Anomaly Detection System in Industrial Control System Using Machine Learning

by

Ahammed Sakir Nabil (170021037)
Ahnaf Akif Rahman (170021153)
Imtihan Ahmed (170021025)

A Thesis Submitted to the Academic Faculty in Partial Fulfillment of the Requirements for the Degree of

BACHELOR OF SCIENCE IN ELECTRICAL AND ELECTRONIC ENGINEERING

Department of Electrical and Electronic Engineering
Islamic University of Technology (IUT)
Gazipur, Bangladesh

May 2022

Declaration of Authorship

We, Ahammed Sakir Nabil (170021037), Ahnaf Akif Rahman (170021153), Imtihan Ahmed (170021025), declare that this thesis titled, 'Anomaly Detection System in Industrial Control System Using Machine Learning' and the works presented in it are our own.

We confirm that:

- This work has been done for the partial fulfillment of the Bachelor of Science in Electrical and Electronic Engineering degree at this university.
- Any part of this thesis has not been submitted anywhere else for obtaining any degree.
- Where we have consulted the published work of others, we have always clearly attributed the sources.

Submitted By:

Sakir

Ahammed Sakir Nabil (170021037)

Ahnaf Akif Rahman

Ahnaf Akif Rahman (170021153)

Imtihan

Imtihan Ahmed (170021025)

Anomaly Detection System in Industrial Control System Using Machine Learning

Approved by:

May 10, 2022

Mr. Safayat Bin Hakim

Supervisor and Assistant Professor,
Department of Electrical and Electronic Engineering,
Islamic University of Technology (IUT),
Boardbazar, Gazipur-1704.

Date: May 6, 2022

Table of Contents

List of Tables

List of Figures

List of Acronyms

ACM	Association for Computing Machinery
ADS	Anomaly Detection System
ANOVA	Analysis of Variance
AUC	Area Under the ROC Curve
BMC	Baseboard Management Controller
CA,	Certificate Authority
CAN	Campus Area Network
CD	Cyber Defence
CIA	Central Intelligence Agency
CPS	Cyber Physical System
CV	Cross Validation
DCS	Distributed Control System
DL	Deep Learning
DOA,	Direction of Arrival
DT	Decision Tree
EN	European Standards
FN	False Negative
FP	False Positive
GB	Gradient Boosting
GBDT,	Gradient Boosted Decision Trees
GBM,	Gradient Boosting Machine
GNB	Gaussian Naïve Bayes
GSCV	Grid Search Cross Validation
H0	Null Hypothesisi
HAI	HIL-based Augmented ICS
HIL	Hardware-in-the-loop
ICPHM	IEEE International Conference on Prognostics and Health Management
ICS	Industrial Control System
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
IO	Input-Output
IS,	Information System
IT	Information Technology
KNN	K Nearest Neighbors
LR	Logistic Regression

MCC	Matthews Correlation Coefficient
MD,	Mobile Internet Device
MIDS	Multifunctional Information Distribution System
ML	Machine Learning
MLP	Multilayer Perceptron
MPS	Malware Protection System
NB	Naïve Bayes
OCC	One-Class Classification
OCSVM	One Class Support Vector Machine
ODBASE	International Conference on Ontologies, Databases, and Applications of Semantics
PL	Program Lock
PLC	Programmable Logic Controllers
RF	Random Forest
RN	Regulatory Notice
RNA	Reconnaissance Network Appliance
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic Curve
SCADA	Supervisory Control and Data Acquisition
SG	Security Guide
SMOTE	Synthetic Minority Over-sampling Technique
SW	Spyware
TN	True Negative
TP	True Positive
USA	United States of America
XGB	Extreme Gradient Boosting
XGBOOST	Extreme Gradient Boosting

Acknowledgements

First of all, we would like to thank our parents for their patience, inspiration and motivation that has helped us to come this far in our life. Although we can never completely repay our debts to them, this research is a small token of appreciation for all the hardships they have endured throughout our journey.

We would like to express our heartfelt gratitude to our thesis advisor, Mr. Safayat Bin Hakim for his guidance and motivation. One cannot ask for a better advisor. His tricks and tips were the most useful in solving many of the issues during the research. Apart from everything else, his attention to minute details is really praiseworthy.

Last but not the least, we appreciate the EEE department of Islamic University of Technology for enabling us to conduct a research based on Machine Learning which is very important for Industry 4.0

Abstract

An industry is composed of various types of machines and instruments interconnected through a system of network performing in harmony following specific instructions assigned to specific nodes or equipment. Industrial control system refers to the whole environment that keeps everything included in the industrial system in order. Like any other system, industrial control system is also prone to attacks which might result in massive loss. In this paper, six machine learning algorithms have been applied for detecting the presence of anomaly in industrial control system using HIL-based Augmented ICS (HAI 21.03) Security Dataset. The dataset has been analyzed using analysis of variance to extract 50 of the most important features from each sample in the dataset. All the machine learning models' performances are recorded, and a full comparative analysis for hyperparameter optimization, downsampling-upsampling with hyperparameter tuning, and without hyperparameter tweaking is shown. Random search cross validation has been employed for hyperparameter optimization, and synthetic minority oversampling technique has been used for upsampling. In terms of several evaluation metrics like accuracy, recall, precision, F1-score, Receiver Operating Characteristic (ROC) Area Under the Curve (AUC) and specificity, satisfactory performances have been observed. In addition to these evaluation metrics, which have also been used by other researchers in previous studies, we have evaluated the performance of our models using Geometric Mean(G-Mean) and Matthews Correlation Coefficient (MCC), which are considered two of the most important evaluation metrics in imbalanced datasets. Using our proposed approach, a maximum recall score of 99.77% and an F1-score of 99.50% have been achieved, which are significantly higher than previous studies. Maximum G-Mean of 99.89% and MCC of 0.9950 have been obtained by the application of K-Nearest Neighbors (KNN) model. Therefore, our proposed approach has the prospect to be an efficient method for detecting anomalies in industrial control systems and taking appropriate actions.

Chapter 1

Introduction

This chapter provides an introduction to the Industrial Control System and the scopes and significance of deploying Machine Learning Algorithms to prepare an autonomous Anomaly Detection System.

1.1 Problem Statement and Motivation

Industry modernization requires digitization of manufacturing along with technological advancement. The emerging Industry 4.0 manifests the incorporation of classic industrial processes with digital technology. Employment of advanced technologies is increasing rapidly as the production lines are becoming large-scale, resulting in a more critical production infrastructure. To name just a few major strategic large-scale infrastructures, industrial control systems (ICSs) are mainstreamed in smart grids, electricity supply, transportation, and water treatment [1]. The conventional CIA triad (Confidentiality, Integrity, and Availability) is assigned a converse order in the case of ICSs, which are part of the cyber-physical system (CPS) [2]. An ICS, as opposed to an IT system where the primary concern is the confidentiality of data, is dependent on availability in order to ensure human safety and fault tolerance [3]. Furthermore, along with legacy industrial system modernization, modern cynicism has spread like a pandemic. The fast-track digitization and integration of legacy ICSs exposes these systems to intruders, creating new vulnerability surfaces [4]. Systems regulating physical and potentially harmful operations necessitate security, as they are extremely prone to malevolent activity by unscrupulous attackers.

The gravity of ensuring the security of ICSs cannot be overstated. ICSs play a crucial part in power plants, and a plethora of computerized systems operate on a SCADA (Supervisory Control and Data Acquisition) framework [5]. Besides SCADA, ICSs contain control systems like, Distributed Control Systems (DCS) and Programmable Logic Controllers (PLC) [5]. As evidenced by previous occurrences,

ICSs are highly susceptible to cyber-attacks. An early cyberattack on SCADA systems resulted in a massive explosion on the Trans-Siberian pipeline in 1982 [6]. ICS security flaws have been disclosed in a number of cases over the consecutive years since that causal incident. Several viruses that attempted ICSs have been detected in recent years. The most well-known virus in this category is arguably Stuxnet. This computer worm succeeded in inducing 984 nuclear centrifuges to self-destruct at an Iranian uranium enrichment site by targeting PLCs [7]. ICSs were targeted by BlackEnergy, a variant of an infamous trojan family, in 2014 [8]. A number of different sectors, including the media, energy, mining, railways, airports, and rails, have been attacked in Ukraine [8]. On December 23, 2015, a BlackEnergy3 attack prompted the Kyiv Power Distribution business to disconnect 30 substations for three hours, culminating in several hours of blackouts in the vicinity [8]. Following the haphazard shutdown of a petrochemical plant in Saudi Arabia in 2017, TRITON emerged as a serious malware threat [9]. This virus reprograms several unique PLCs that are intended for safety purposes, leading them to fail [9]. According to a Kaspersky Lab analysis, 39.2% of industrial devices safeguarded by Kaspersky's solutions were targeted in the second half of 2016, indicating that threats to ICSs are becoming rampant [10]. Every successful attack against ICS is a tribulation for the involved organizations as they suffer financially. These repercussions include operational halts, equipment impairment, business waste, intellectual property deceit, safety risks, and extensive health jeopardy [4].

It is critical to deploy innovative security-by-design measures to prevent such catastrophes. If such measures are not feasible, prevention or mitigation techniques must be implemented. A robust testing infrastructure is necessary to build a new security-by-design paradigm [4]. In general, researchers use ad hoc models of real ICSs to create realistic infrastructures inside a structured setting known as test bench [4]. The generation of a new test bench is laborious. It inflates a series of challenges, including implementation expenditures, sharing competences, and authenticity [4]. Nowadays, researchers deploy machine learning strategies to detect misconduct or prospective attacks through training classification algorithms in order to build prevention and mitigation techniques. In this research, we also used machine learning classification methods on a security dataset based on industrial systems to detect anomalies. Gaussian Naive Bayes (GNB), K- Nearest Neighbor (KNN), Logistic Regression (LR), Random Forest (RF), Extreme Gradient Boosting (XGBOOST), and Multilayer Perceptron

(MLP) are the classification algorithms we trained for our research-based simulation. The algorithms are implemented considering their characteristics and functionalities.

All the algorithms that we employed here are supervised learning algorithms. GNB is a variant of Naive Bayes that follows Gaussian normal distribution assuming no co-variance between dimensions [11]. KNN is an effective method for classification though it lacks efficiency in many applications for example dynamic web mining for a large repository [12]. Besides, its effectiveness solely depends on the selection of the best value of k [12]. Another supervised learning algorithm is LR, which is very efficient to train [13]. For estimating the categorical dependent variable using a given set of independent variables, LR is used [14]. It is an easily implementable as well as interpretable algorithm [13]. RF is employed in high dimensional and multi-source data reduction applications [15]. In case of accuracy, RF is surpassed by XGBOOST [16], a scalable end-to-end tree boosting system [17]. XGBOOST gives less prediction error than RF or any boosting algorithm [16]. It is extremely efficient if speed is a concern without affecting accuracy [16]. Designing this algorithm is aimed at solving linearly inseparable problems as well as approximate solutions to continuous functions [18]. Other notable applications of MLP include pattern categorization, recognition, prediction, and approximation [18].

The easiest technique of data collection is to capture and distribute data from real ICSs to academics. Despite this, all such processes seem to be frequently critical as well as vital to societal structure, so the approach may become difficult in a number of ways. Due to the physical damage that can occur as a result of an attack, they are extremely difficult to execute in real-world conditions [4]. Furthermore, privacy is an issue: private organizations may be hesitant to provide system data obtained from their ICSs. Disclosing system data may result in intellectual property theft as well as expose the infrastructure's vulnerabilities in reality, alluring cybercriminals. Data from a test bench can be generated and distributed among researchers in order to analyze and enhance the effectiveness of various classification methods. Such collections become known as datasets which might include physical measurements as well as network traffic. Because of their ease of use and accessibility, datasets are a good testing solution. They are, nevertheless, hard in many ways, such as the generation process and lack of modularity. However, due to a paucity of industrial control system datasets and

challenges in selecting suitable anomaly detection models, these insights have subsided in practice [19]. To address this, our study used the HIL-based Augmented ICS (HAI 21.03) dataset. The National Security Research Institute released this dataset for conducting research on the security and integrity of ICSs [20].

1.2 Research Objectives

This paper addresses the concerns of industrial cybersecurity researchers and firms that seek to ameliorate the contemporaneous security status of ICS. Besides developing an anomaly detection model, the objective of this research is to explore what factors the Industrial Control Systems can use to detect anomalies. The study strategy includes data pre-processing, supervised machine learning models fitting, and assessing the classification accuracy of each model. Analysis of Variance (ANOVA) correlation test was performed for feature selection process. Down sampling and Synthetic Minority Over-sampling Technique (SMOTE) after up sampling were conducted for data manipulation. The confusion matrix, recall, Area Under the Curve (AUC), and Receiver Operating Characteristics (ROC) curve are the traditional approaches for assessing model effectiveness. Apart from these, some other performance metrics such as: Geometric Mean (G-Mean), Matthews Correlation Coefficient (MCC) are also generated. Ideally, our vision is to obtain machine-learning-based solutions for anomaly detection in ICSs and develop prevention measures against such exploitation.

1.3 Literature Review

Several studies on intrusion detection, anomaly detection, and similar activities have been conducted utilizing control and security systems. Based on voltage measurement data, Choi et al. [21] has proposed an Intrusion Detection System (IDS) that utilizes the unique characteristics of electrical signals to detect Controller Area Network (CAN) intrusions within vehicles. Due to their reliance on only one type of variable to detect suspicious activity in their IDS, they experienced a high rate of false positives despite having a well-designed approach. Generally, IDS monitors network traffic in an ICS and attempts to identify unusual activity in data packet transmissions [5]. Whereas, the Measurement Intrusion Detection System (MIDS) probes unusual

activity in the system's measurement data rather than monitoring network traffic [5]. Mokhtari et al. [5] a MIDS technique through which the system can detect any anomaly even if an attacker conceals it within the control layer of the system. They constructed an ML model based on supervised learning that can distinguish between normal and anomalous actions within an Industrial Control System. In addition, a HIL (Hardware-in-the-loop) based test bench was created for analyzing units for power generation and also exploiting attack datasets.

Tai et al. [22] examined multiple machine learning models against the HIL-based Augmented ICS (HAI) dataset to find the best performing model indicating anomalous activity in an ICS under attack. Whereas, Zhong et al. [23] executed anomaly detection on the basis of a real gas turbine data, using Isolation Forest (iForest). Ahmed et al. [24] also employed the iForest algorithm but for detecting anomalies in a smart grid environment. For anomaly detection, Kim et al. [25] proposed the Clustered Deep One-Class Classification (CD-OCC) model, a hybrid of the clustering algorithm and Deep Learning (DL) model. The proposed model was evaluated by using the SWaT and HAI datasets. Using the HAI dataset for learning, Kim et al. [26] projected a technique on anomaly detection in an ICS employing both supervised and unsupervised machine learning algorithms. This dataset has 2 versions, both of them were utilized in [27]. Kim et al. [27] evaluated their ensemble model for anomaly detection using the HAI Datasets 20.07 and 21.03. Following that, different performance evaluation techniques were conducted to compare the detection performance of the single model as well as the ensemble Recurrent Neural Networks (RNN) model. HAI dataset was employed in several researches for training machine learning models as it is publicly accessible on the internet. There has been much more research regarding the detection, mitigation, and prevention of anomalies employing various techniques based on industrial environments. From an architectural and security standpoint, Conti et al. [4] provided a meticulous overview of ICSs. Moreover, the most effective IDS algorithms on each dataset are reported in order to create a foundation inside this field's design approach. In our study, we attempted to train some supervised machine learning algorithms on the latest version of the HAI dataset in order to detect anomalies with a level of accuracy that outperformed all previous research. As part of our work, we aimed to improve the recall parameter, which represents how accurately we can identify relevant data.

Chapter 2

Machine Learning Algorithms

A data that does not follow the distribution of the rest of the data, as if it were formed by a different system". As a result, anomaly detection entails looking for patterns in data that suggest unusual behavior. Models for anomaly identification, on the other hand, are difficult to create because it's difficult to define normal areas that encompass all conceivable normal behaviors, and data often contains noise that resembles true anomalies [10]. Furthermore, because CPS datasets are large and most of the data is normal, developing detection algorithms is more challenging. Nonetheless, a significant amount of effort has gone into developing the CPS anomaly detection algorithm.

This research integrates a number of Machine Learning algorithms for intrusion and anomaly detection which have been discussed in the following sections.

2.1 Gaussian Naïve Bayes (GNB) :

GNB is a Naive Bayes derivative which accepts consistent information and utilize the Gaussian standard allocation.

The Bayes principle serves as the foundation for the Naive Bayes set of autonomous ML classifying methods. It's a simple classifying technique having a strong core. They're handy once the diversity of the sources becomes significant. The NB Classifier is similarly applied to address difficult identification problems.

2.2 Logistic Regression (LR) :

The goal of LR is predicting the probabilities of two alternative outcomes in classification situations. It's a characteristic of the linear regression model for classification problems that's quite substantial. The logistic sigmoid function modifies the logistic regression output values to provide a statistical significance which may be converted to two or additional independent categories.

Figure 2.1 is the graphical representation of how a Logistic Regression algorithm functions. The chance of an event occurring within a given class is estimated using logistic regression, a statistical classification model. Spite of the notion that the term "regression" appears in its name, LR becomes a widely employed algorithm. A discriminator becomes a boundary imposed for forecasting the category of information a set of data corresponds with. This categorization possibility is determined using the linear model, and that is basically a sigmoid function.

$$\hat{p}_\theta = h_\theta(x) = \sigma\left(x^T\theta\right)$$

Here,

$$\sigma(t) = \frac{1}{1+e^{-t}}$$

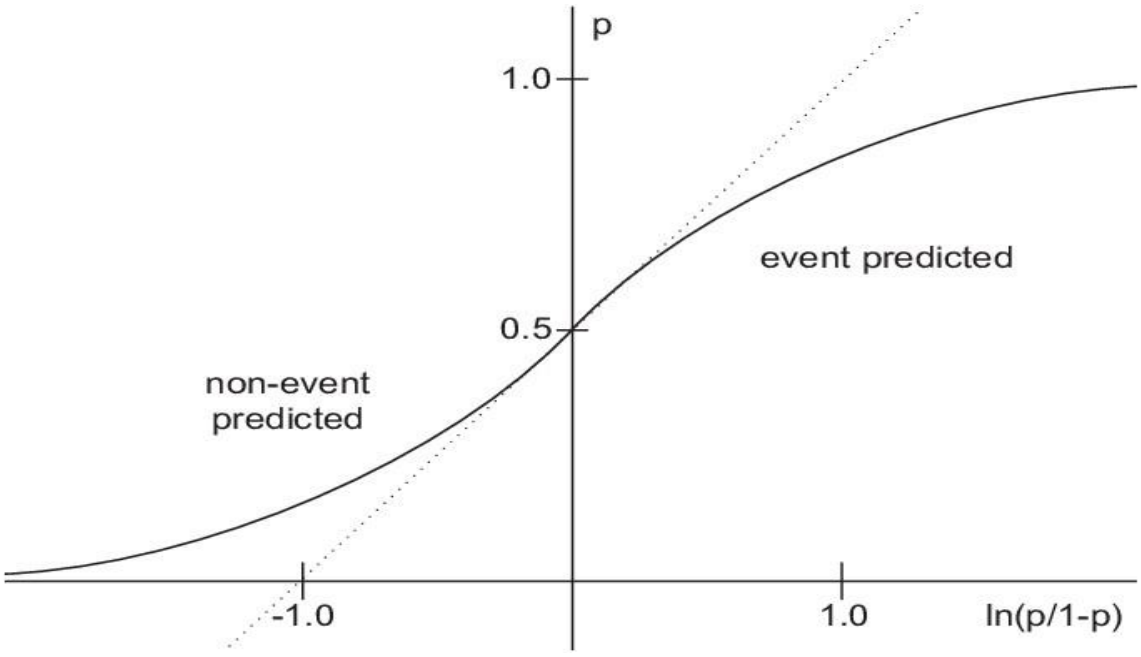

Figure 2.1: Logistic Regression Algorithm

2.3 K-Nearest Neighbors (KNN) :

One of the very fundamental and extensively used guided ML techniques includes KNN. The KNN method considers that the fresh information and current circumstances are comparable and allocates the contemporary instances to the category that is most similar to the prior groups.

It does not train specific sample; rather, each event becomes forecasted for belonging to the class with the most k-nearest neighbors.
The value of K is chosen so that accurate predictions are made with the fewest possible errors. The nearest datapoint to the viewing platform can be considered the far more comparable towards the observation item, which is why distance is used as a metric to determine resemblance. Distance measures come in a wide range of forms.
Euclidean distance:

$$d_{euclidean} = \sqrt{\sum_{i=1}^{n}\left(x_i^2 - y_i^2\right)}$$

Manhattan distance:

$$d_{manhat\,\tan} = \sum_{i=1}^{n}\left|x_i - y_i\right|$$

2.4 Random Forest (RF) :

RF is a regression and categorization learning technique which operates through constructing many decision trees during training and provides output classes for individual trees. RF is premised on the idea that a huge proportion of significantly statistically independent systems operating as a group can surpass all of the other structural algorithms individually. Even without hyper parameter adjustment, it produces reasonable prediction results. Bagging is a minor adjustment that uses the orthogonalized trees via producing new spate of schemas using rebooted batches from

trained data. During bootstrap, it selects a limited combination of multiple columns from across all characteristic elements. Bootstrapping reduces variation while increasing bias. Prediction of unknown inputs can be determined by the formula below:

$$\hat{f} = \frac{1}{B}\sum_{b=1}^{B} f_b(x')$$

Here, B= Optimal number of trees

Also, uncertainty of the prediction can be written as:

$$\sigma = \sqrt{\frac{\sum_{b-1}^{B}\left(f_b(x') - \hat{f}\right)^2}{B-1}}$$

2.5 Extreme Gradient Boosting :

XGBoost is a GB-relying multiple classifier ML technique. XGBoost focuses on model performance and computational speed, and it comes with a lot of advanced features. The model is very effective in case of regression, categorization, order, and forecasting when dealing with datasets that are tiny to big in size and are organized as well as hierarchical. It performs a range of tasks by enhancing sparse features in datasets using gradient convergence framework. XGBoost offers GBDT, GBM, a concurrent tree enhancing approach which rapidly and reliably addresses a number of data analytics problems.

2.6 Multilayer Perceptron :

MLP is a feedforward artificial cognitive infrastructure composed of several stacks of perceptrons. It has three layers where the first one is input layer, second one is hidden layer, and the final one is output layer. This method utilizes a non-linear

activating mechanism for transforming leaden inputs to neuronal outcomes. Inside the work, the activating parameters used were sigmoid functions.

Figure 2.2 shows the working procedure of MLP. First hyperbolic tangent has a range of -1 to 1, while the next one has linear functions. In order to decrease error, training entails modifying the model's parameters, such as weights and biases. To perform those weight and bias adjustments related to the mistake, backpropagation is used.

$$y(v_i) = \tanh(v_i)$$

$$y(v_i) = (1 + e^{-v_i})^{-1}$$

Backward propagation is used in perceptron development, and the reduced loss function at sending end j following optimization algorithms may indeed be represented as:

$$\varepsilon(n) = \frac{1}{2}\sum_j e_j^2(n)$$

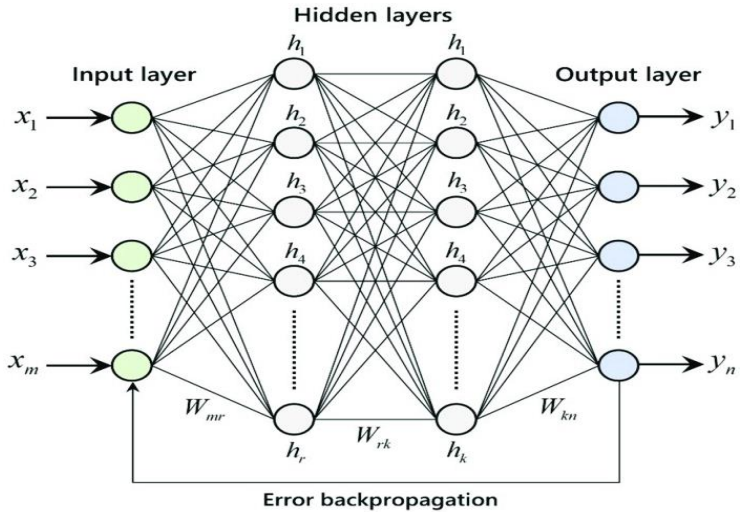

Figure 2.2: Multilayer Perceptron Algorithm

Chapter 3

Methodology

In this chapter the workflow that has been used to construct an effective Machine Learning based anomaly detection system that can be deployed in an Industrial Control System has been discussed.

3.1 Dataset Description

This study utilizes the latest version of the HIL based augmented ICS test bench (HAI) dataset [28][29], which is freely accessible over the Internet. This database was created to aid in the detection of anomalies in CPSs and it contains both normal and abnormal datasets, with the normal abnormal dataset collected in response to various attack scenarios.

A complex process system was built using an Hardware- in-the-loop simulator in September 2018 to combine the systems associated with the three test benchs: GE's turbine test bench, Emerson's boiler test bench, and FESTO's "MPS" modular production system, which are a year earlier released laboratory-scale CPS test benchs. Thermal power generation and pumped-storage hydropower generating were both simulated using the HIL simulator. The water level, flow rate, pressure, temperature, water feed pump, and heater control for the boiler process, which was water-to-water heat transfer at low pressure and mild temperature, were all controlled using Emerson's Ovation distributed control system (DCS). There was a rotor kit test bench inside the turbine system that closely resembled the behavior of an actual rotating machine and was controlled by GE's Mark VIe DCS to monitor vibration and control speed. During the HIL simulation, water was first pumped into the higher reservoir and after that into the lower reservoir based on a pumped-storage hydropower generating model, and it was controlled by a Siemens S7-300 PLC for water level and pump control. A

dSPACE® SCALEXIO system is used for the HIL simulations, which is coupled to the real-world processes through ET200 remote IO devices and a Siemens S7-1500 PLC. HAI 21.03, which was released in 2021, is built on a tighter-knit HIL simulator that delivers clearer attack effects with more strikes.

Table 3.1 shows a summary of the different versions of HAI 21.03 dataset in a tabulated form. Compared to the previous version 20.07, HAI 21.03 has almost 1.5 times more data points, as well as 21 additional features recorded. For 50 attacks, this dataset comprised ICS operating data from both normal and aberrant scenarios. This supplied more quantitative information and covered a wide range of operational scenarios, as well as improved insights into the physical system's dynamic changes.

Table 3.1 Information about the HAI 21.03 dataset

Dataset: HAI 21.03				
Released Date: 2021-03-25				
Total Data: 1,323,608				
Data Points: 78 points/second				
Type	***Files***	***Interval (hour)***	***Size (MB)***	***Attack Count***
Normal Data (1,314,661)	train1.csv	60	110	-
	train2.csv	63	116	
	train3.csv	229	245	
Abnormal Data (8,947)	train1.csv	12	22	5
	train2.csv	33	61	20
	train3.csv	30	55	8
	train4.csv	11	20	5
	train5.csv	26	47	12

3.2 Data Visualization

The dataset consists of 3 train files and 5 test files. The dataset has 84 features. The train files together hold 921,603 samples of which none are anomalous samples. The test files together hold 402,005 samples. It consists of 8947 anomalous samples. Hence, in total there are a sum total of 1,323,608 samples. Among these, there are 13,146,661 normal samples and 8,947 anomalous samples. The anomaly rate stands at 0.67% which is very common in Industrial Control System. Adversarial attacks are an

irregular phenomenon. But one such attack may result in a catastrophe. We first visualized our dataset through various parameters.

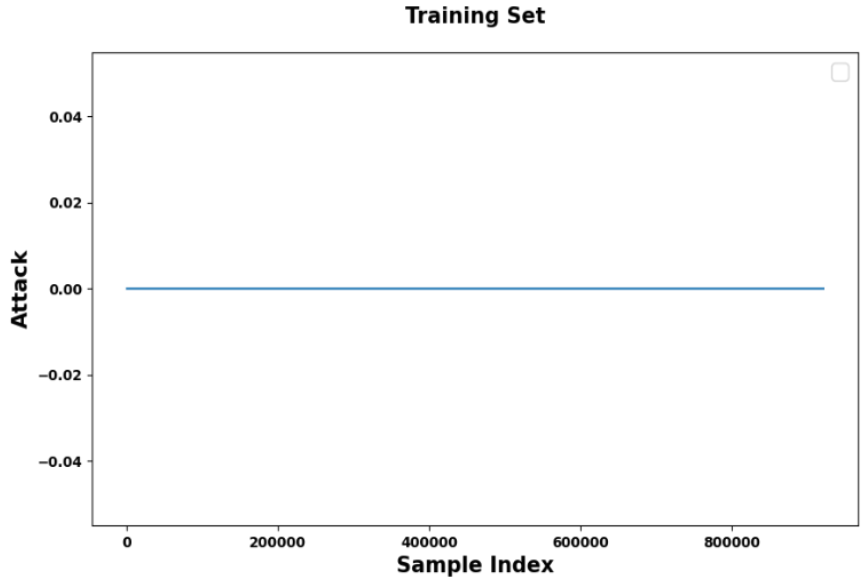

Figure 3.1: Attack instances in training set

Fig. 3.1 shows a summary of all the samples from the 3 train files. It can be seen that there are no anomaly data point and hence it just a horizontal straight line.

Fig. 3.2 shows a summary of all the datapoints of the 5 test files. It can be seen that there are multiple vertical straight lines. These vertical straight line signifies the existence of anomaly at that instance.

We then took all the datapoints under a single dataframe and then observed the data. Fig. 3.3 is a summary of all the data points together. It can be seen that the anomalous samples are all shifted towards the right. This has happened because the anomalous samples were all in the test set and the test set has been appended to the training set that contained no anomaly data points.

From the initial observation of data it can be concluded that the dataframe under consideration is highly skewed and imbalanced.

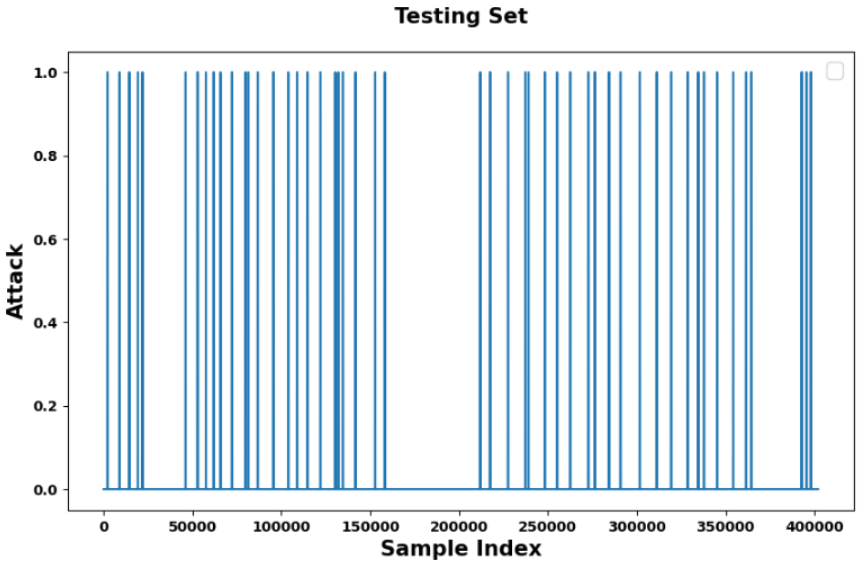

Figure 3.2: Attack instances in testing set

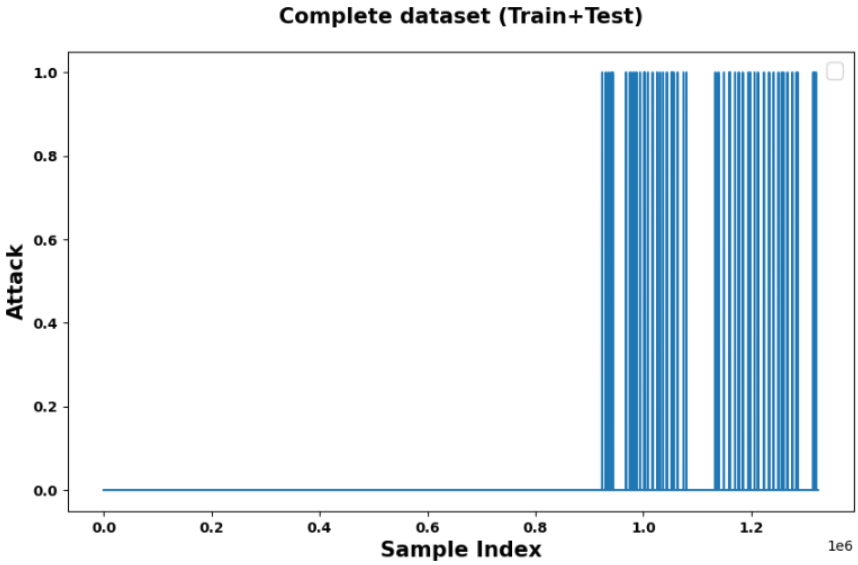

Figure 3.3: Attack instances in joined dataset

3.3 Pipeline

In order to get the best out of our study, we took a 3-step observation and optimization procedure. At first, we worked on the default/raw dataset without any data manipulation. We observed the result from the unmanipulated dataset and then moved forward to hyperparameter tuning. Lastly, to best serve our purpose and to get the most optimized result with least runtime, we performed data manipulation. The overall pipeline diagram is shown in Fig. 3.4.

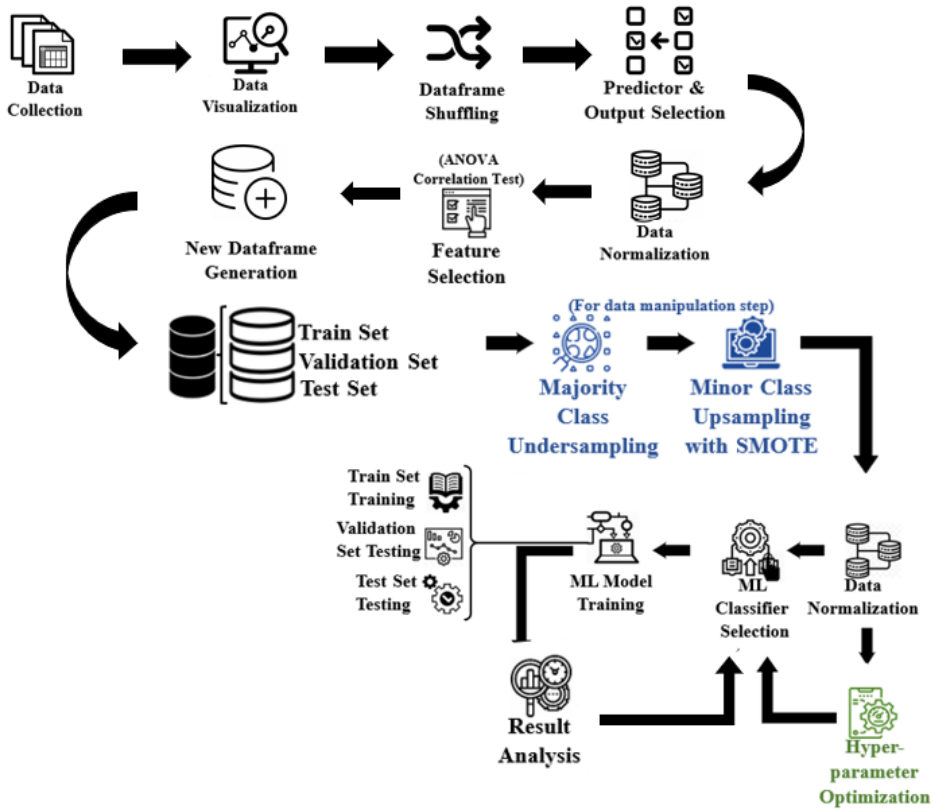

Figure 3.4: Proposed pipeline for an efficient ML based ADS in ICS

3.4 Data Preprocessing

3.4.1 Elimination of Skewness

To get rid of the skewing, we used a fixed random state to shuffle our data frame. This was done to evenly distribute the anomalous data points across the whole data frame without changing any of the values in original dataset.

Fig. 3.5 shows the anomaly data distribution of the first 1000 data points after shuffling which shows clearly how the skewness of the dataset has been solved.

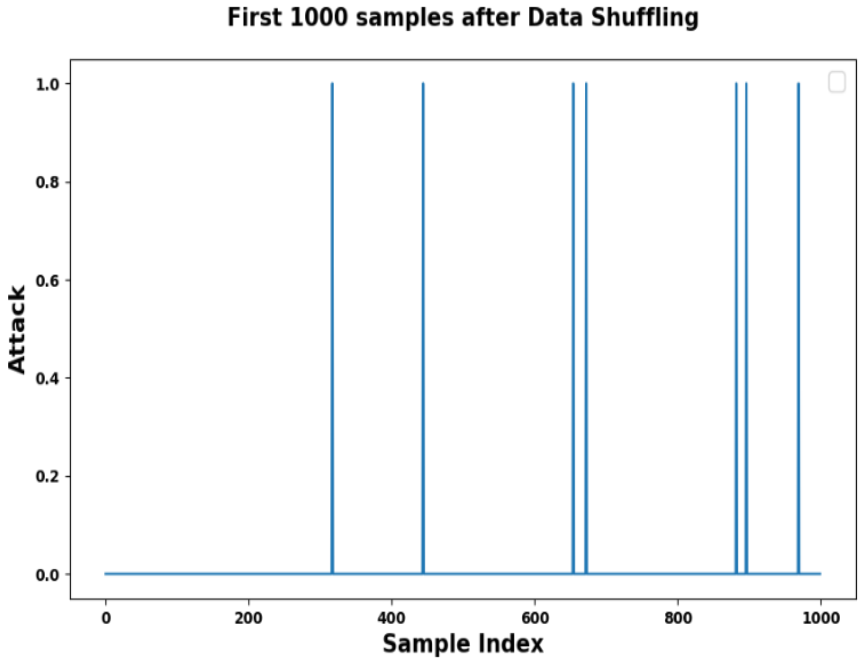

Figure 3.5: Attack instances in first 1000 samples after data shuffling

3.4.2 Predictor/Output Selection

After that the predictors and output variables were chosen from the data frame. Four features were dropped at first that mentioned the time and which among the 3 processes of HIL ICS was being affected as that was not a concern for this study. Among 79 remaining features, the feature ‘attack’ was chosen as output variable and the rest of the features were chosen as our predictors.

3.4.3 Feature Selection using ANOVA Correlation Test

ANOVA correlation testing is a method of determining the contribution of each numerical predictor to the categorical output variables. As opposed to the generic correlation testing where the dependence and independence factor does not play much significance, ANOVA test helps correlate in terms of dependent and independent features. For a numerical input to categorical output ANOVA correlation coefficient is one of the best evaluation criteria for understanding the relation among predictors and outputs. The dataset contains a large number of samples as well as a significant number of features. For minimizing runtime, feature selection has been done using ANOVA correlation testing, in the next step. The test result returned 19 null/ not applicable features.

From the rest 60 features, top 50 features were chosen in terms of ANOVA test score for the next phase. The top 50 features along with their ANOVA test scores (also called co-efficients) have been provided in table 3.2 and has been graphically shown in figure 3.6.

Then a new dataframe was created with only the selected features from the original dataset.

A train and test split were done once more. This time training set remained 80% as before, but the previous test set was divided into equal parts of validation set (10%) and test set (10%). Normalization was also incorporated on the dataset. And that marks the end of the data pre-processing part of our first among 3 core optimization phases

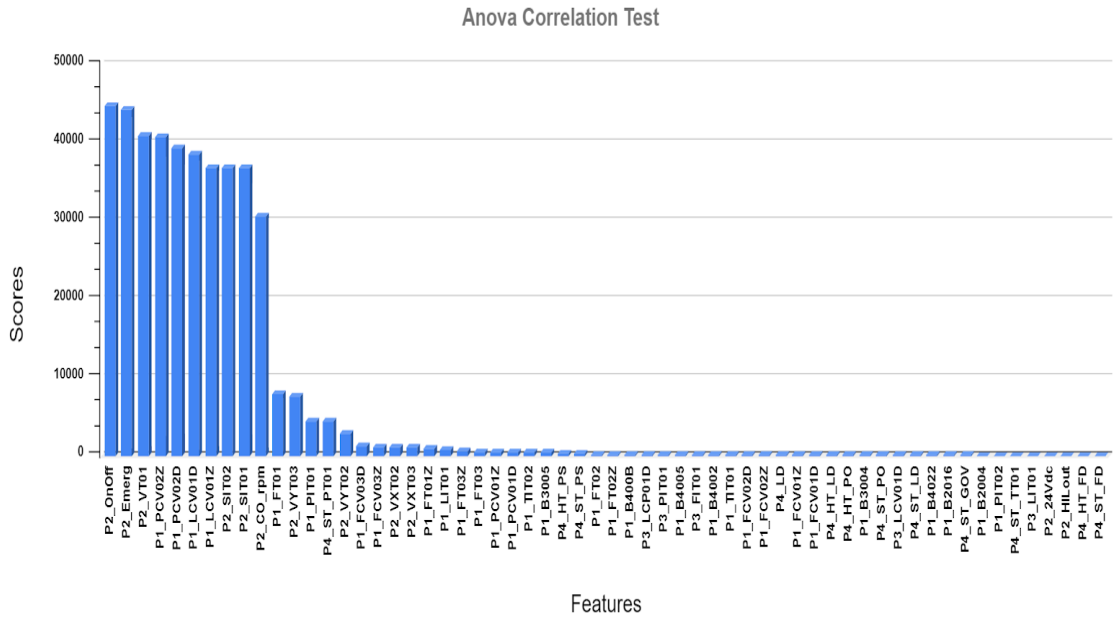

Figure 3.6: ANOVA test scores for feature selection

Table 3.2 ANOVA test scores of the top 50 features

Features	***Scores***	***Features***	***Scores***
P2_OnOff	44618.44	P1_TIT02	349.301
P2_Emerg	44216.74	P1_B3005	346.2579
P2_VT01	40910.69	P4_HT_PS	214.6442
P1_PCV02Z	40698.08	P4_ST_PS	214.6442
P1_PCV02D	39278.66	P1_FT02	170.1509
P1_LCV01D	38527.71	P1_FT02Z	166.8306
P1_LCV01Z	36818.05	P1_B400B	166.4269
P2_SIT02	36776.5	P3_LCP01D	157.7815
P2_SIT01	36770.21	P3_PIT01	112.0158
P2_CO_rpm	30534.62	P1_B4005	109.8485
P1_FT01	7832.106	P3_FIT01	102.8014
P2_VYT03	7615.269	P1_B4002	101.1173
P1_PIT01	4417.682	P1_TIT01	90.20611
P4_ST_PT01	4380.628	P1_FCV02D	83.10977
P2_VYT02	2844.932	P1_FCV02Z	73.51914
P1_FCV03D	1135.057	P4_LD	68.23568

P1_FCV03Z	1120.14	P1_FCV01Z	60.94246
P2_VXT02	1115.839	P1_FCV01D	59.33446
P2_VXT03	972.5336	P4_HT_LD	57.83826
P1_FT01Z	812.7218	P4_HT_PO	53.81125
P1_LIT01	791.6849	P1_B3004	51.5366
P1_FT03Z	516.6129	P4_ST_PO	48.40944
P1_FT03	462.8482	P3_LCV01D	45.51998
P1_PCV01Z	391.1211	P4_ST_LD	42.34632
P1_PCV01D	379.3134		

3.5 ML Model Training and Testing

3.5.1 Without Hyperparameter Tuning

The machine learning model was trained using 6 machine learning classifiers i.e., KNN, XGB, GNB, LR, RF, MLP with no tuning. Performance evaluation metric results were obtained for each classifier.

3.5.2 With Hyperparameter Optimization

The 2nd phase of the optimization phases is hyperparameter tuning. Hyperparameter optimization was done using RandomizedSearchCV. RandomizedSearchCV helps to find out the most appropriate parameter values for a classifier using ‘fit’ and ‘score’ method. With data preprocessing steps remaining the same, the machine learning model was then trained with the same ML classifiers, but this time each of the classifiers’ hyper-parameters were tuned to achieve maximum recall value.

False negative outcomes are the most undesirable for any anomaly detection or intrusion detection system. Maximizing recall value minimizes the false negative rate. The evaluation metrics were again recorded to observe how each of the classifiers performed.

The tuned hyper-parameters thus obtained for each classifiers has been shown in Table 3.3.

3.5.3 With data Manipulation

From the previous analysis it is evident not all classifiers are suitable for attack - Data manipulation for maximizing performance. There are a few additional steps conducted in the preprocessing phase. After the new data frame was created with selected features, dataset was split into 80% training set,10% validation set and 10% testing set with stratification along the output feature. It was observed that the training set consisted of 1,051,728 normal data samples
and 7,158 anomalous data samples, validation set consisted of 131,467 normal data samples and 894 anomalous data samples, while test set consisted of 131,466 normal data samples and 895 anomalous data samples. The huge imbalance in the training set is a major cause for performance reduction in ML classifiers. Undersampling was done first to reduce the number of normal data points. 1,000,000 normal data points were randomly selected first. After that SMOTE was employed to increase the number of anomalous data points of the training set to 10,000. SMOTE stands for Synthetic Minority Oversampling Technique. SMOTE is a type of data augmentation for the minority class through which new samples are synthesized. Thus, the anomaly rate was increased to 1%. Further up sampling was avoided to maintain consistency with real life anomaly rate. After that the testing set was equally divided into validation and testing set and the machine learning model was again trained with tuned hyperparameters (obtained from the previous phase of optimization) of the mentioned classifiers. This resulted in significant improvement as observed from the evaluation metrics.

Table 3.3 Tuned Hyper-parameters for each classifier

<table>
<tr><td>KNN

algorithm='auto',
leaf_size=50,
metric='minkowski',
metric_params=None,
n_jobs=None,
n_neighbors=1, p=1,
weights='uniform'</td><td rowspan="2">XGBoost

alpha=3,
base_score=0.5,
booster='gbtree',
colsample_bylevel=1,
colsample_bynode=1,
colsample_bytree=1,
eta=3,
gamma=0.1,
learning_rate=0.1,
max_delta_step=0,
max_depth=3,
min_child_weight=1,
missing=None,
n_estimators=100,
n_jobs=1,
nthread=None,
objective='binary:logistic',
random_state=0,
reg_alpha=0,
reg_lambda=3,
scale_pos_weight=1,
seed=None,
silent=None,
subsample=1,
verbosity=1</td></tr>
<tr><td>DT

ccp_alpha=0.0,
class_weight=None,
criterion='gini',
max_depth=100,
max_features=None,
max_leaf_nodes=None,
min_impurity_decrease=0.0,
min_samples_leaf=1,
min_samples_split=20,
min_weight_fraction_leaf=0.0,
random_state=0,
splitter='best'</td></tr>
<tr><td>GNB

priors=None,
var_smoothing=1e-07</td><td>LR

C=100, class_weight=None,
dual=False, fit_intercept=True,
intercept_scaling=1, l1_ratio=None,
max_iter=100,
multi_class='auto', n_jobs=None,
penalty='l2',
random_state=1, solver='liblinear',
tol=0.0001, verbose=0,
warm_start=False</td></tr>
</table>

Table 3.3(continued) Tuned Hyper-parameters for each classifier

RF	***MLP***
bootstrap=True, ccp_alpha=0.0, class_weight=None, criterion='gini', max_depth=None, max_features='auto', max_leaf_nodes=None, max_samples=None, min_impurity_decrease=0.0, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0, n_estimators=73, n_jobs=None, oob_score=False, random_state=None, verbose=0, warm_start=False	activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9, beta_2=0.999, early_stopping=False, epsilon=1e-08, hidden_layer_sizes=(50, 100, 50), learning_rate='constant', learning_rate_init=0.001, max_fun=15000, max_iter=200, momentum=0.9, n_iter_no_change=10, nesterovs_momentum=True, power_t=0.5, random_state=None, shuffle=True, solver='adam', tol=0.0001, validation_fraction=0.1, verbose=False, warm_start=False

Chapter 4

Performance Metrics

The effectiveness of a predictive model is determined using performance metrics. In classification operations, the most widely used metric is accuracy. However, for unbalanced datasets, the accuracy values are biased towards the ruling class, which can be very misleading [30].

Class-specific counters are more convenient for non-equilibrium classification operations. The confusion matrix helps to convey these points more clearly (shown in Table II). The binary classification problem has two possible consequences. False (0) and True (1)

Table 4.1 Confusion Table format

	Predicted		
Actual	*Categories*	*False (0)*	*True (1)*
	False (0)	TN	FP
	True (1)	FN	TP

The term TP refers to correct predictions for positive (1) class instances, while TN indicates the correct predictions for negative (0) class instances. FP shows the prediction of negative cases as positive and FN shows the prediction of positive cases as negative. In this study, anomalous instances are labeled as 1 to represent a minority

class. Normal instances are considered as 0 which form a majority class. Evaluation key figures are defined based on these four basic terms. Table 4.1 provides a clearer insight in the confusion matrix format.

4.1 Accuracy

Accuracy is a metric that expresses how the model will perform throughout all classes in general. It is calculated by dividing the number of accurate predictions by the number of total predictions.

$$Accuracy = TP+TNTP+TN+FP+FN$$

Where,
TP = True Positive ,
TN = True Negative ,
FP = False Positive,
FN = False Negative.

The majority class of the dataset dominates a model's classification accuracy [31]. If the data is imbalanced, the classifier can achieve high accuracy. However, this high level of accuracy is too optimistic to accurately reflect the performance of the classifier. As a result, various evaluation metrics that describe possible class imbalance scenarios should be considered in order to reflect the classifier's performance.

4.2 Recall/Sensitivity

The probability of correctly identifying a positive test sample or a minority test sample is indicated by recall [32]. A higher recall value indicates that the classifier is effective in predicting instances of the minority class. As the dataset is hugely imbalanced, we have put more emphasis on this evaluation metric.

$$Recall/Sensitivity = (TP+TN)/(TP+TN+FP+FN)$$

4.3 Precision

Precision represents the proportion of detected minorities that are actually correct. Precision is affected by the order of the classes as it combines results from both major and minor samples [32].

$Precision = = TP/(TP+FP)$

4.4 F1-Score

F1-score consolidates two metrics (precision and recall) of a classifier into a solitary metric by using their harmonic mean. F1-score is used to evaluate classifications that have a significant proportion of false negatives and false positives. [33].

$F1\ Score = 2 \times Precision \times Specificity/(Precision+Specificity)$

4.5 Specificity

Specificity is an indicator of how accurately the negative class was foreseen [34]. It's quite often used in the context of sensitivity. Because negative classes obtain the major part of instances, models tend to have higher specificity and yet lower sensitivity.

$Specificity = TP/(TP+FP)$

4.6 Receiver Operating Characteristic (ROC) and Area Under the Curve (AUC)

ROC plot combines sensitivity and specificity into one figure, allowing researchers to evaluate the accuracy of the proposed metrics on the y-axis graphically. For any reliable indicator, a transition in specificity produces a permanent effect in sensitivity, which is schemed as a curve in the ROC graph. AUC stands for Area Under the Curve. The value of ROC AUC varies from 0 to 1. Here 0.5 expresses a 50-50 assumption and 1.0 indicates a 100% reliable forecasting model [35].

4.7 Geometric Mean (G-Mean)

G-mean was suggested in [36] as the product of sensitivity and specificity. A low G-Mean score indicates poor classification results if only one of the two classes has a greater classification error.

$G\text{-}mean = \sqrt{(Sensitivity \times Specificity)}$

4.8 Matthews Correlation Coefficient (MCC)

MCC is a reliable analytical metric that only yields successful outcome if the forecasting showed good performances in all four categories of the confusion matrix. However, it does not distinguish between majority and minority class results. The value can be anywhere between -1 and +1 [37]. A prediction with a +1 value is the best possible prediction, a prediction having a 0 MCC value is just an arbitrary prediction, and a prediction with a -1 value is the worst possible prediction.

$MCC = (TP \times TN - FP \times FN)/((TP+FP)(TP+FN)(TN+FP)(TN+FN))$

Chapter 5

Results and Discussion

Table 5.1 Confusion Matrix for GNB, KNN, LR, RF, XGB MLP

			Algorithm											
	Case	***Predicted***	***GNB***		***KNN***		***LR***		***RF***		***XGB***		***MLP***	
Actual		Categories	0	1	0	1	0	1	0	1	0	1	0	1
	1	0	130920	546	131453	13	131444	22	119904	11562	125139	6327	131461	5
		1	425	470	24	871	585	310	47	848	204	691	279	616
	2	0	130920	546	131459	7	131432	34	116179	15287	123729	7737	131455	11
		1	425	470	3	892	529	366	45	850	196	699	62	833
	3	0	130906	560	131459	7	131422	44	114432	17034	120764	10702	131443	23
		1	419	476	2	893	519	376	38	857	170	725	43	852

Table 5.2 Comparison of recall, precision & accuracy (before tuning, after tuning & upsamplingdownsampling+tuning)

	Performance Metrics								
Algorithms	***Recall (%)***			***Precision (%)***			***Accuracy (%)***		
	1	2	3	1	2	3	1	2	3
GNB	52.51	52.51	53.18	46.26	46.26	45.95	99.27	99.27	99.26
KNN	97.31	99.66	99.77	98.53	99.22	99.22	99.97	99.99	99.99
LR	34.63	41.89	42.01	93.37	91.50	89.52	99.54	99.57	99.57
RF	94.75	94.97	95.75	6.83	5.27	4.79	91.23	88.41	87.10
XGB	77.20	78.10	81.01	9.85	8.29	6.34	95.07	94.00	91.79
MLP	68.82	93.07	95.20	99.2	98.7	97.37	99.79	99.94	99.95

Table 5.3 Comparison of f1 score, specificity & roc auc (before tuning, after tuning & upsampling- downsampling+tuning)

Algorithms	Performance Metrics								
	F1 Score (%)			*Specificity*			*ROC_AUC*		
	1	2	3	1	2	3	1	2	3
GNB	49.19	49.19	49.30	99.58	99.58	99.57	0.7605	0.7605	0.7638
KNN	97.92	99.44	99.50	99.99	99.99	99.99	0.9865	0.9983	0.9989
LR	50.53	56.53	57.19	99.98	99.97	99.97	0.6731	0.7043	0.7099
RF	12.75	9.98	9.12	91.20	88.37	87.04	0.9298	0.9167	0.9400
XGB	17.46	14.98	11.77	95.19	94.11	91.86	0.8620	0.8610	0.8643
MLP	81.27	95.80	96.27	99.99	99.99	99.98	0.8441	0.9653	0.9759

Table 5.4 Comaprison of g-mean & mcc (before tuning, after tuning & upsampling-downsampling+tuning)

Algorithms	Performance Metrics					
	G-Mean (%)			*MCC*		
	1	2	3	1	2	3
GNB	72.32	72.32	72.77	0.4892	0.4892	0.4906
KNN	98.65	99.83	99.89	0.9791	0.9944	0.9950
LR	58.85	63.94	64.81	0.5672	0.6102	0.6117
RF	92.96	91.61	91.29	0.2417	0.2088	0.1985
XGB	85.73	85.73	86.26	0.2648	0.2423	0.2126
MLP	82.96	96.47	97.56	0.8254	0.9582	0.9625

5.1 Simulation Results

Python has been used to carry out the simulations, with Google Colab serving as the coding platform. The confusion matrices are shown in Table III for three cases:

1. Without hyperparameter tuning
2. With hyperparameter tuning
3. With upsampling-downsampling and hyperparameter tuning

The confusion matrices showed in Table 5.1 provided the performance metrics for anomaly detection. Evaluation indicators such as accuracy, precision, recall, G-Mean, MCC, F1 score, ROC- AUC and specificity were calculated and reported from Table 5.2 to Table 5.6 for all three cases.

In Table 5.2, recall, precision and accuracy are shown with default hyperparameters first, then with efficient hyperparameters and finally with down sampling-up sampling along with hyperparameter optimization
According to the results, with a maximum recall score of 99.77%, KNN outperformed the other classifiers in aspects of recall whereas RF and MLP were close with recall score of 95.75% and 95.20% respectively. However, the lowest recall was achieved by LR with about 42.01%. In the case of precision, KNN is the most promising classifier with a score of 99.22%. Besides, MLP (99.20%) and LR (93.37%) also showed respectable results. RF showed the lowest precision score with about 6.83 For accuracy, it can be seen that all of the algorithms performed similarly, although KNN outperforms the others with a value of 99.99%.

In Table 5.3, F1-score, ROC-AUC and specificity values are shown for three different cases. KNN showed the highest value of F1-score with over 99.50%, followed by MLP (96.27%). RF showed the lowest F1 score value (12.75%). For specificity, all the algorithms performed well. MLP (99.99%) and KNN (99.99%) showed highest specificity values among them. KNN even performed the best regarding ROC score with a value of 0.9989, closely pursued by MLP and RF with 0.9759 and 0.9400, correspondingly. LR had the lowest ROC score, which was around 0.7099.

In Fig. 5.1, the ROC of the classifiers after undersampling and SMOTE has been shown. In Table 5.4, G-mean and MCC are shown for all three cases. In terms of G-mean, KNN outperformed the other models with a value of 99.89%. MLP and RF are the other virtually implemented ML models, having G- mean values of 97.56% and 97.56%, respectively.

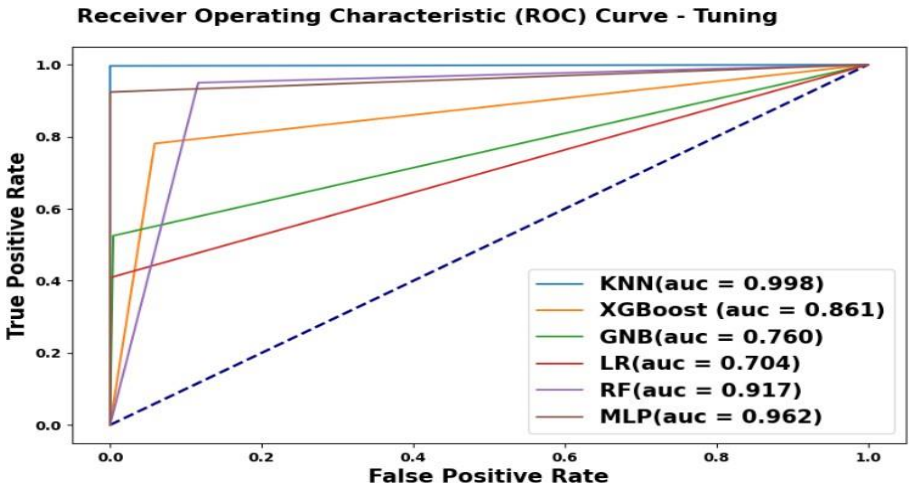

Figure 5.1: ROC curve for SMOTE after tuning

LR achieved the lowest G-mean of around 64.81%. For MCC, KNN again performed the best with a value of 0.9950 whereas MLP, with 0.9625, came up just short. RF (0.2417) and XGB (0.2648) both performed poorly. As the recall score is the basis of hyperparameter tuning, recall values increased for each algorithm except GNB after hyperparameter optimization.

The default hyperparameters were good enough for GNB. So, recall value remained same for this algorithm. After performing downsampling and upsampling with hyperparameter optimization, recall values got increased more. As F1 score depends on recall value, it maintained the same pattern for all algorithms except RF and XGB.

Deceasing value of precision is the reason for falloff of F1 score in these algorithms as F1 score also depends on precision. As the hyperparameters were tuned on the basis of recall, some algorithms showed lower values of precision after optimizing the hyperparameters. The same thing also happened for some other evaluation metrics. But none of the values decreased significantly. Like recall, G-mean values increased after hyperparameter adjustment for all algorithms except RF and GNB.

The values increased further considerably after performing downsampling-upsampling with hyperparameter tweaking. MCC values almost followed the same pattern.

From Table 5.2 – 5.4, it is evident that among all evaluation metrics, highest values have been achieved after performing downsampling-upsampling along with hyperparameter optimization. In Fig. 5.2, the comparison between recall and F1 score, and in Fig. 5.3, the comparison between G-mean and MCC, is shown, respectively.

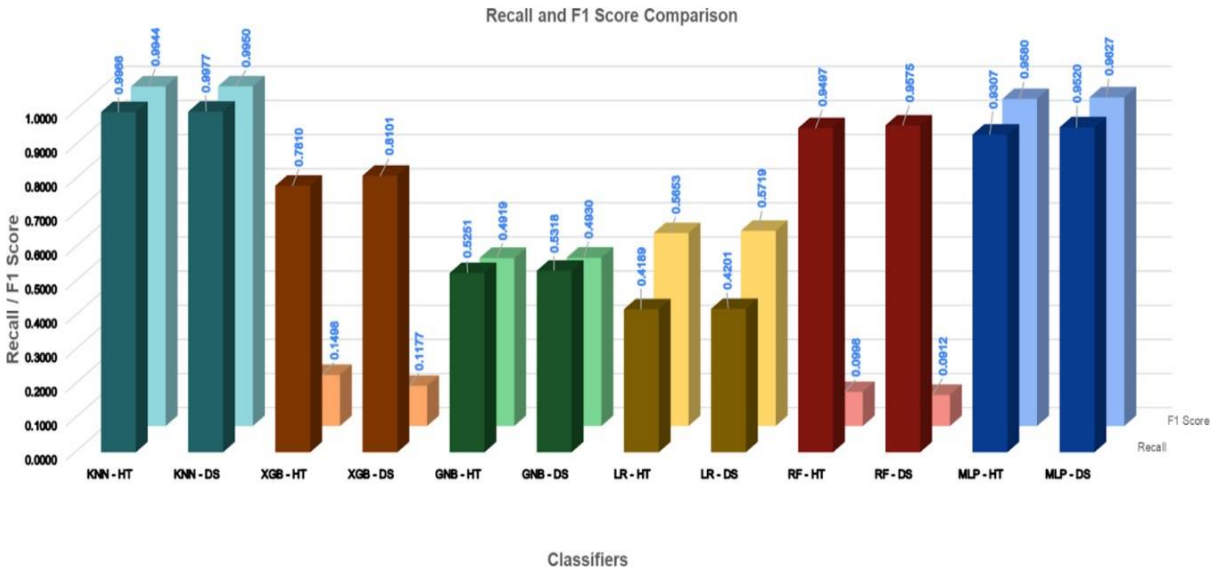

Figure 5.2: Comparison of recall and F1 score

From figure 5.2, the comparative analysis of recall and F1 score can be observed. For KNN, GNB and MLP; F1 score and recall acted in the same way. That means precision value also acted in the same way for these three algorithms. But on the contrary, for XGB and RF, recall values were much higher than F1 score which indicate poor precision values. But in case of LR, F1 score values were much higher than recall value. This indicates strong performances of precision for LR.

From figure 5.3, the comparative analysis of G-Mean and MCC score can be observed. For KNN, LR and MLP; G-Mean and MCC acted in the same way. But on the contrary, for XGB and RF, G-Mean values were much higher than MCC. For GNB, G-Mean values were slightly better than MCC

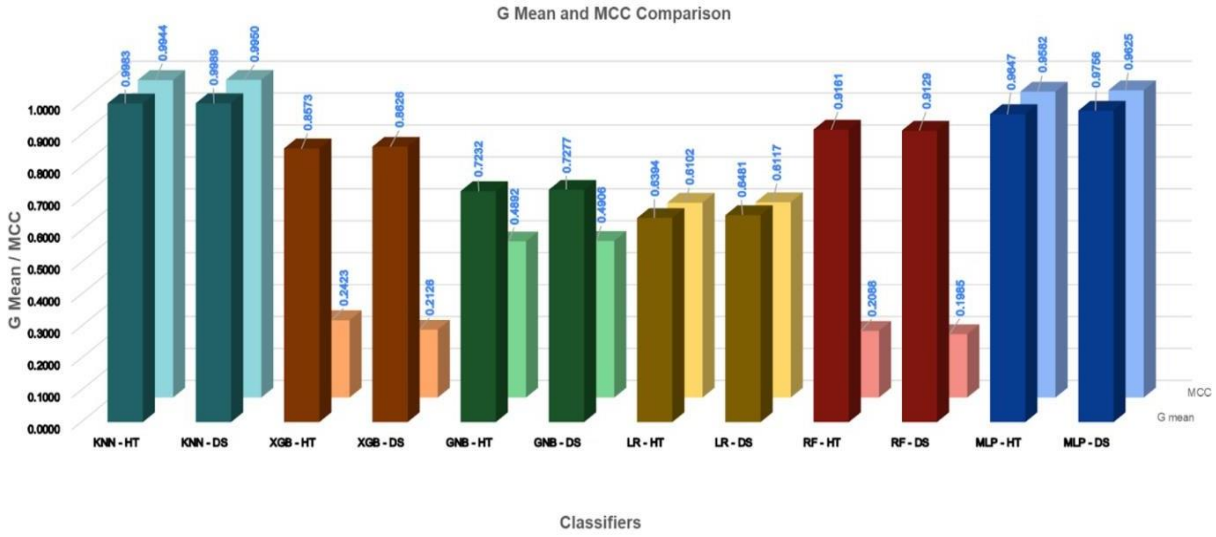

Figure 5.3: Comparison of G-Mean and MCC score

5.2 Comparative Analysis

Table 5.5 shows the differences between this study and other anomaly detection studies that have been done previously. First two rows of the comparison table are based on the same dataset that has been used in this paper. The remaining three rows are based on earlier version of our used dataset (HAI 20.07). Bian et al. [38] presented the Stacked Gated Recurrent Unit-Infrequent Residual Analysis (SG-IRA) which is an unsupervised learning approach and 80.6% recall, 96.8% precision and 97.7% F1-score were achieved there with time-series awareness. In another study, Kim and Kim used Ensemble Recurrent Neural Networks with time-series awareness to achieve 90.2% recall, 97.7% precision and 93.8% F1-score consecutively [6]. Kim et al. used One Class Support Vector Machine (OCSVM) to detect the presence of anomaly where HAI 20.07 dataset was used, and their learning model approach demonstrated 95% recall, 96% precision, and 95% F1 score [5].

Table 5.5 Comparison with other Researches

Reference	*Version*	*Algorithm*	*Recall*	*Precision*	*F1-Score*
[38]	21.03	SG-IRA	80.6%	96.8%	97.7%
[27]	21.03	Ensemble RNN	90.2%	97.7%	93.8%
[19]	20.07	OCSVM	95%	96%	95%
[25]	20.07	CD-OCC	84%	84%	83%
[4]	20.07	N/A	N/A	90.8%	80.4%
This Study	21.03	KNN	99.77%	99.22%	99.50%

However, in another study, for HAI 20.07 dataset, Kim and Kim showed 84% recall, 84% precision and 83% F1 score using Clustered Deep One-Class Classification (CD-OCC) framework [3]. Moreover, Conti et al. got 90.8% precision and 80.4% F1-score in their study [4]. Tai et al. only showed accuracy for HAI 20.07 where maximum 83.63% accuracy was achieved using Gradient Boosting Machine GSCV (Grid Search Cross Validation) model [22]. Finally, the approach used in this study achieved the highest recall of 99.77%, highest precision of 99.22%, maximum F1 score of 99.50%, and best accuracy of 99.99% using the KNN algorithm. Hyperparameter tuning with the RandomSearchCV, as well as the use of downsampling and the SMOTE method, played a significant role in achieving comparatively higher performance across all results. Furthermore, the evaluation metrics G-mean and MCC had never been used in any of the previous studies. In our study, highest G-mean of 99.89% and MCC value of 0.9950, both also indicate satisfactory performance of our explored methodology

Chapter 6

Conclusion

Industrial control system (ICS) serves a significant role in implementing required functions along with enhancing safety. An industrial control system keeps the overall harmony of all the machines within an industry through collecting data from sensors. In this study, we have observed, analyzed and optimized the performance of six ML algorithms for anomaly detection in an HIL based Industrial Control System. Our study serves two purpose, one is the evaluation of the effectivity of ML in such an environment for failure proof system and to what extent it can be done, and the other is anomaly detection in the absence of any time related pattern as there might be insufficient timed data to begin with. Among the six classifiers, KNN outperforms all other classifiers in terms of all the 8 evaluation metrics under our consideration

Therefore, it is reasonable to infer that ML techniques have a beneficial influence on industrial control system sector with the introduction of an efficient anomaly detection system, which will ultimately strengthen the cyber resilience of its computer-controlled framework. Future work on our topic can be to study our proposed pipeline or workflow to evaluate the algorithms in a multiclass system. Also the evaluation of our proposed model in terms of other industrial control systems is also a topic of interest.

Future Development

There are some scopes of development of our work:

- ***Application with Different Datasets:*** More datasets can be used and thus the effectuality of our proposed approach can be understood more clearly
- ***Hybrid Models:*** No hybrid models have been used in this study. So there is a scope of developing a hybrid model that is more efficient
- ***Positions of anomaly:*** We have found out whether there is any anomaly or not. But the exact positions of anomaly have not been detected in this study. In future work, positions of anomaly and thus more exact information about anomalous data can be extracted.

References

[1] K. Paridari, N. O'Mahony, A. El-Din Mady, R. Chabukswar, M. Boubekeur, and H. Sandberg, "A framework for attack-resilient industrial control systems: Attack detection and controller reconfiguration," Proc. IEEE Inst. Electr. Electron. Eng., vol. 106, no. 1, pp. 113–128, 2018.

[2] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn, Guide to Industrial Control Systems (ICS) Security Supervisory Control and Data Acquisition (SCADA) systems Distributed Control Systems (DCS) and other control system configurations such as Programmable Logic Controllers (PLC) Special Publication 800-82. Gaithersburg, MD, 2015.

[3] B. Filkins, D. Wylie, and A. J. Dely, "Sans 2019 state of ot/ics cybersecurity survey," in SANSTM Institute, 2019.

[4] M. Conti, D. Donadel, and F. Turrin, "A survey on industrial control system testbeds and datasets for security research," IEEE Commun. Surv. Tutor., vol. 23, no. 4, pp. 2248–2294, 2021.

[5] S. Mokhtari, A. Abbaspour, K. K. Yen, and A. Sargolzaei, "A machine learning approach for anomaly detection in industrial control systems based on measurement data," Electronics (Basel), vol. 10, no. 4, p. 407, 2021.

[6] B. Miller and D. Rowe, "A survey SCADA of and critical infrastructure incidents," in Proceedings of the 1st Annual conference on Research in information technology - RIIT '12, 2012.

[7] T. Armerding, "Throwback Thursday: Whatever Happened to Stuxnet?," Synopsys, 2019.

[8] S. Shrivastava, "Blackenergy-malware for cyber-physical attacks," Singapore, vol. 74, 2016.

[9] A. D. Pinto, Y. Dragoni, A. Carcano, and ". Triton, The First ICS Cyber Attack on Safety Instrument Systems Understanding the Malware, Its Communications and Its OT Payload. Black Hat USA, 2018.

[10] Kaspersky.com. [Online]. Available: https://ics-cert.kaspersky.com/media/KASPERSKY_H1_2020_ICS_REPORT_EN.pdf. [Accessed: 16-Feb-2022].

[11] R. D. S. Raizada and Y.-S. Lee, “Smoothness without smoothing: why Gaussian naive Bayes is not naive for multi-subject searchlight studies,” PLoS One, vol. 8, no. 7, p. e69566, 2013.

[12] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, “KNN model-based approach in classification,” in On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE, Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 986–996.

[13] A. Narzullaev, Z. Muminov, and M. Narzullaev, “Wi-Fi based student attendance recording system using logistic regression classification algorithm,” in INTERNATIONAL UZBEKISTAN-MALAYSIA CONFERENCE ON “COMPUTATIONAL MODELS AND TECHNOLOGIES (CMT2020)”: CMT2020, 2021.

[14] P. Schober and T. R. Vetter, “Logistic regression in medical research,” Anesth. Analg., vol. 132, no. 2, pp. 365–366, 2021.

[15] A. Sarica, A. Cerasa, and A. Quattrone, “Random forest algorithm for the classification of neuroimaging data in Alzheimer’s disease: A systematic review,” Front. Aging Neurosci., vol. 9, 2017.

[16] “eXtreme Gradient Boosting (XGBoost): Better than random forest or gradient boosting,” Github.io. [Online]. Available: https://liuyanguu.github.io/post/2018/07/09/extreme-gradient-boosting-xgboost-better-than-random-forest-or-gradient-boosting/. [Accessed: 14-Feb-2022].

[17] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system,” in Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, 2016, pp. 785–794.

[18] S. Abirami and P. Chitra, “Energy-efficient edge based real-time healthcare support system,” in Advances in Computers, Elsevier, 2020, pp. 339–368.

[19] J. Kim, H. Choi, J. Shin, and J. T. Seo, “Study on anomaly detection technique in an industrial control system based on machine learning,” in Proceedings of the 2020 ACM International Conference on Intelligent Computing and its Emerging Applications, 2020

[20] H. Shin, W. Lee, J. Yun, and H. Kim, “HAI 1.0: HIL-based Augmented ICS Security Dataset,” in 13 USENIX Workshop on Cyber Security Experimentation and Test, 2020.

[21] W. Choi, K. Joo, H. J. Jo, M. C. Park, and D. H. Lee, “Voltageids: Low-level communication characteristics for automotive intrusion detection system,” IEEE Trans. Inf. Forensics Secur, vol. 13, pp. 2114–2129, 2018.

[22] J. Tai, I. Alsmadi, Y. Zhang, and F. Qiao, “Machine Learning Methods for Anomaly Detection in Industrial Control Systems,” in 2020 IEEE International Conference on Big Data, IEEE, 2020, pp. 2333–2339.

[23] S. Zhong, S. Fu, L. Lin, X. Fu, Z. Cui, and R. Wang, “A novel unsupervised anomaly detection for gas turbine using isolation forest,” in IEEE International Conference on Prognostics and Health Management (ICPHM), San Francisco, CA, USA, 2019, pp. 1–6.

[24] S. Ahmed, L. Youngdoo, and I. Seung-Ho Hyun, Unsupervised Machine Learning-Based Detection of Covert Data Integrity Assault in Smart Grid Networks Utilizing Isolation Forest”. IEEE, 2019.

[25] Y. K. Younghwan Kim and H. K. K. Younghwan Kim, “Cluster-based deep one-class classification model for anomaly detection,” J. Internet Technol., vol. 22, no. 4, pp. 903–911, 2021.

[26] J. Kim, H. Choi, J. Shin, and J. T. Seo, “Study on anomaly detection technique in an industrial control system based on machine learning,” in Proceedings of the 2020 ACM International Conference on Intelligent Computing and its Emerging Applications, 2020.

[27] H. Kim and Y.-M. Kim, “Abnormal Detection for Industrial Control Systems Using Ensemble Recurrent Neural Networks Model,” Journal of the Korea Institute of Information Security & Cryptology, vol. 31, no. 3, pp. 401–410, 2021.

[28] H. K. Shin, W. Lee, J. H. Yun, and H. Kim, “Implementation of programmable CPS testbed for anomaly detection,” in 12th USENIX Workshop on Cyber Security Experimentation and Test, 2019.

[29] W. S. Hwang, J. H. Yun, J. Kim, and H. C. Kim, “Time-series aware precision and recall for anomaly detection: considering variety of detection result and addressing ambiguous labeling,” in Proceedings of the 28th ACM International Conference, 2019.

[30] D. Chicco and G. Jurman, “The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation,” BMC Genomics, vol. 21, no. 1, p. 6, 2020.

[31] U. Bhowan, M. Johnston, and M. Zhang, “Evolving ensembles in multi-objective genetic programming for classification with unbalanced data,” in Proceedings of the 13th annual conference on Genetic and evolutionary computation - GECCO ’11, 2011.
[32] A. P. Bradley, R. P. W. Duin, P. Paclik, and T. C. W. Landgrebe, “Precision-recall operating characteristic (P-ROC) curves in imprecise environments,” in 18th International Conference on Pattern Recognition, 2006.
[33] Z. C. Lipton, C. Elkan, and B. Narayanaswamy, “Thresholding classifiers to maximize F1 score,” arXiv [stat.ML], 2014.
[34] Y. Baştanlar and M. Özuysal, Introduction to machine learning. miRNomics: MicroRNA biology and computational analysis. 2014.
[35] A. J. Bowers and X. Zhou, “Receiver operating characteristic (ROC) area under the curve (AUC): A diagnostic measure for evaluating the accuracy of predictors of education outcomes,” J. Educ. Stud. Placed Risk, vol. 24, no. 1, pp. 20–46, 2019.
[36] M. Kubat and S. Matwin, Addressing the curse of imbalanced training sets: one-sided selection. 1997.
[37] Y. Liu, J. Cheng, C. Yan, X. Wu, and F. Chen, “Research on the Matthews correlation coefficients metrics of personalized recommendation algorithm evaluation,” Int. J. Hybrid Inf. Technol., vol. 8, no. 1, pp. 163–172, 2015.
[38] X. Bian, Detecting Anomalies in Time-Series Data using Unsupervised Learning and Analysis on Infrequent Signatures. 202