosting Machine – "Boosting is an iterative produced used to adaptively change the distribution of training examples for learning base classifiers so that they increasingly focus on examples that are hard to classify" [17].
- Gradient Boosting Machine (GridSearchCV): Same as above, but the best hyperparameters were found with GridSearchCV, which algorithmically finds the best estimator.
- Artificial Neural Network – "Artificial neural networks (ANN) are powerful classification models

that are able to learn highly complex and nonlinear decision boundaries purely from the data. They have gained widespread acceptance in several applications such as vision, speech, and language processing" [17].

- Long Short-Term Memory – "Long Short-Term Memory networks, or LSTMs for short, can be applied to time series forecasting" [6]. They are a type of recurrent neural network with feedback connections compared to artificial neural networks, which are feedforward. There "are many types of LSTM models that can be used for each specific type of time series forecasting problem" [6].
- Long Short-Term Memory Autoencoder – "Autoencoders are a type of self-supervised learning model that can learn a compressed representation of input data" [5].

For every model researched in the experiment, either the entire dataset or selected features were passed to the model. In the case of neural networks, the entire dataset was passed to the model, but only selected features were chosen in the case of all other models. Furthermore, multiple models were tested to find the model that best predicts the presence of anomalous activity as a potential indicator of compromise at a given moment in time. However, to carry out this experiment and to execute a thorough and accurate survey of the models, data preprocessing and exploratory data analysis was conducted first.

VI. Data Preprocessing

In the data preprocessing and exploratory data analysis steps, the data was separated according to the method described in [7], which states that "in the case of anomaly detection, the normal traffic pattern is defined in the training phase. In the testing phase, the learned model is applied to new data, and every exemplar in the testing set is classified as either normal or anomalous" [6]. In this step, the DateTime values were also converted from strings into Python DateTime objects.

VII. Exploratory Data Analysis

In the exploratory data analysis step, the per column distribution and correlation matrix of the columns from the normal dataset were analyzed as shown in Fig. 1 for per column distribution and in Fig. 2 for correlation matrix.

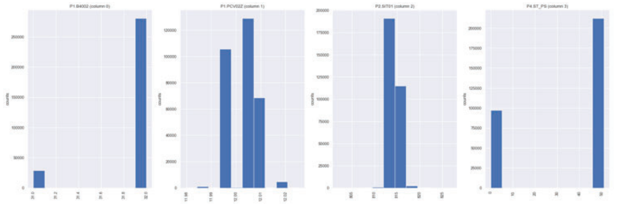


Fig. 1. Per Column Distribution

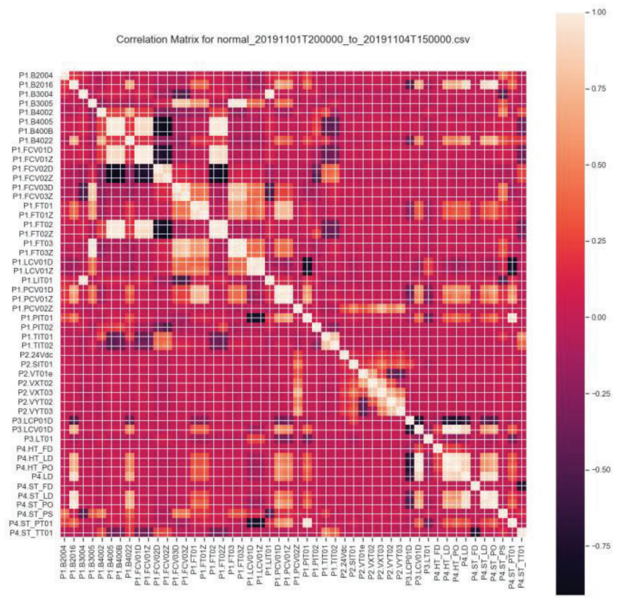


Fig. 2. Correlation Matrix

Columns P1.B4022 (temperature demand to follow P1.B4005 and electrical load from the steam-turbine model measured in Celcius), P1.FCV03D (position command for FCV03 measured in percent), and P1.FCV03Z (current position of FCV03 valve measured in percent) from the boiler process were compared to find the difference between the data points in the normal time-series data (Fig. 3) and the abnormal time series data (Fig. 4).

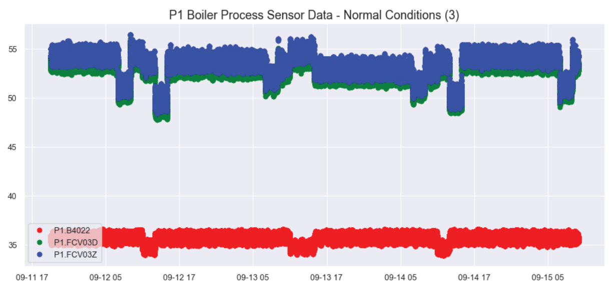


Fig. 3. P1 Boiler Process Sensor Data – Normal Conditions

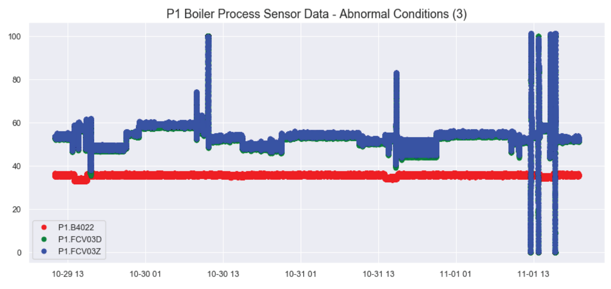


Fig. 4. P1 Boiler Process Sensor Data – Abnormal Conditions

When comparing the three data points in Fig. 3 and Fig. 4, we can see that there are significant anomalies in the behavior of these data points when the system is under attack.

Similarly, this difference in behavior is also shown in the industrial control system's turbine process sensor data. Fig. 5 and Fig. 6 show a comparison between the P2.VT01 (Shaft-vibration-related y-axis displacement near the first mass wheel measured in µm) and P2.VYT03 (Shaft-vibration-related y-axis displacement near the second mass wheel measured in µm) shows significant differences in behavior between the data points when the system is operating under normal conditions and when the system is under attack.

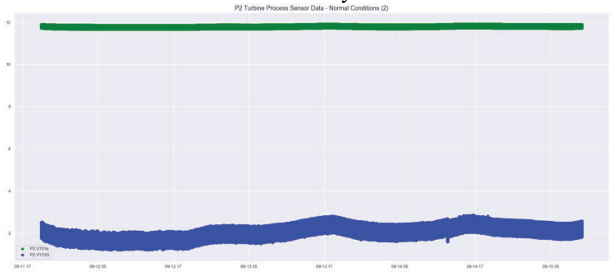


Fig. 5. P2 Turbine Process Sensor Data – Normal Conditions

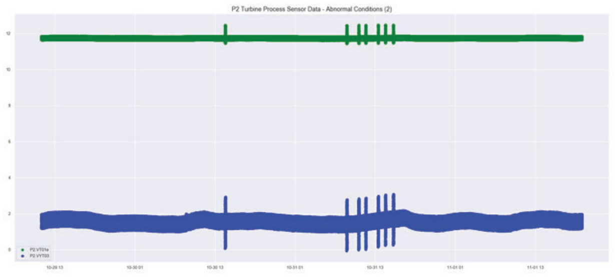


Fig. 6. P2 Turbine Process Sensor Data – Abnormal Conditions

We can also see from the P3.LT01 (Water level in upper tank measured in %) in Fig. 7 and Fig. 8 that, the water level in the industrial control system under attack exceeds 80% whereas the water level in the industrial control system operating under normal conditions never exceeds the mid-70% range.

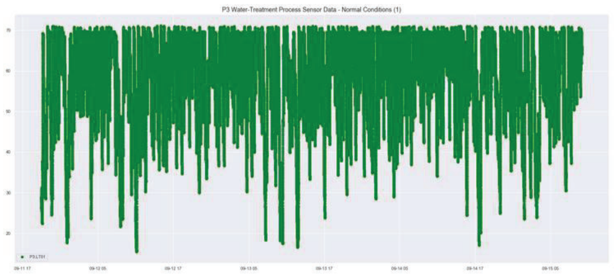


Fig. 7. P3 Water-Treatment Process Sensor Data – Normal Conditions

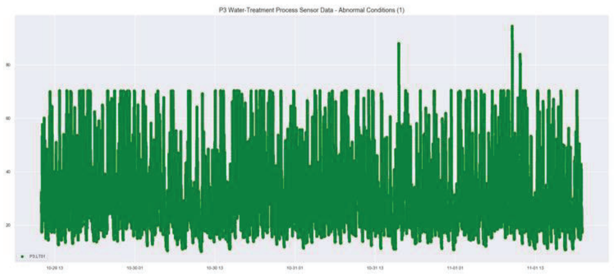


Fig. 8. P3 Water-Treatment Process Sensor Data – Abnormal Conditions

However, we can see that the data points from the P4 Hardware-In-the-Loop simulation in Fig. 9 and Fig. 10 do not lead to any obvious conclusions. In other words, there are no easily discernible differences between the data captured from the P4 sensors operating under normal conditions and the P4 sensors operating under abnormal conditions. Therefore, we can conclude that some amount of feature engineering is needed to remove features from the dataset that are not strong predictors of the attack column.

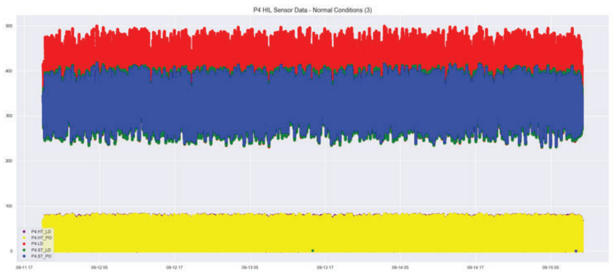


Fig. 9. P4 HIL Sensor Data – Normal Conditions

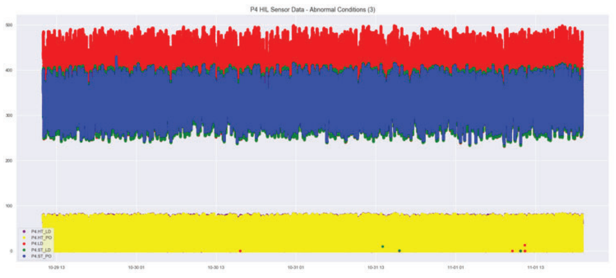


Fig. 10. P4 HIL Sensor Data – Abnormal Conditions

VIII. Feature Selection

Common feature selection techniques were applied to reduce the number of features from 59 to 9. The techniques included the application of the SelectKBest function in scikit-learn using f_classif as the test. Following the 9 best predictors' selection, the 9 predictors were scaled and projected onto a 5-dimensional subspace through principal component analysis.

In principal component analysis, the components were selected based on the proportion of variance explained by the given component, which resulted in the 5 components that represented the greatest proportion of variance being selected for projection onto the subspace. In the feature selection step, the Python f_classif function was selected due to the presence of negative values in the dataset. If negative values were not present in the dataset, then chi2 would have been chosen as the test.

The 9 highest scoring features from the application of the f_classif test in the SelectKBest function were:

- Heat-exchanger outlet pressure setpoint (P1.B2004),
- Water level setpoint in the return water tank (P1.B3004),
- Heat-exchanger outlet temperature setpoint (P1.B4002),
- Temperature demand to follow P1. B4005 and electrical load from the steam-turbine model (P1.B4022),
- Digital value of FT01 flow transmitter (P1.FT01),
- Water level of return water tank (P1.LIT01),
- Position command for LCV01 valve (P1.PCV01D),
- Current position of PCV01 valve (P1.PCV01Z), and

- User speed demand (P2.SD01).

Following their selection by SelectKBest(), they were reduced to 5 principal components.

IX. Model Training

Next, with the data preprocessed and with the feature selection steps complete, candidate models such as Random Forest, Gradient Boosting Machine, Artificial Neural Network, Long-Short Term Memory, and Long-Short Term Memory Autoencoder models were trained to detect anomalous activity in the industrial control system using the normal dataset (train). The accuracy of these models was tested using the abnormal dataset (test). The model, training time, prediction time, cross-validated mean, cross-validated standard deviation, and accuracy are summarized in Table 2.

Table 2. Anomaly Detection Accuracies of Various Models in Wired Intrusion Detection Systems

Classifier	Notes	Training Time	Prediction Time	Cross-Validated Mean (cv=5)	Cross-Validated Standard Deviation (cv=5)	Accuracy
Gaussian Naïve Bayes	The first model test was Gaussian Naïve Bayes, which yielded an accuracy of 54% on the test dataset. Model hyperparameters were not tuned.	~.1s	~.1s	N/A	N/A	54.0%
Random Forest	Next, a Random Forest classifier was trained with hyperparameters manually set to 20 trees, Gini as a function for measuring node impurity, and random state set to 0.	~.9s	~.1s	76.40%	0.074	82.930%
Random Forest GSCV	{'criterion': 'gini', 'max_depth': 4, 'max_features': 'log2', 'min_samples_leaf': 1, 'min_samples_split': 10, 'n_estimators': 1000}	~109.8s	~8.2s	86.51%	0.054	82.934%
Gradient Boosting Machine	Default Parameters	~583.02s	~.1s	83.61%	0.054	77.58%
Gradient Boosting Machine GSCV	{'max_depth': 7, 'min_samples split': 1000}	~1274.28s	~10-20s	77.18%	0.103	83.63%
Artificial Neural Network	3 hidden layers and 12 nodes per layer	~76s	~10-20s	81.25%	0.036	82.79%
Long Short-Term Memory	1 layer with 10 nodes and 1 dropout layer	~111s	~-20s	82.78%	5.578994750976563e-06	82.81%
Long Short-Term Memory Autoencoder	6 layers – 16 nodes in layer 1, 4 nodes in layer 4, 1 node in layer 3, 4 nodes in layer 4, 16 nodes in layer 5, and 1 node in layer 6.	~809s	~10-20s	82.78%	0.0001536130905151367	82.79%

X. Metrics and Model Evaluation

Figure 11 shows the receiver operating characteristic (ROC) curves for the Random Forest classifier with 20 trees, a Random Forest classifier with 1,000 trees, a Gradient Boosting Machine classifier using default parameters, a Gradient Boosting Machine classifier with max depth of 7 and minimum sample split of 1,000, an artificial neural network with 3 hidden layers and 12 nodes per layer, and a long short-term memory recurrent neural network with 10 nodes and a dropout layer.

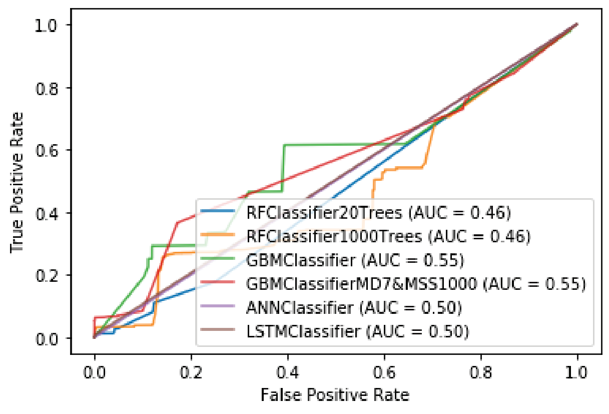


Fig. 11. Receiver Operating Characteristic (ROC) Curves

From this, we can see that the area under the curve for the Gradient Boosting Machine classifier was the greatest despite sharing a similar accuracy with all other classifiers.

Once the models were complete, the models were then compared with models that were surveyed in [7] and some of the results shared by Maglaras and Jiang in [15]. In [7], the authors surveyed different measures such as ANN, Associate Rules Bayesian Network, Clustering k-means, Clustering, Hierarchical Clustering, DBSCAN Decision Trees, GA, Naive Bayes, K-Nearest Neighbors, HMM, Random Forest, and Support Vector Machines [3]. The time complexities and ranges of accuracies gleaned from the document are shown in Table 3.

Table 3. Time Complexities and Accuracy Ranges of Different Models [3]

Classifier	Typical Time Complexity	Accuracy
Artificial Neural Network	O(emnk)	Roughly 80% but varies

Association Rules	O(n^3)	Roughly 100% with 13% FP Rate
Bayesian Network	O(mn)	Roughly 93% with 1.39% FP Rate
Clustering, K-Means	O(kmni)	80%-90% but varies
Clustering, hierarchical	O(n^3)	80%-90% but varies
Clustering, DBSCAN	O(n^3)	80%-90% but varies
Decision Trees	O(mn^2)	98.5% FAR was 0.9%
Genetic Algorithms (GA)	O(gkmn)	100% Best with FAR between 1.4% and 1.8%
Naïve Bayes	O(mn)	Reported 98% and 89% accuracies
K-Nearest Neighbors	O(nlogk)	80%-90% but varies
Hidden Markov Models (HMM)	O(nc^2)	Higher than 85%
Random Forest	O(mnlogn)	99% Range
Sequence Mining	O(n^3)	A real-time scenario where 84% was detected
SVM	O(n^2)	Results enhanced SVM "87.74%"

The results by Buczak et al. [7] show that 90% detection accuracy is achievable in a network-based intrusion detection system (IDS), which means that there is still room for hyperparameter tuning in the theorized Python-based anomaly detection system discussed in this paper.

However, there is room to tune the hyperparameters of the models discussed in this paper, but there is also room to test additional models. In particular, Maglaras and Jiang's research work has shown that "segmentation and clustering algorithms" show great promise in detecting intrusions in SCADA systems [15]. The models tested in this paper fit into the category of neural networks or decision trees. Nonetheless, Maglaras and Jiang note the success of clustering and segmentation algorithms in their paper because these algorithms "do not need to know the signatures" from network activity collected by more commonly used rules-based IDS systems [15]. Thus, not only is their room to tune hyperparameters of the tested models, there is room to test other types of models.

XI. Practical Considerations & Limitations

Furthermore, while there is room to continue to tune the model hyperparameters, there also exists room to consider the limitations of this research, namely:

- The inability of this system to defend itself from zero-day attacks.
- Discussion of how the anomaly detection system would be implemented in a real-world industrial control system.
- Discussion of the computational cost of the researched anomaly detection system in low-level hardware.

For the aforementioned points, complete defense of an industrial control system would include a thorough review of a system's software and hardware components – of which both types of components would undergo formal verification at design time in an ideal scenario. However, this is unlikely to happen in every industrial control system implementation due to the cost of formal verification. Further, history has shown that even the world's most well-tested components may contain critical vulnerabilities– e.g. Spectre and Meltdown [11, 13].

Thus, the true fix to defend ones own industrial control system against adversaries involves disconnecting the industrial control system from the grid while formally verifying software and hardware components at design time and only utilizing parts from trusted suppliers. However, in the absence of this, then an organization should implement a form of detection.

XII. Conclusion

Altogether, research carried out in this paper shows that, the Random Forest, Gradient Boosting Machine, Artificial Neural Network, and Long Short-Term Memory models have great potential for anomaly detection in industrial control systems based on the results of testing various Python-based models against a sample industrial control system dataset in an experimental environment, however Random Forest with hyperparameters tuned with GridSearchCV slightly outperformed all other tested models. As a result, we have identified areas of this paper where there is room for additional research, including the tuning of hyperparameters in the laboratory environment, testing of additional segmentation and clustering models, and additional research into the practical implementation of each model.

References

[1] Alazab, Mamoun, et al. "A Multidirectional LSTM Model for Predicting the Stability of a Smart Grid." *IEEE Access*, vol. 8, 2020, pp. 85454–85463., doi:10.1109/access.2020.2991067.

[2] Alazab, Mamoun, et al. "Malicious Spam Emails Developments and Authorship Attribution." *2013 Fourth Cybercrime and Trustworthy Computing Workshop*, 2013, doi:10.1109/ctc.2013.16.

[3] Athalye, Anish, et al. "Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples." *ArXiv.org*, 31 July 2018, arxiv.org/abs/1802.00420.

[4] Azab, Ahmad, et al. "Machine Learning Based Botnet Identification Traffic." *2016 IEEE Trustcom/BigDataSE/ISPA*, 2016, doi:10.1109/trustcom.2016.0275.

[5] Brownlee, Jason. "A Gentle Introduction to LSTM Autoencoders." *Machine Learning Mastery*, 27 Aug. 2020, machinelearningmastery.com/lstm-autoencoders/.

[6] Brownlee, Jason. *Deep Learning for Time Series Forecasting*. Machine Learning Mastery, 2020.

[7] Buczak, Anna L., and Erhan Guven. "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection." *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, 2016, pp. 1153–1176., doi:10.1109/comst.2015.2494502.

[8] Hwang, Won-Seok, et al. "Time-Series Aware Precision and Recall for Anomaly Detection: Considering Variety of Detection Result and Addressing Ambiguous Labeling." *Time-Series Aware Precision and Recall for Anomaly Detection | Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 1 Nov. 2019, dl.acm.org/doi/abs/10.1145/3357384.3358118.

[9] ICS Security Datasetnovice tier Starter: HAI Security Dataset eb7995d4-c, Python notebook using data from HAI Security Dataset, https://www.kaggle.com/icsdataset/starter-hai-security-dataset-eb7995d4-c, Accessed: Sep., 14th 2020.

[10] "Iran Sanctions - United States Department of State." *U.S. Department of State*, U.S. Department of State, 21 Aug. 2020, www.state.gov/iran-sanctions/.

[11] Kocher, Paul, et al. "Spectre Attacks: Exploiting Speculative Execution." *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, doi:10.1109/sp.2019.00002.

[12] Kurakin, Alexey, et al. "Adversarial Machine Learning at Scale." *ICLR 2017*, 2017.

[13] Lipp, Moritz, et al. "Meltdown." *Communications of the ACM*, vol. 63, no. 6, 2020, pp. 46–56., doi:10.1145/3357033.

[14] Madry, Aleksander, et al. "Towards Deep Learning Models Resistant to Adversarial Attacks." *ArXiv.org*, 4 Sept. 2019, arxiv.org/abs/1706.06083.

[15] Maglaras, Leandros A., and Jianmin Jiang. "Intrusion Detection in SCADA Systems Using Machine Learning Techniques." *2014 Science and Information Conference*, 27 Aug. 2014, doi:10.1109/sai.2014.6918252.

[16] Shin, Hyeok-Ki, et al. "Implementation of Programmable CPS Testbed for Anomaly Detection."

[17] Tan, Pang-Ning, et al. *Introduction to Data Mining*. Pearson Education, 2019.

[18] Tramèr, Florian, et al. "Ensemble Adversarial Training: Attacks and Defenses." *ArXiv.org*, 26 Apr. 2020, arxiv.org/abs/1705.07204.

[19] "Throwback Thursday: Whatever Happened to Stuxnet?: Synopsys." *Software Integrity Blog*, 28 Feb. 2019, www.synopsys.com/blogs/software-security/whatever-happened-to-stuxnet/.

[20] Vasan, Danish, et al. "Image-Based Malware Classification Using Ensemble of CNN Architectures (IMCEC)." *Computers & Security*, vol. 92, 2020, p. 101748., doi:10.1016/j.cose.2020.101748.

[21] Yin, Chuanlong, et al. "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks." *IEEE* Access, vol. 5, 12 Oct. 2017, pp. 21954–21961., doi:10.1109/access.2017.2762418.

[22] Zhang, Chaoyun, et al. "Deep Learning in Mobile and Wireless Networking: A Survey." IEEE Communications Surveys & Tutorials, 2019.

[23] A Visualized Botnet Detection System Based Deep Learning for the Internet of Things Networks of Smart Cities - IEEE Journals & Magazine, ieeexplore.ieee.org/document/8985278.