Machine Learning for Reliable Network Attack Detection in SCADA Systems

Rocio Lopez Perez*, Florian Adamsky†, Ridha Soua†, and Thomas Engel†
* rocio.lopez.001@student.uni.lu, CSC, University of Luxembourg, Luxembourg
† {name.surname}@uni.lu, SnT, University of Luxembourg, Luxembourg

***Abstract*—Critical Infrastructures (CIs) use Supervisory Control And Data Acquisition (SCADA) systems for remote control and monitoring. For a long time, operator of CIs applied the *air gap* principle, a security strategy that physically isolates the control network from other communication channels. True isolation, however, is difficult nowadays due to the massive spread of connectivity: using open protocols and more connectivity opens new network attacks against CIs. To cope with this dilemma, sophisticated security measures are needed to address malicious intrusions, which are steadily increasing in number and variety. Traditional Intrusion Detection Systems (IDSs) cannot detect attacks that are not already present in their databases. In this paper, we assess Machine Learning (ML) for intrusion detection in SCADA systems using a real data set collected from a gas pipeline system and provided by the Mississippi State University (MSU). The contribution of this paper is two-fold: 1) The evaluation of four techniques for missing data estimation and two techniques for data normalization, 2) The performances of Support Vector Machine (SVM), Random Forest (RF), Bidirectional Long Short Term Memory (BLSTM) are assessed in terms of accuracy, precision, recall and F_1 score for intrusion detection. Two cases are differentiated: binary and categorical classifications. Our experiments reveal that RF and BLSTM detect intrusions effectively, with an F_1 score of respectively $> 99\%$ and $> 96\%$.**

I. Introduction and Problem Statement

Supervisory Control And Data Acquisition (SCADA) systems are commonly used by Critical Infrastructures (CIs) or industries which are vital to citizens' daily lives and countries' economies. It includes oil pipelines, water treatment, and chemical manufacturing plants to name but a few. Typically, SCADA systems consist of (1) field instrument devices for sensing conditions of the CI (power level, pressure, throughput, etc.); (2) operating equipment such as valves, pumps, etc. controlled by actuators; (3) field local processors such as Programmable Logic Controllers (PLCs) and Remote Terminal Units (RTUs) that communicate with field instrument devices and operating equipment; and finally (4) the Human Machine Interface (HMI) that acts as a central controller and monitoring host. To operate properly in a synchronized manner, these different components must communicate. While short-range communications are used to establish links between local processors, instrument devices and operating equipments, long-range communications are used to connect PLCs and RTUs with the HMI or the Master Terminal Unit (MTU).

Historically, SCADA systems implemented a security principle known as *air gap*, a strategy that physically isolates the control network from the rest of the network, including the Internet. True isolation, however, is difficult in a real-world environment. First, true isolation may lead to outdated software [1], [2]. Without connectivity to the Internet, the software cannot easily receive security updates from the vendor. Second, true isolation is hard to implement since CI is often geographically distributed. To avoid the high costs of laying direct fiber cable to substations, CI operators make use of radio, Asymmetric Digital Subscriber Line (ADSL), General Packet Radio Service (GPRS), or leased lines. Moreover, malware like Stuxnet [3] or Flame [4] has shown us that even a USB flash drive can provide connectivity to the outside world. Besides the air gap principle, SCADA systems have made use of proprietary software, hardware, and communication protocols which have provided a false sense of security through obscurity [1].

Nowadays, the use of standardized communications protocols has enabled the integration of SCADA systems with the Internet and corporate networks. Given this new context, SCADA systems are prone to numerous threats due to their large deployment areas, distributed operating mode and growing interconnectivity [5]. Indeed, the widespread use of the TCP/IP stack has led to the its adoption in SCADA systems. Modicom Communication Bus (Modbus) TCP, Distributed Network Protocol (DNP3) [6], and IEC 60870-5-104 are the main communication protocols used. These protocols were designed over twenty years ago and are known to be highly vulnerable to simple network attacks [7]–[10]. Mirian et al. [11], using Internet-wide scanners such as ZMap [12], identified 60,000 vulnerable SCADA devices connected to the Internet. Clearly, these protocols stacks are subject to increasing risks. This can also be seen in the cyberattacks against the Ukrainian power grid in 2015, were 225,000 Ukrainian people were without electricity. These attack were the first that resulted in a power outage [13].

Our contributions: In this paper, we focus on assessing the performances of Machine Learning (ML) techniques such as Support Vector Machine (SVM), Random Forest (RF), and Bidirectional Long Short Term Memory (BLSTM) in detecting intrusion in SCADA systems. Section II lays out the foundation of the SCADA architecture and the ML algorithms used. We analyze SCADA protocols from monthly Internet-wide scans and see an increasing number of SCADA services reachable and attackable over the Internet. Section III describes the data set and the experimental setup in detail. In Section IV, we analyze four missing data strategies and

two data normalization techniques, characterizing the performances of the ML algorithms in terms of accuracy, precision, recall and F_1 score for binary and categorical classification. We describe related works in Section V and compare them with our approach. Finally, we conclude in Section VI and give directions for future research.

II. Technical Background

In this section, we provide a brief overview of the SCADA architecture, its network protocols, and the ML algorithms that we have used in this work. While discussing the technical background, we also highlight the vulnerabilities that exist in SCADA protocols.

A. Attack Vectors on SCADA

As described in Section I, adversaries often can reach the control system from the Internet, because the air gap principle is no longer not applicable in modern SCADA networks [1]. Most of these networks are geographically distributed. Hence, they need to be connected to the HMI, either via ADSL, GPRS, or leased lines. All of these connections can be used to gain access to the control system.

After an attacker has gained access to the network, there are three attack vectors against a SCADA protocol: First, by exploiting vendor-specific implementation faults like memory-corruption bugs; second, by exploiting weaknesses in the infrastructure like missing or inadequate firewall rules; and third, by exploiting protocol-specific weaknesses in the specification. In this paper, we focus on the third attack vector. An attacker wanting to exploit SCADA protocol weaknesses, has four general attacks to choose from [7], as shown in Figure 1:

1) Interception: An attacker is able to analyse the network traffic and gather information about the network infrastructure;
2) Interruption: An attacker intercepts packets and does not forward them to the next node;
3) Modification: The attacker is a man-in-the-middle (MitM) modifying packets in a network stream;
4) Fabrication: An attacker is able to inject packets into the network.

Figure 1 depicts a simplified SCADA architecture in which an attacker (red square) has gained access to the network. All four attacks can target the HMI (a), the network infrastructure itself (b), or the RTU/PLC (c). The field devices shown in Figure 1 are sensors and actuators. A sensor monitors the environment, e.g. the pressure of a gas pipeline, and sends the information to the next higher level; an actuator, in contrast, receives commands to control the environment, e.g. opening and closing a valve. The RTU or PLC controls and monitors the field devices, building a substation. One advantage of the SCADA architecture is that substations can be geographically distributed; this is often a necessity for a CI. The control centre is located in a different physical location and contains the HMI which monitors and controls the RTUs and PLCs. The RTUs/PLCs are connected to the HMI via communication links such as radio, fibre-optics, or dial-up lines. All information converges to the HMI or SCADA master, which is monitored and controlled by an employee.

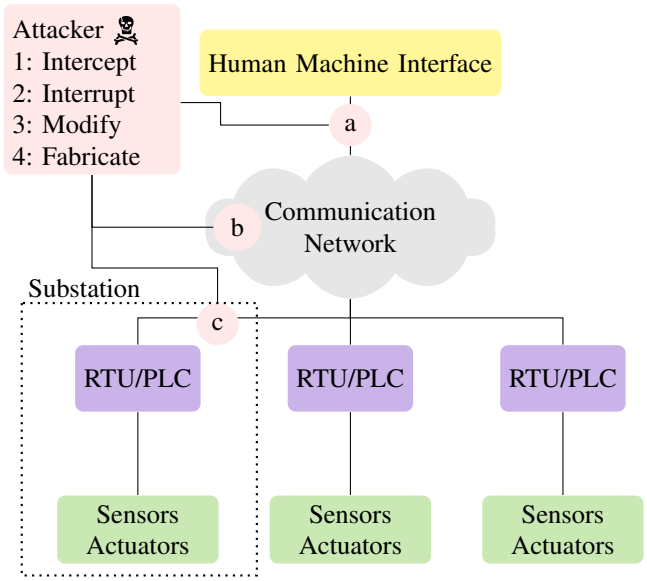

Fig. 1. Attack model demonstrating four network attacks, denoted as (1–4), against a simplified SCADA architecture with three attack targets (a–c) based on [7].

III. Anomaly Detection in SCADA Systems: Data Set and Methodology

To investigate the merits of the ML-based techniques for anomaly detection in SCADA systems, a real-world gas pipeline data set is used for anomaly detection in our experiments. We now describe the data set in detail, as well as the different steps of our methodology for anomaly detection.

A. The Gas Pipeline Data Set

The SCADA data set used in this work is hosted on the Industrial Control System (ICS) Cyber Attack Data Sets [26] website. The real-world raw data was generated using a gas pipeline system provided by the Mississippi State University (MSU)'s in-house SCADA lab. It contains a total of 274,628 instances.

The methodology for the data set collection is described in the study carried out by Turnipseed [27]. The data set, present in the Attribute-Relation File Format (ARFF), is used to create ML models once it has been pre-processed. It contains 20 features from Modbus RTU packets, three different types of labels and also pure raw data, which is provided to aid in the pre-processing stage. Table I lists the features and their corresponding types.

The address feature is a unique eight-bit value used for device identification. It is assigned to each master and slave device allowing them to recognize each other while establishing a communication. This feature is used to overcome scan attacks which broadcast commands to all possible station addresses to determine which addresses are in use. The second feature is the function code. Some function codes can be used

TABLE I
LIST OF FEATURES FROM THE GAS PIPELINE DATA SET.

Nr.	Features	Types
1	Address	Network
2	Function	Command Payload
3	Length	Network
4	Setpoint	Command Payload
5	Gain	Command Payload
6	Reset Rate	Command Payload
7	Deadband	Command Payload
8	Cycle Time	Command Payload
9	Rate	Command Payload
10	System Mode	Command Payload
11	Control Scheme	Command Payload
12	Pump	Command Payload
13	Solenoid	Command Payload
14	Pressure Measurement	Response Payload
15	CRC rate	Network
16	Command Response	Network
17	Time	Network
18	Binary Result	Label
19	Categorized Result	Label
20	Specific Result	Label

for malicious purposes (DoS attack), such as '0x08', which can be used to force a slave device to stay in listening mode. The length field gives the Modbus frame length. This feature may help detecting attacks by identifying frames which are not of an ordinary length. The set point feature is the most critical, since it controls the pressure in the gas pipeline when the system is in automatic mode.

Other features such as gain, reset rate, dead band, cycle time, and rate allow the PID controller to open and/or close the gas valve as well as turn on and/or turn off the pump, based on a calculated error value. The system mode, which represents how the system is operating, may have three possible values: (1) off or inactive, (2) manual configuration or (3) automatic configuration. The control scheme feature determines whether the gas pipeline system will be controlled by the pump or by the solenoid. The pump field controls the pump state when the system mode is set to manual. An adversary were able to change the gas pipeline system mode to manual and turn the pump on, the system would become over-pressurized.
The pressure measurement feature provides the gas pressure measurement value provided by a pressure gauge attached to the pipeline. An attacker could use this feature to provide false measurements emulating fabricated behaviours in the system. An adversary may perform an attack by constantly transmitting a bad Cyclic Redundancy Check (CRC) to cause a Denial-of-Service (DoS) attack. The command response feature, as its name indicates, helps the Intrusion Detection System (IDS) to differentiate between commands and requests. This feature, along with the timestamp, the binary result, the categorized result and the specific result features were not parsed from the Modbus RTU frame itself, but from Modbus TCP/IP traffic.

As discussed in Section I, SCADA systems are a focus of attention for cyber-attacks. The MSU's in-house SCADA lab used seven categories of attacks which were previously developed in Gao's research [28] to provide a broader perspective on the attacks that SCADA systems may suffer.

TABLE II
DESCRIPTION, CATEGORY AND TYPE OF THE ATTACKS.

Description	Category	Attack Type	#
Naive Response Injection	Response Injection	Modify/Fabricate	7,753
Complex Response Injection	Response Injection	Modify/Fabricate	13,035
State Command Injection	Command Injection	Modify/Fabricate	7,900
Parameter Command Injection	Command Injection	Modify/Fabricate	20,412
Function Code Injection	Command Injection	Modify/Fabricate	4,898
Denial of Service	Denial of Service	Interrupt	2,176
Reconnaissance	Reconnaissance	Intercept	3,874

These attacks, set out in the Table II, are the result of one or a series of external malicious activities through Modbus RTU packets. The attacks in Table II include a description of the attacks, a category and a attack type according to our attack model in Figure 1.

B. Methodology

Developing an ML-based IDS for intrusion detection in SCADA systems requires the steps illustrated in Figure 4. In some cases attribute values ("features") were missing from the data set used in our experiments. As these values are useful in prediction modelling, the first phase of our approach cleans and transforms the data to eliminate incomplete records. Next, to train our data, it was fundamental to follow the *Holdout* method and split each of the sixteen data sets into training, validation and test sets containing respectively 60% (164,776 instances), 20% (54,926 instances) and 20% (54,926 instances) of the observations. The validation set and the test set were respectively pre-processed based on the statistics obtained from the training set and the combined training set and validation set. Because parameters of prior distribution, called hyperparameters, may significantly impact the performance of ML methods, we performed a hyperparameter search for the selected ML algorithms. Given that the data set is comprised of normal traffic and variants of attack types, we distinguish two classifiers: binary classification (normal, anomaly) and seven-category classification (see attacks depicted in Table II).

1) Data Cleaning: We observed that many feature values were missing or non-existent. The Table III depicts the first three rows of the data set in ARFF. Addr, funct and c/r refer respectively to the address, function and command response features.

Table III, presents three different types of payloads where data is missing: 1) All values are missing or nonexistent; 2) only the pressure measurement is present; and 3) all values except the pressure measurement are present. To handle the feature values in the data set that do not have any representation or meaning, we used four techniques:

Gaussian Mixture Model (GMM) can find the best $number$ k of Gaussian distributions needed to cluster our data. To this end, the algorithm finds the best mean or centre, μ and variance σ of the Gaussian distributions that best separate our data.

K-means allows us to find the best number k of clusters by computing the Euclidean distance between the given

TABLE III
EXAMPLES OF THE MISSING VALUES IN THE GAS PIPELINE DATA SET.

Address	Function	Length	Payload	CRC	C/R	Timestamp
4	3	16	?,?,?,?,?,?,?,?,?,?	12869	1	1418682163.170388
4	3	46	?,?,?,?,?,?,?,?,?,?,0.689655	12356	0	1418682163.269946
4	16	90	10,115,0.2,0.5,1,0,0,1,0,0,?	17219	1	1418682164.995590

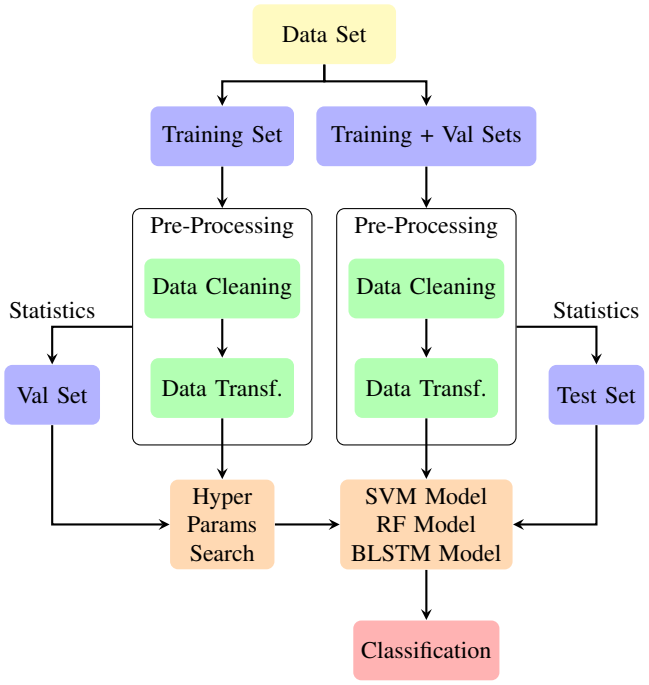

Fig. 2. Flow chart diagram illustrating the steps of our work pipeline.

samples and a pre-assigned centroid point, assigning them to a certain cluster and updating the centroids of the clusters until convergence on the best separation of the data.

In both GMM and K-means techniques, the first payload type were considered as cluster $k = 0$, and the second and third payload types were be assigned to k number of clusters defined by the elbow method. This method determined the best number of clusters based on the *cost function* or *distortion*:

$$\jmath = \sum_{k=1}^{K} \sum_{i \in C_k} ||x_i - \mu_k||^2. \quad (1)$$

Lower values of $\jmath$ determine a preferable number k of clusters and thus, better data separation. With this strategy, payloads are classified into k clusters, which are represented in the pre-processed data as a *one-hot encoded* notation. One hot encoding is a process of converting categorical variables into form more suitable for ML algorithms.

Zero imputation & indicators is a technique in which we substituted missing values with 0 and indicated their positions by adding corresponding indicators with 1 values to the payload feature. If the feature value existed in the payload then the value was kept and the indicator set to 0 [29].

Keep prior value, also known as forward-filling, deals with the non-existent values by replacing them with the immediately preceding existing feature value. In the case where forward-filling is not possible due to a lack of existing prior feature values, backward-filling is conducted. The intuition behind this technique is that the missing values are not dues to data loss but simply cannot exist, since the type of the packet does not support these features. Therefore, they appear in the data as non-existent values and they may be inferred from previously seen feature values.

2) Data Transformation: This step was conducted by performing, first, the *mean-standard deviation* and then *min-max* methods. The mean-standard deviation method consists of subtracting the calculated overall mean and dividing by the calculated overall standard deviation for each of the values within a certain feature. Thus,

$$z_i = \frac{x_i - \mu}{\sigma}, \quad (2)$$

where x is a feature value, μ is the mean, and σ is the standard deviation. Performing this pre-processing strategy ensures the minimization of the sample deviations from the mean. The second method is min-max approach, which consists of finding the minimum and maximum value from a given feature and normalizing the feature values between 0 and 1. Hence,

$$z_i = \frac{x_i - min(x)}{max(x) - min(x)}, \quad (3)$$

where x_i is a feature value, $min(x)$ and $max(x)$ are the minimum and maximum values calculated from the overall feature values.

3) Hyperparameter Search: In a SVM, the hyperparameters C and γ must be correctly set for each of the sixteen data sets. Hence, we performed a random search to determine the best hyperparameters for our models. Although grid search and manual search are the most widely used techniques for hyperparameter optimization, it has been empirically and theoretically demonstrated that randomly chosen tests are more efficient [30].

For each of the sixteen pre-processed data sets, we ran thirty different prediction trials over the corresponding validation set, during the hyperparameter search. The seven most notable results are analyzed to investigate how the algorithms converge to a good result after the best hyperparameters are found. Due

to the long training time of SVMs, we used only 25% from the entire data set.

In RF, the hyperparameters *number of estimators* and *maximum depth of the trees* must be correctly set for each of the sixteen data sets. Once again, we performed a random search, through thirty different prediction trials, to define the best hyperparameters for these models.

In BLSTM, the hyperparameters *learning rate*, *batch size*, *sequence length*, *dropout* and *hidden layer size* must be correctly set for each of the sixteen data sets. Again, we conducted a random search, by running through fifty epochs, a parameter for BLSTM, to define the best hyperparameters for these models. For each data set, we ran thirty different predictions over the corresponding validation set during the hyperparameter search. The seven most significant results are used in this study to show how the algorithm converges once the best hyperparameters are found.

4) Classification: In this step, models are created with the aim of classifying novel observations on a set of predefined classes. If only two possible classes exist, then it is called binary classification. In contrast, if more than two classes are differentiated, it is called multi-class classification. In the context of this work, a classification task is performed to correctly classify benign and malicious packets. The trained model output would be 0 or 1, for a binary classification approach and from 0 to n classes', for a multi-class classification approach.

IV. Detecting Intrusion in SCADA: Experiment and Analysis of Results

We developed our classification scripts with Scikit-learn[1], TensorFlow[2] and Keras[3]. Our source code is available on GitHub [31]. In the following, we evaluate our test results, together with the performance results of SVM, RF and BLSTM for anomaly detection using the gas pipeline system data set.

A. Anomaly Detection Results

We split each of the sixteen data sets into training, validation and test sets according to the division in Section III-B. Once we obtain the best configuration for a given classifier, the validation set is combined with the training set, leaving the final split into 80% of the observations in training set and 20% in the testing set. Two classifiers were used to study the performance of the different SVM, RF and BLSTM-based IDS models: binary (normal, anomaly) and categorical (see attacks listed in Table II). We denote these respectively by "BIN" and "CAT". For each experiment, we compared the performance of each ML technique under mean-standard deviation (MEAN) and min-max (MIN-MAX) approaches.

1) SVM Performance: Figures 5a, 5d, 5g, and 5j show the performance of SVMs for the binary and categorical classifier and for the MEAN and MIN-MAX data normalization. As we can see, the BIN classifier achieves a better F_1 score in both cases of data normalization (MEAN and MIN-MAX) using the *Keep prior value*. Indeed, the lowest F_1 score for binary classification is 92.04% (see Figure 5g) while for CAT classification this value drops to 88.45 % (see Figure 5(j)). The worst performance in terms of F_1 score for both classifiers was obtained by GMM and K-means algorithms. The Zeros & Indicators method performs better that GMM but worse than Keep prior value. For both binary and categorical classifiers, the MEAN normalization strategy outperforms MIN-MAX normalization. Table IV summarizes the results, highlighting the best for BIN and CAT SVM classifiers employing the split criterion of 80% for the training set and 20% for the test set, and using the hyperparameters that gave us the best performance. We obtained a F_1 score of 94.34% for BIN and a F_1 score of 92.50% for the CAT classifier. These were achieved using MEAN normalization and *keep the prior existing value* strategy respectively to deal with missing values.

TABLE IV
Best binary and categorical classifiers modeled with SVM.

SVM	Hyper-parameters		Measurements			
Test sets	C	gamma	Acc	Prec	Recall	F1-score
binary-mean-keep	**346.219**	**0.3975**	**94.36 %**	**94.33 %**	**94.36 %**	**94.34 %**
binary-minmax-keep	579.161	0.6270	92.78 %	92.91 %	92.78 %	92.83 %
categorical-mean-keep	**107.411**	**0.2689**	**92.56 %**	**92.47 %**	**92.56 %**	**92.50 %**
categorical-minmax-keep	536.672	0.7150	89.70 %	90.50 %	89.70 %	89.97 %

2) RF Performance: Figures 5b, 5e, 5h, and 5k present the contrasting configurations in a binary and categorical classifier modelled with the RF algorithm. The highest F_1 score was achieved by the binary classifier: 99.40% with MIN-MAX technique for data normalization and using the *Keep prior value* approach for dealing with missing data.

Table V depicts the final results obtained using the best hyperparameters, and the 80%–20% split criterion, for the training and test sets. We obtained a F_1 score of 99.58% for BIN and a F_1 score of 99.41% for CAT. It is worth mentioning that for the final results, the difference between MEAN and MIN-MAX normalization strategies is very small: as illustrated in Table V, the difference is 0.02% for binary classification and 0.03% for categorical classification. Therefore, similar results can be achieved with both normalization strategies.

TABLE V
Best binary and categorical classifiers modelled with Random Forest Algorithm; ne and md correspond to number of estimators and maximum depth.

Random Forest	Hyper-parameters		Measurements			
Test sets	ne	md	Acc	Prec	Recall	F1-score
binary-mean-keep	**47**	**49**	**99.58 %**	**99.58 %**	**99.58 %**	**99.58 %**
binary-minmax-keep	44	71	99.56 %	99.57 %	99.56 %	99.56 %
categorical-mean-keep	**71**	**80**	**99.41 %**	**99.41 %**	**99.41 %**	**99.41 %**
categorical-minmax-keep	64	88	99.39 %	99.39 %	99.39 %	99.38 %

3) BLSTM Performance: In Figures 5c, 5f, 5i, and 5l, which show the results for BLSTM, the *Zeros imputation & indicators* strategy for dealing with missing values outperforms other techniques, such as K-means and GMM, and slightly outperforms the Keep prior value approach. This is consistent with the theory and experiments presented in

[1]http://scikit-learn.org/
[2]https://www.tensorflow.org/
[3]https://keras.io/

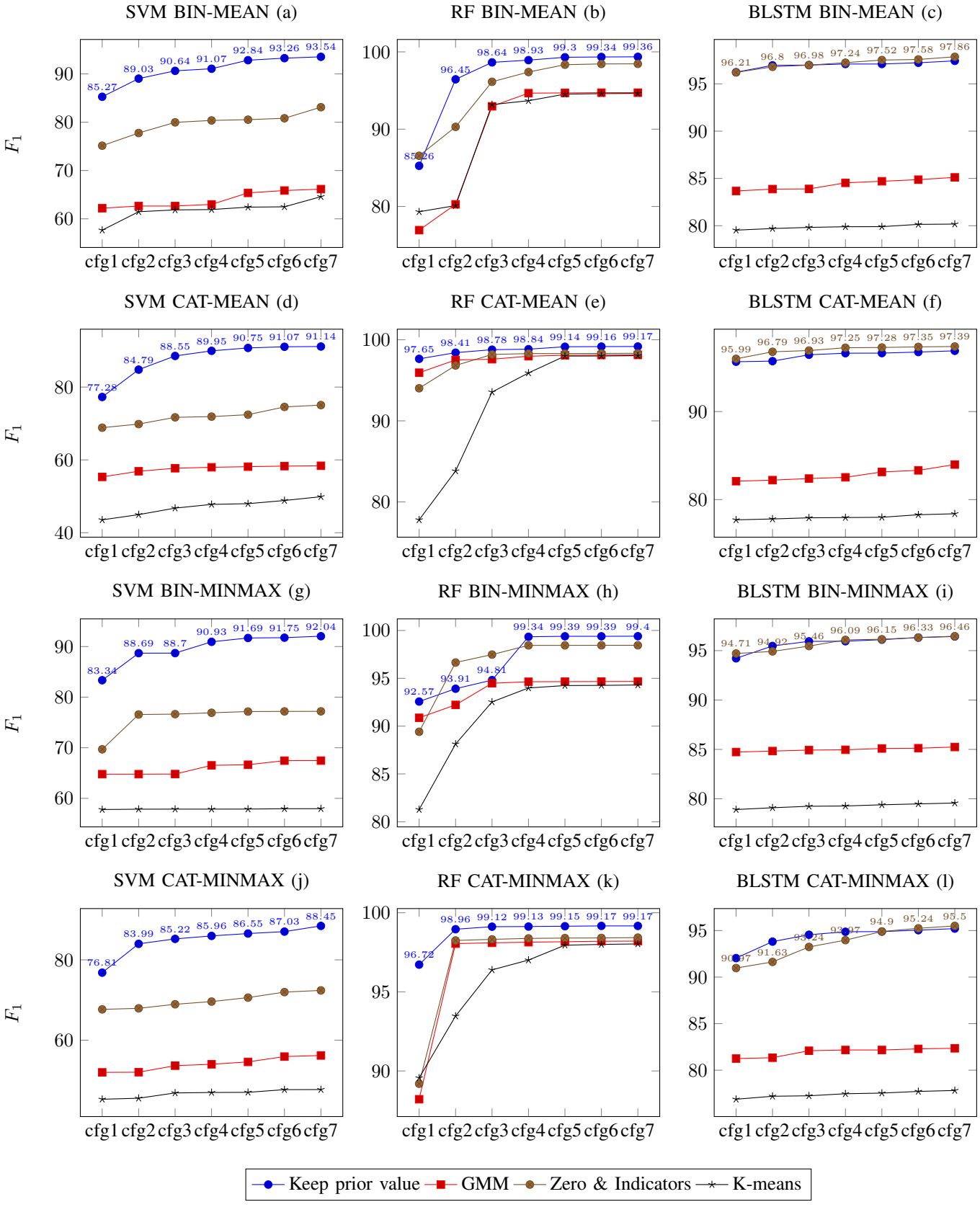

Fig. 3. Results for the hyperparameter search for SVM, RF, and BLSTM. The first row shows the results for the binary classification (BIN) using the mean-standard deviation normalization strategy (MEAN), the second row for the categorical classification (CAT) using MEAN. The third row shows the results for BIN using the min-max normalization strategy (MIN-MAX) and finally the fourth row for CAT using MIX-MAX. On the x axis are the different configurations for the hyperparameter depicted and on the y axis is the F_1 score depicted.

[29]. The Table VI summarizes the results for BIN and CAT BLSTM classifiers, running three hundred epochs with the best hyperparameters and using the 80%–20% split criterion. Bidirectional Long Short Term Memory outperforms SVM. We obtained a F_1 score of 98.39% for BIN and a F_1 score of 97.68% for CAT. As shown in Table VI, for both binary and categorical classifiers, the MEAN is better than MIN-MAX. The difference between these two normalization strategies is 0.77% for BIN and 1.2% for CAT classification.

TABLE VI
BEST BINARY AND CATEGORICAL CLASSIFIERS MODELLED WITH BLSTM ALGORITHM; LR, BATCH, SEQ, DROP AND H_LAYER CORRESPOND TO LEARNING RATE, BATCH SIZE, SEQUENCE LENGTH, DROPOUT AND HIDDEN LAYER SIZE.

BLSTM	Hyper-parameters					Measurements			
Test sets	lr	batch	seq	drop	h_layer	Acc	Prec	Recall	F1-score
binary-mean-indi	**0.008308**	**67**	**4**	**0.019025**	**110**	**98.40 %**	**98.40 %**	**98.40 %**	**98.39 %**
binary-minmax-indi	0.011490	121	4	0.027915	218	97.64 %	97.64 %	97.65 %	97.62 %
categorical-mean-indi	**0.009908**	**138**	**4**	**0.032404**	**136**	**97.71 %**	**97.69 %**	**97.71 %**	**97.68 %**
categorical-minmax-indi	0.013236	138	4	0.039841	254	96.57 %	96.53 %	96.57 %	96.48 %

4) Results Analysis: Although BLSTM models are widely used for time-dependent problems given their capabilities of using forward and backward information, RF results outperform those achieved with BLSTM algorithm, as illustrated in Table V & VI. This may be due to both a lack of collective attacks, and the existence of high randomness in the occurrence of attacks within the data set. Since the data set was generated, the developers made sure to avoid the appearance of unintended patterns and did not inject collective attacks. For instance, DoS could be performed as a set of packets that overwhelm the system, of which one single packet may not mean anything to the predictor. Taken together, however, they do matter and represent an attack. In our case, DoS attacks are performed by sending Modbus packets with incorrect CRC values. We emphasize that the data was generated, whereas in reality collective or sequential attacks may appear. This is why it is interesting to study the BLSTM algorithm and integrate it into a Network Intrusion Detection System (NIDS).

The results from RF, which are listed in Table VIII show that it correctly classifies large numbers of normal and malicious packets. The categorical classification report in Table VII shows the detection rate for each of the data type. The distinction between Complex Malicious Response Injection (CMRI) and Naive Malicious Response Injection (NMRI) presents low recall value. This is due to the randomness of NMRI attacks, which are likely to overlap in values with the CMRI attacks and normal data: since a CMRI attack consists of designing malicious packets that imitate normal behaviours, some of these overlap with normal packets. For a DoS attack, the cause for the low detection rate, in comparison with the rest of attacks, is due to the bad CRC attack. This attack injects an invalid CRC value in a write multiple register command, which makes the RTU to disregard the command, in turn causing a DoS. Random Forest algorithm was able to accurately classify the write command with the incorrect CRC value as an attack, but some responses from the RTU were not classified as a DoS attack.

TABLE VII
CLASSIFICATION REPORT OF THE RF ALGORITHM.

Random Forest	**Accuracy test data = 99.41 %**			
Type of Data	**precision**	**recall**	**f1-score**	**support**
Normal	99.48 %	99.90 %	99.69 %	42953
NMRI	98.14 %	96.99 %	97.56 %	1526
CMRI	98.84 %	96.40 %	97.60 %	2641
MSCI	99.28 %	98.63 %	98.96 %	1538
MPCI	99.90 %	98.00 %	98.94 %	4101
MFCI	98.77 %	100 %	99.38 %	967
DoS	97.54 %	95.42 %	96.47 %	415
Recon	99.61 %	97.96 %	98.78 %	786
avg / total	**99.41 %**	**99.41 %**	**99.41 %**	**54927**

TABLE VIII
CONFUSION MATRIX OF THE RF ALGORITHM.

Normal	**NMRI**	**CMRI**	**MSCI**	**MPCI**	**MFCI**	**DoS**	**Recon**	
42908	12	9	9	4	0	8	3	**Normal**
25	**1480**	21	0	0	0	0	0	**NMRI**
79	16	**2546**	0	0	0	0	0	**CMRI**
20	0	0	**1517**	0	0	1	0	**MSCI**
79	0	0	2	**4019**	0	1	0	**MPCI**
0	0	0	0	0	**967**	0	0	**MFCI**
19	0	0	0	0	0	**396**	0	**DoS**
4	0	0	0	0	12	0	**770**	**Recon**

V. RELATED WORK

The SCADA systems were originally designed following the air gap principle and therefore without security measures in mind [1]. Nowadays, these systems are in the spotlight of network attacks, due to standardization and connectivity to the Internet [2], [35]. While using ML for predicting anomalies in networks has motivated many studies, little research has tackled the advantage of using ML in SCADA systems by using real data sets and a varied set of ML algorithms. In the literature, a large number of studies used the Knowledge Discovery and Data Mining (KDD) 99 data set to evaluate their solutions for intrusion detection [36]–[39]. However, this data set does not consider the specificities of SCADA architecture, communication protocols and traffic patterns. Moreover, it is seen by the research community as biased, outdated, and not relevant for modern network attacks detection. In the following, we detail different intrusion detection approaches for SCADA systems using real data sets.

The authors of [40] combine the signature-based and model-based approaches to design a rule-based IDS for SCADA networks. Their IDS overcomes the main disadvantage of signature-based systems, i.e only known attacks are detected using pre-established rules. In [41], authors presented a multi-algorithm model-based IDS. Models that represent the expected/acceptable system behaviour are created, and any behaviour that causes violations of these models is detected as an attack.

Both [42] and [43] presented an IDS that detects malicious network traffic in SCADA systems, based on One Class Support Vector Machine (OCSVM) technique. While authors of [42] use OCSVM to classify malicious observations by comparing them with benign ones, the study carried out in [43] aims at detecting intruders in SCADA networks by analysing

variables of the control devices. Two different approaches of one-class classification, the Support Vector Data Description (SVDD) and the Kernel Principle Component Analysis (KPCA), were proposed as well in [44]. L^p-norms are studied in Radial Basis Function (RBF) kernels for intrusion detection.

An IDS that detects SCADA attacks based on the network traffic behaviour was proposed in [45]. The IDS extracts the time correlation between different network packets and then monitors the system to determine if it is behaving normally or not. An alarm is raised when anomalies are detected.

Authors of [46] presented an IDS using Neural Network based Modelling (IDS-NNM) algorithm following the supervised learning approach. They adopted a specific window based attribute extraction approach to capture the time series nature of the network packet stream. More recently, a Recurrent Neural Network (RNN) with unidirectional Long Short Term Memory (LSTM) architecture was proposed in [47] to detect industrial control system anomalies.

VI. Conclusion and Future Work

Until not too long ago, the most common security strategy for SCADA systems was the air gap principle: an operator of SCADA networks segregated the control network from other networks. Hence, attackers could not access them. The attacker had to be physically close to the SCADA system to access the communication channel, inject malicious data or even interfere with the protocol. Nowadays, with growing demands for connectivity between the SCADA control network and the corporate network, novel network attacks have appeared as PLCs or RTUs devices are managed over IP communication protocols. This increased interconnectivity results in the de-isolation of SCADA systems, making them more vulnerable. Attackers no longer need to gain physical access to on-site circuits to perform a hostile action but instead, malicious network packets can reach the field devices from anywhere.

In this paper, we have shown that ML techniques can detect network attacks against SCADA systems. We used a SCADA data set provided by the MSUs's in-house SCADA lab. It was generated using a gas pipeline SCADA system hosted in their laboratory. We used SVM, RF, and BLSTM to implement diverse IDS classifiers. We provided a complete comparison between these algorithms along with the random hyper-parameter search results. We published our source code on GitHub [31] to help other researchers to verify, compare, and/or extend their studies. In contrast to the state-of-the-art studies, the use of the test set accuracy, precision, recall and F_1 score allowed us to assess their performance correctly and comprehensively. The RF algorithm gives the best performance by detecting 99.90% of benign data and 98.46% of attacks, with an overall detection rate (recall) of 99.58%.

Our approach can be applied to different SCADA environments, because SCADA is based on a well-defined architecture (see Section II). The used data set was generated in a real gas pipeline following a typical SCADA architecture. Although, the data set contains only Modbus RTU traffic, other SCADA protocols (e.g. DNP3 or IEC 60870-5-104) have similar messages to monitor (read) and control (write) sensors and actuators. In addition, these protocols can be the victim of attacks that we have highlighted in Figure 1.

An interesting future investigation would be the extraction of rules from RF algorithms to integrate them with signature-based NIDSs such as Snort.

Acknowledgment

This work was partially funded by ATENA H2020 EU Project (H2020-DS-2015-1 Project 700581). We thank Dominic Dunlop for his review and comments that greatly improved the manuscript.

References

[1] E. Byres, "The Air Gap: SCADA's Enduring Security Myth," *Communications of the ACM*, vol. 56, no. 8, pp. 29–31, Aug. 2013. [Online]. Available: http://doi.acm.org/10.1145/2492007.2492018

[2] C. S. Wright. (2011, September) SCADA: Air Gaps Do Not Exist. Accessed: 2017-12-04. [Online]. Available: http://infosecisland.com/blogview/16770-SCADA-Air-Gaps-Do-Not-Exist.html

[3] R. Langner, "Stuxnet: Dissecting a Cyberwarfare Weapon," *IEEE Security Privacy*, vol. 9, no. 3, pp. 49–51, May 2011.

[4] K. Zetter. (2012, May) Meet 'Flame,' The Massive Spy Malware Infiltrating Iranian Computers. Accessed: 2017-12-04. [Online]. Available: https://www.wired.com/2012/05/flame/

[5] V. M. Igure, S. A. Laughter, and R. D. Williams, "Security Issues in SCADA Networks," *Elsevier Computers & Security*, vol. 25, no. 7, pp. 498–506, 2006.

[6] "IEEE Standard for Electric Power Systems Communications-Distributed Network Protocol (DNP3)," *IEEE Std 1815-2012 (Revision of IEEE Std 1815-2010)*, pp. 1–821, Oct 2012.

[7] S. East, J. Butts, M. Papa, and S. Shenoi, "A Taxonomy of Attacks on the DNP3 Protocol," in *International Conference on Critical Infrastructure Protection*. Springer Berlin Heidelberg, 2009, pp. 67–81.

[8] N. R. Rodofile, K. Radke, and E. Foo, "Real-Time and Interactive Attacks on DNP3 Critical Infrastructure Using Scapy," in *Proceedings of the 13th Australasian Information Security Conference (AISC 2015)*, 2015, pp. 67–70.

[9] P. Huitsing, R. Chandia, M. Papa, and S. Shenoi, "Attack Taxonomies for the Modbus Protocols," *International Journal of Critical Infrastructure Protection*, vol. 1, pp. 37–44, 2008.

[10] P. Maynard, K. McLaughlin, and B. Haberler, "Towards Understanding Man-In-The-Middle Attacks on IEC 60870-5-104 SCADA Networks," in *Proceedings of the 2nd International Symposium for ICS & SCADA Cyber Security Research 2014 (ICS-CSR 2014)*, Sep. 2014. [Online]. Available: http://ewic.bcs.org/content/ConWebDoc/53228

[11] A. Mirian, Z. Ma, D. Adrian, M. Tischer, T. Chuenchujit, T. Yardley, R. Berthier, J. Mason, Z. Durumeric, A. J. Halderman, and M. Bailey, "An Internet-wide view of ICS devices," in *14th IEEE Privacy, Security, and Trust Conference (PST'16)*, 2016.

[12] Z. Durumeric, E. Wustrow, and J. A. Halderman, "ZMap: Fast Internet-wide Scanning and Its Security Applications," in *Proceedings of the 22Nd USENIX Conference on Security*. USENIX Association, 2013, pp. 605–620. [Online]. Available: http://dl.acm.org/citation.cfm?id=2534766.2534818

[13] R. M. Lee, M. J. Assante, and T. Conway, "TLP: White Analysis of the Cyber Attack on the Ukrainian Power Grid," E-ISAC, Tech. Rep., Mar. 2016.

[14] S. W. A.-H. Baddar, A. Merlo, and M. Migliardi, "Anomaly detection in computer networks: A state-of-the-art review." *JoWUA*, vol. 5, no. 4, pp. 29–64, 2014.

[15] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman, "A Search Engine Backed by Internet-Wide Scanning," in *Proceedings of the 22nd ACM Conference on Computer and Communications Security*, Oct. 2015.

[16] "ModBus Application Protocol Specification V1.1b 3," http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf, 2012, accessed: 2017-11-24.

[17] C. C. Aggarwal, *Data Mining: The Textbook*. Springer, 2015.

[18] C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep. 1995. [Online]. Available: https://doi.org/10.1023/A:1022627411411

[19] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[20] C. Olah. (2015, Aug) Understanding LSTM Networks. Accessed: 2017-12-04. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[21] A. Karpathy. (2015, May) The Unreasonable Effectiveness of Recurrent Neural Networks. Accessed: 2017-12-04. [Online]. Available: https://karpathy.github.io/2015/05/21/rnn-effectiveness/

[22] M. Schuster and K. K. Paliwal, "Bidirectional Recurrent Neural Networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[23] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional lstm networks for improved phoneme classification and recognition," in *Artificial Neural Networks: Formal Models and Their Applications – ICANN 2005*, W. Duch, J. Kacprzyk, E. Oja, and S. Zadrożny, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 799–804.

[24] S. Latif, M. Usman, and J. Q. R. Rana, "Abnormal heartbeat detection using recurrent neural networks," *arXiv preprint arXiv:1801.08322*, 2018.

[25] X. Zhang, W. Kou, E. I. Chang, H. Gao, Y. Fan, Y. Xu *et al.*, "Sleep stage classification based on multi-level feature learning and recurrent neural networks via wearable device," *arXiv preprint arXiv:1711.00629*, 2017.

[26] "Industrial Control System (ICS) Cyber Attack Datasets," https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets, accessed: 2017-12-04.

[27] I. Turnipseed, "A New Scada Dataset For Intrusion Detection Research," M. Sc., Mississippi State University, August 2015.

[28] T. Morris and W. Gao, "Industrial Control System Traffic Data Sets for Intrusion Detection Research," *Advances in Information and Communication Technology Critical Infrastructure Protection VIII*, pp. 65—78, 2014.

[29] Z. C. Lipton, D. C. Kale, and R. Wetzel, "Directly Modeling Missing Data in Sequences with RNNs: Improved Classification of Clinical Time Series," in *Proceedings of Machine Learning for Healthcare 2016*, 2016, pp. 253–270.

[30] J. Bergstra and Y. Bengio, "Random Search for Hyper-Parameter Optimization," *The Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[31] "Machine learning techniques for Intrusion Detection in SCADA Systems," https://github.com/Rocionightwater/ML-NIDS-for-SCADA.git.

[32] L. Talavera, "Dynamic Feature Selection in Incremental Hierarchical Clustering," in *Proceedings of the European Conference on Machine Learning*. Springer, 2000.

[40] Y. Yang, K. McLaughlin, T. Littler, S. Sezer, B. Pranggono, and H. Wang, "Intrusion detection system for iec 60870-5-104 based scada networks," in *Power and Energy Society General Meeting (PES), 2013 IEEE*. IEEE, 2013, pp. 1–5.

[33] F. J. Valverde-Albacete and C. Peláez-Moreno, "100% Classification Accuracy Considered Harmful: The Normalized Information Transfer Factor Explains the Accuracy Paradox," *PloS one*, vol. 9, no. 1, 2014.

[34] X. Zhu, *Knowledge Discovery and Data Mining: Challenges and Realities: Challenges and Realities*. Igi Global, 2007.

[35] B. Zhu and S. Sastry, "Scada-specific intrusion detection/prevention systems: a survey and taxonomy," in *Proceedings of the 1st Workshop on Secure Control Systems (SCS)*, vol. 11, 2010.

[36] A. George, "Anomaly Detection Based on Machine Learning: Dimensionality Reduction using PCA and Classification using SVM," *International Journal of Computer Applications*, vol. 47, no. 21, 2012.

[37] G. Wang, J. Hao, J. Ma, and L. Huang, "A new Approach to Intrusion Detection using Artificial Neural Networks and Fuzzy Clustering," *An International Journal of Expert Systems with Applications*, vol. 37, no. 9, pp. 6225–6232, 2010.

[38] J. Zhang and M. Zulkernine, "A Hybrid Network Intrusion Detection Technique using Random Forests," in *The First International Conference on Availability, Reliability and Security*. IEEE, 2006, pp. 8–pp.

[39] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection," in *Proceedings of the International Conference on Platform Technology and Service (PlatCon)*. IEEE, 2016, pp. 1–5.

[41] S. Cheung, B. Dutertre, M. Fong, U. Lindqvist, K. Skinner, and A. Valdes, "Using model-based intrusion detection for scada networks," in *Proceedings of the SCADA security scientific symposium*, vol. 46. Citeseer, 2007, pp. 1–12.

[42] L. A. Maglaras and J. Jiang, "Intrusion Detection In SCADA Systems using Machine Learning Techniques," in *Science and Information Conference (SAI), 2014*, 2014, pp. 626–631.

[43] A. F. S. Prisco and M. J. F. Duitama, "Intrusion detection system for scada platforms through machine learning algorithms," in *Communications and Computing (COLCOM), 2017 IEEE Colombian Conference on*. IEEE, 2017, pp. 1–6.

[44] P. Nader, P. Honeine, and P. Beauseroy, "l^p-norms in one-class classification for intrusion detection in scada systems," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2308–2317, 2014.

[45] N. Sayegh, I. H. Elhajj, A. Kayssi, and A. Chehab, "Scada intrusion detection system based on temporal behavior of frequent patterns," in *Electrotechnical Conference (MELECON), 2014 17th IEEE Mediterranean*. IEEE, 2014, pp. 432–438.

[46] O. Linda, T. Vollmer, and M. Manic, "Neural network based intrusion detection system for critical infrastructures," in *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*. IEEE, 2009, pp. 1827–1834.

[47] Feng, Cheng and Li, Tingting and Chana, Deeph, "Multi-level Anomaly Detection in Industrial Control Systems via Package Signatures and LSTM Networks," in *Proceedings of the 47th IEEE/IFIP International Conference on Dependable Systems and Networks*. IEEE, 2017, pp. 261–272.