EAI Endorsed Transactions
on Security and Safety
Research Article EAI.EU

Forget the Myth of the Air Gap: Machine Learning for Reliable Intrusion Detection in SCADA Systems

Rocio Lopez Perez[1], Florian Adamsky[2,3], Ridha Soua[3], and Thomas Engel[3]

[1]CSC, University of Luxembourg, Luxembourg
[2]Hof University, University of Applied Sciences, Germany
[3]SnT, University of Luxembourg, Luxembourg

Abstract

Since Critical Infrastructures (CIs) use systems and equipment that are separated by long distances, Supervisory Control And Data Acquisition (SCADA) systems are used to monitor their behaviour and to send commands remotely. 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. However, traditional Intrusion Detection Systems (IDSs) cannot detect attacks that are not already present in their databases. To this end, we assess in this paper Machine Learning (ML) techniques for anomaly 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 $> 96\%$.

Keywords: Critical Infrastructures, SCADA, Anomaly detection, Machine Learning, SVM, Random Forest, BLSTM

doi:xx.yyy/trans.journalid.V.n.i

1. 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, where 225,000 Ukrainian people were without electricity. These attacks were the first that resulted in a power outage [13].

Intrusion Detection Systems (IDSs) are the defacto protection standard for any IT system. The detection of intrusions with a traditional IDS requires a database containing signatures of different attacks, each signature corresponding to a specific attack and its characteristics. The main disadvantage of this technique is the need for human intervention to analyse vulnerabilities and threats to create unique signatures. Consequently, Machine Learnings (MLs) techniques are a good candidate to develop anomaly detection algorithms that can learn about normal behaviour and autonomously adapt to variations, acting without being pre-programmed or provided with an explicit pattern [14].

Our contributions: In this paper, we focus on assessing the performances of ML techniques such as Support Vector Machine (SVM), Random Forest (RF), and Bidirectional Long Short Term Memory (BLSTM) in detecting anomalies in SCADA systems. Section 2 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 3 describes the data set and the experimental setup in detail. In Section 4, 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 5 and compare them with our approach. Finally, we conclude our work in Section 6 and give directions for future research.

2. 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.

2.1. Attack Vectors on SCADA

As described in Section 1, adversaries can often 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 protocols 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.

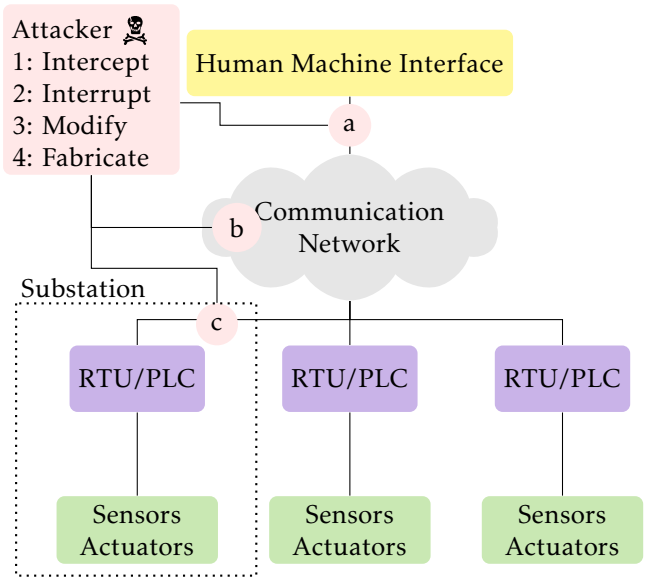

Figure 1. Attack model demonstrating four network attacks, denoted as (1–4), against a simplified SCADA architecture with three attack targets (a–c) based on [7].

In this context, one question arises: how many SCADA services are really reachable and therefore attackable over the Internet? With Internet-wide scanners such as ZMap [12], we can make an estimate of which SCADA protocols and how many sites are publicly available. The Censys [15] database uses ZMap to scan the whole IPv4 address space monthly for common network protocols. Among these SCADA protocols are Tridium Fox, Modbus [16], BACnet, Siemens S7, and DNP3 [6]. We analysed the data starting from August 2015 for Modbus and for the other protocols from September 2015 up to the beginning of October 2018. Results are plotted in Figure 2.

As shown in Figure 2, Tridium Fox and Modbus are the SCADA protocols most frequently encountered. On average over the measurement period, there are 402 DNP3 services available. We can measure a small decrease from 2016 to 2017 of 9% for DNP3. From 2017 to 2018 we have an 12% increase. For the Siemens S7 protocol, we have on average 4,142 available services with an clear increase in 2016. We measure 19.3% increase from 2016 to 2017 and 10.8% increase from 2017 to 2018. On average there are 16,686 available

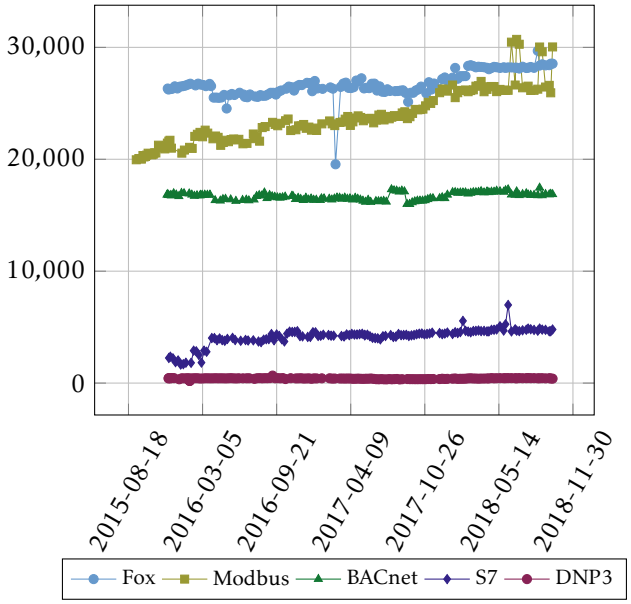

Figure 2. Available SCADA services reachable from the Internet starting from September 2015 up to the beginning of October 2018 based on data from the Censys database [15].

BACnet services without any significant change over the years. For Modbus, however, we measured 23,725 reachable services on average with an increase of 9.7% from 2016 to 2017 and from 2017 to 2018 we have an increase of 12%. The Tridium Fox protocol has on average 26,743 available services in the IPv4 address space without any significant changes over the previous year. It can be seen that Modbus is one of most heavily used protocols and its use is increasing. This is a dangerous trend and is our main motivation when working on the gas pipeline system data set provided by the Mississippi State University (MSU), as detailed in Section 3.

It is also worth mentioning that services of different operators are increasingly interlinked and consequently the boundaries of CIs are expanding beyond the previous single operator scenario. Hence, CIs are cyber-dependent: the information outputs of one CI are inputs to an other infrastructure. A failure in one CI can spread through cascading effects to other CIs, impacting large geographical regions. Consequently, attacks that manage to target interdependencies between CIs can result in the disappearance of vital operator services.

2.2. Machine-Learning Techniques for Anomaly Detection

Machine Learning methods play a key role in building models that separate normal behavioural patterns from abnormal ones within a data set [17]. Consequently, these techniques are considered suitable for anomaly detection problems [14]. Usually, anomalous patterns

appear in the data as outliers, peculiarities, alterations, conflicting observations or exceptions. In general, complying system behaviour patterns are easily localized and labelled as normal or expected behaviour within large amount of data. In contrast, non-complying patterns are less frequent and it is a challenging task to detect them. The use of an unsupervised algorithm can hinder anomalous patterns detection, since several anomaly detection approaches require labelled observation samples of normal and/or anomalous behaviours. Hence, researchers often opt for the use of semi-supervised or supervised learning algorithms to perform anomaly detection.

In this work, which is part of the Advanced Tools to assEss and mitigate the criticality of ICT compoNents and their dependencies over Critical InfrAstructures (ATENA) architecture [18], we aim to develop a Network Intrusion Detection System (NIDS) based on ML techniques to recognize attacks targeting SCADA systems, in an industrial use-case. In the following, SVM, RF and BLSTM will be introduced.

Support Vector Machine (SVM). SVM is a type of feed-forward neural network utilizing kernelized learning algorithms [19]. SVMs can be used for classifying linearly separable and non-linearly separable patterns, as a precursor to finding the optimal separating hyperplane by using a linear SVM.

Random Forest (RF). RF are based on the standard automatic learning technique known as a decision tree. The decision tree ensemble technique, developed by Leo Breiman, improves classification accuracy by incorporating randomness in the construction of each individual tree or classifier [20]. The training phase of RF works as follows:

- k independent and different bootstrapped samples, each of size n, are drawn as separate decision trees.
- Each of these decision trees is trained separately. The final classifier is the set of these trees.

In the testing phase, a novel or test observation flows through every decision tree, following one path or another, depending on the decisions at each node. The predicted class label is given by the decision having most votes from the different classifiers.

Long Short Term Memory (LSTM). LSTM is a powerful type of Recurrent Neural Network (RNN) used in Deep Learning (DL), designed to handle sequence dependences. It is trained using back-propagation through time, overcoming the vanishing and exploding gradient problems. A LSTM model uses a gating mechanism that allows the cells to keep a block of information for long periods of time [21] and protects the gradient inside the cell from harmful changes during the training phase. A LSTM model is composed of memory blocks, which contain one or more memory cells with a forget gate, an input gate and an output gate. The gates are multiplicative units with continuous activation, and are shared by all cells belonging to the same memory block. The key feature of LSTM units is their ability to to recall values for either long or short time periods. This is due to the lack of an activation function within its recurrent components and the presence of an additive interaction between the input transformations. This interaction ensures a smooth change of the cell state instead of abruptly modifying it. Thus, the stored value is not continuously reduced over time. Therefore, the gradient does not tend to vanish when performing back-propagation. Generally, additive operations distribute gradients equally and therefore LSTM allows much more effective back propagation of the error.

Long Short Term Memory can operate over sequences at the input and/or output at the same time. Given a fixed input vector size (red block in Figure 3), the recurrent neural network processes it using some hidden layers (green blocks) to produce a fixed output vector (blue blocks). Long Short Term Memory offers four different architectures [22]. In the current study, we use the many-to-many architecture, depicted in Figure 3. This architecture is suitable for synchronized input and output sequences. The LSTM layer returns sequences in order to transfer its information both to the next layer and to itself for the next time step. Thus, the many-to-many architecture will allow us to perform classification of sequence of packets where we wish to label each packet of the sequence, while still using the temporal information.

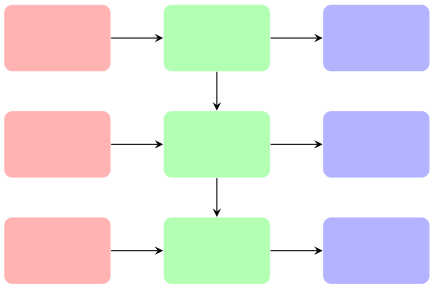

Figure 3. Many-to-many architecture of the LSTM algorithm. Rectangles represent vectors and arrows represent functions. Based on [22].

The mathematical concepts of the computational operations of a LSTM cell (depicted in Figure 4) are the following [21]:

f_t or forget gate: defines the amount of information, coming from the preceding cell that the current cell will maintain. f_t is multiplied by the C_{t-1} information and added to the current cell $\tilde{C}_t$ state.

i_t or input gate: defines the new candidate values that the current cell will maintain, scaled by the amount of computed information. Therefore, the current cell state becomes $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$.

o_t or output gate: defines the amount of information the cell needs to output: $o_t = sigm(W_{oh} \cdot h_{t-1} + W_{ox} \cdot x_t)$, and finally the current hidden block state is $h_t = o_t \cdot tanh(\tilde{C}_t)$. The hidden state contributes to both the prediction (rnn_{out}) and to the next iteration of the LSTM. The notations W_{oh} and W_{ox} represent the synaptic weights of the output gate, which determine how much importance should be given to both the present input and the past hidden state. The *sigmoid* and *tanh* are the activation functions that regulate the output of the neural network by ensuring that the values stay between 0 and 1, and -1 and 1 respectively.

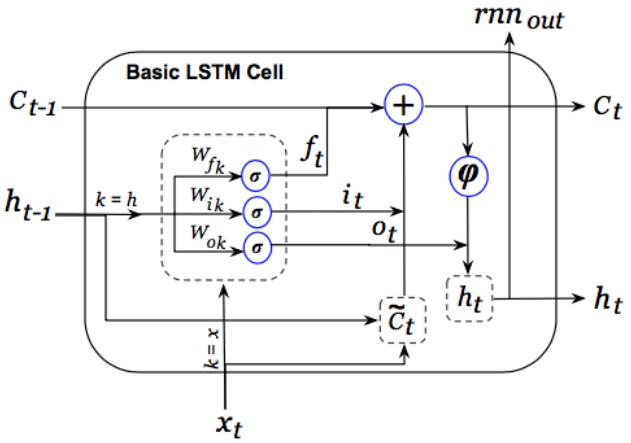

Figure 4. Basic Long Short Term Memory cell.

In our experiments, we used a bidirectional LSTM (BLSTM) algorithm. Unlike unidirectional LSTMs, BLSTMs preserve information from the past and the future [23]. This feature helps the model to converge faster, to consider both forward and backward time series dependencies, and therefore improves the overall prediction. It has been shown in several studies that the BLSTM algorithm performs better than the unidirectional LSTM algorithm [24–26] demonstrating that the future data also contains relevant information.

Figure 5 shows our BLSTM followed by a dropout layer, a dense layer and an activation function. All layers share parameters along the time dimension. The model's input is a sequence of network packets while the output is their corresponding predictions (benign or malicious network packet). For binary classification models, we use sigmoid activation function, binary cross-entropy loss function and adam optimizer. For multi-class classification models, we use softmax activation function, categorical cross-entropy loss function and adam optimizer.

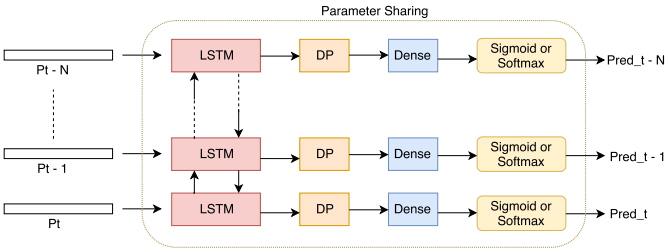

Figure 5. Bidirectional LSTM with a dropout and dense layer.

3. 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.

3.1. 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 [27] website. The real-world raw data was generated using a gas pipeline system provided by the 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 [28]. The data set, presents 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 1 lists the features and their corresponding types.

The features from Table 1 are strongly linked to a Modbus frame in a gas-pipe scenario. The address feature is an 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. An example of a Modbus frame can be seen in Figure 6. The two main modes in a gas pipeline infrastructure are read 0x03 and write commands 0x16, among other 256 different possible function codes. Some function codes can be used for malicious purposes

Table 1. List of features from the gas pipeline data set.

Nr.	Features	Types
1	Address	Network
2	Function Code	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

(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 was 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 incorrect Cyclic Redundancy Check (CRC) value to cause a Denial-of-Service (DoS) attack. The command

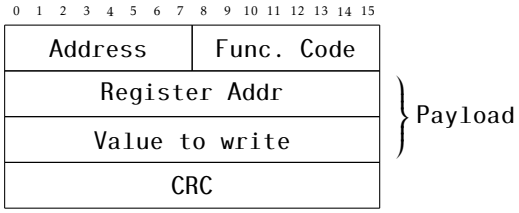

Figure 6. Example of a Modbus frame with a payload containing messages between MTU and RTU.

response feature, as its name indicates, helps the 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 1, 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 [29] to provide a broader perspective on the attacks that SCADA systems may suffer.

Table 2. 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 2, are the result of one or a series of external malicious activities through Modbus RTU packets. The attacks in Table 2 include a description of the attacks, a category and an attack type according to our attack model in Figure 1.

3.2. Methodology

Developing an ML-based IDS for intrusion detection in SCADA systems requires the steps illustrated in Figure 7. 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 2).

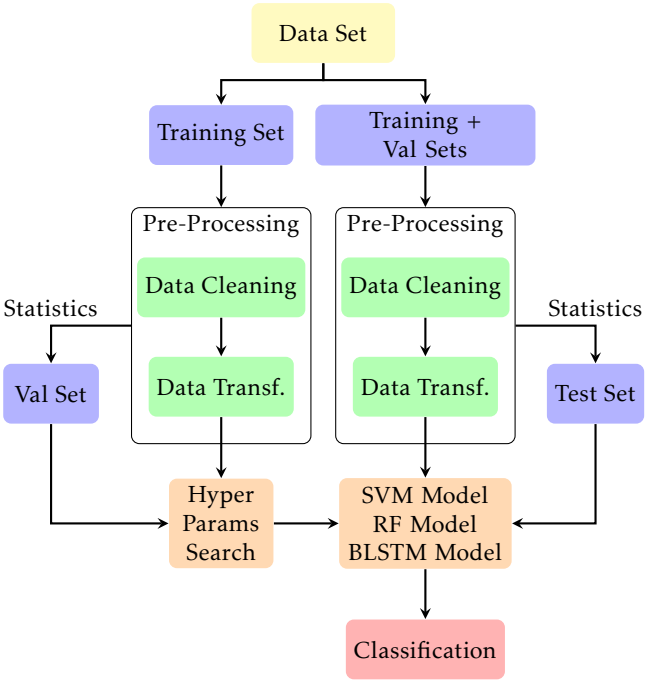

Figure 7. Flow chart diagram illustrating the steps of our work pipeline.

Data Cleaning. We observed that many feature values were missing or non-existent. The Table 3 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 3, 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 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 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*:

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

Lower values of J 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 [30].

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.

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

Table 3. 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

minimum and maximum value from a given feature and normalizing the feature values between 0 and 1. Hence,

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

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

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 [31].

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.

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.

4. Detecting Anomalies in SCADA: Experiment and Results Analysis

We developed our classification scripts with Scikit-learn[1], TensorFlow[2] and Keras[3]. Our source code is available on GitHub [32]. 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.

4.1. Key Performance Indicators

In any binary classification, the result can have four outcomes, two positive and two negative. This is illustrated in Table 4.

Table 4. Confusion matrix summarizing the outcome of any binary classification.

	Actual positive	Actual negative
Predicted positive	True Positive (TP)	False Positive (FP)
Predicted negative	False Negative (FN)	True Negative (TN)

To evaluate the classification of our ML algorithms, we used four Key Performance Indicators (KPIs): accuracy, precision, recall, and F_1 score. The hyperparameter search was performed by calculating the F_1 score of the validation set over the training set. In the literature [33], the frequently used measurement to evaluate the total number of correct predictions is

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{4}$$

In predictive analytics, the accuracy paradox suggests that predictive models with a certain accuracy value may have more significant predictive capability than other models with higher accuracy, making the accuracy an unreliable measure [34] [35]. Consequently, we used the accuracy measurement together with three other less misleading measures: precision, recall and F_1. The first measure is defined by

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

$$precision = \frac{TP}{TP + FP}, \tag{5}$$

which describes the samples correctly classified into a category relative to the total amount of samples classified into this category. The recall is defined by

$$recall = \frac{TP}{TP + FN}, \tag{6}$$

and describes the samples correctly classified into a category relative to the total amount of correct samples in this category. Finally,

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}, \tag{7}$$

is a weighted average of recall and precision. The best value of F_1 is 1 and the worst 0. In the next Section, we describe the techniques that we have used to deal with missing features in our data set.

4.2. Anomaly Detection Results

We split each of the sixteen data sets into training, validation and test sets according to the division in Section 3.2. 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 2). 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.

SVM Performance. Figures 8a, 8d, 8g, and 8j 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 8g while for CAT classification this value drops to 88.45 % (see Figure 8j). 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 than GMM but worse than Keep prior value. For both binary and categorical classifiers, the MEAN normalization strategy outperforms MIN-MAX normalization. Table 5 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 5. 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 %

RF Performance. Figures 8b, 8e, 8h, and 8k 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 6 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 6, 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 6. 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 %

BLSTM Performance. In Figures 8c, 8f, 8i, and 8l, 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 [30]. The Table 7 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 7, for both binary and categorical classifiers, the MEAN is better

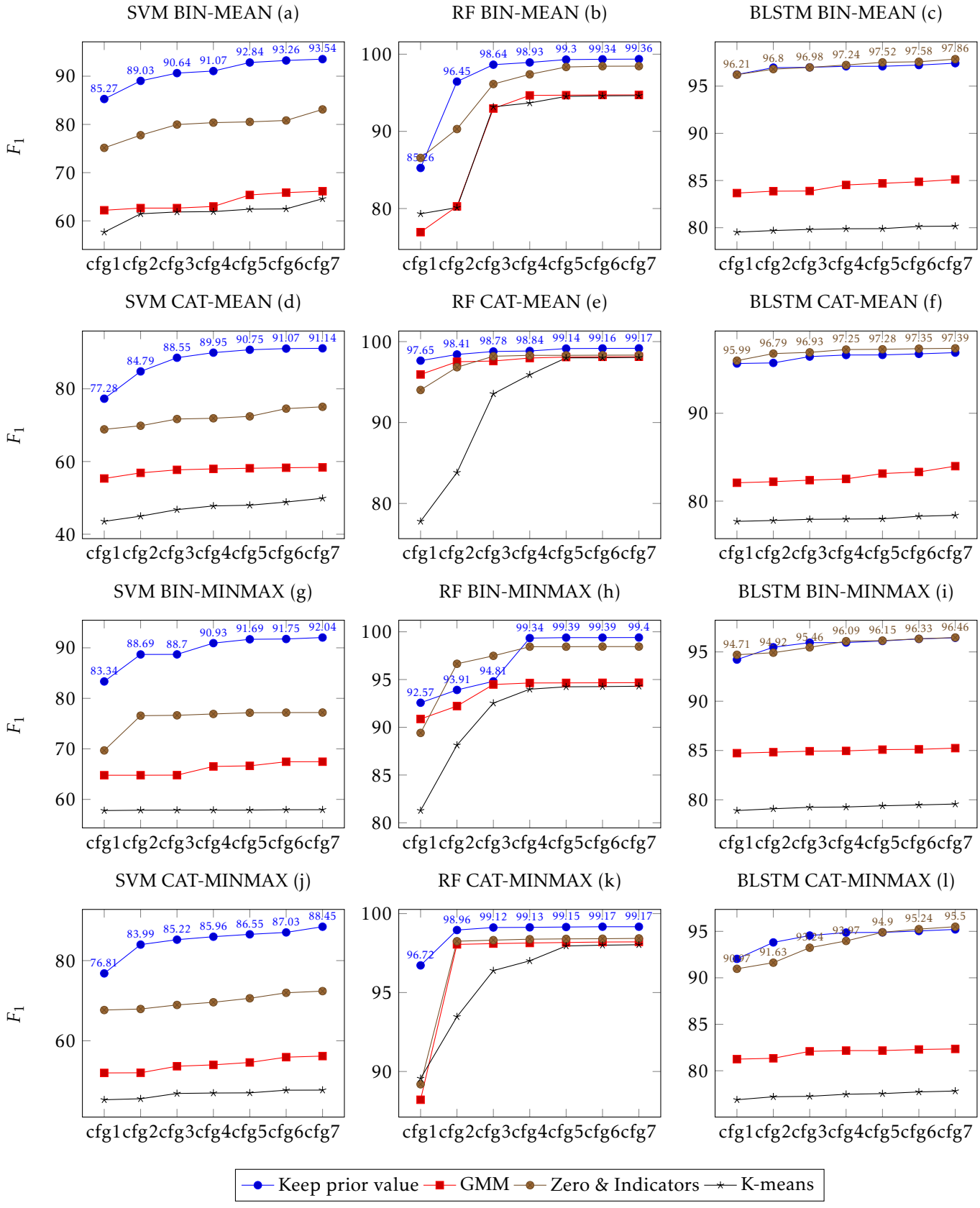

Figure 8. Results for the hyperparameter search for SVM, RF, and BLSTM. The first row shows the results for the binary classification (BIN) using the mean-standard deviation normalization strategy (MEAN), the second row for the categorical classification (CAT) using MEAN. The third row shows the results for BIN using the min-max normalization strategy (MIN-MAX) and finally the fourth row for CAT using MIN-MAX. On the x axis are the different configurations for the hyperparameter depicted and on the y axis is the F_1 score depicted.

than MIN-MAX. The difference between these two normalization strategies is 0.77% for BIN and 1.2% for CAT classification.

Table 7. 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 %

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 6 & 7. 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 NIDS.

The results from RF, which are listed in Table 9 show that it correctly classifies large numbers of normal and malicious packets. The categorical classification report in Table 8 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.

It is worth to mention that our ML-based intrusion detection can be carried out in real time once the model is trained correctly. This real-time capability has been demonstrated in several studies such as [36–39].

Table 8. 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 9. 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**

5. Related Work

Due to the advancements in connectivity and the sharp rise in the number of networks attacks, network anomaly detection has become a dynamic research area. Several surveys and review articles, which are available in the literature [36, 40, 41], discuss datasets and the common tools used during the different steps of anomaly detection. Readers are encouraged to refer to these paper for a comprehensive overview of up-to-date anomaly detection methods, their classification and open challenges. In this section, we focus on ML techniques for detecting anomalies in SCADA systems.

The SCADA systems were originally designed following the air gap principle and therefore without security measures in mind. Nowadays, these systems are in the spotlight of network attacks, due to standardization and connectivity to the Internet [2, 42]. 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 [37, 43–45]. 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 [46] 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 [47], 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. The same approach was adopted in [48] to detect anomalies in wind turbines in an early stage. However ML is used in the first step to estimate the values of Wind Turbines parameters. Deviation between the estimated values and the measurement using the Mahalanobis distance.

Both [38] and [39] presented an IDS that detects malicious network traffic in SCADA systems, based on One Class Support Vector Machine (OCSVM) technique. While authors of [38] use OCSVM to classify malicious observations by comparing them with benign ones, the study carried out in [39] 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 [49]. 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 [50]. 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 [51] 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 LSTM architecture was proposed in [52] to detect industrial control system anomalies.

While majority of researchers often opt for the use of semi-supervised or supervised learning algorithms to perform anomaly detection, Schuster et al. demonstrated the merit of using unsupervised learning for training anomaly detectors to identify intrusions in power plants [53]. To this end, they carried out a deep packet inspection of ongoing traffic followed by unsupervised ML for training the detection process.

In a previous work [54], we investigated how the ML techniques RF and SVM can detect attacks in a SCADA environment with the same data set. In comparison, in this paper we provide in-depth performances study of several ML algorithms for anomaly detection. Specifically, we introduce a more elaborated section on the attacks vectors in SCADA systems; BLSTM algorithm for binary and categorical classification with several normalization techniques; and extended related work section.

6. 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. Compared to the study carried out by [28], our paper provides a better understanding of the performances of machine learning techniques by proposing more KPIs (accuracy, F_1 score) for the binary and categorical classifications. In addition, we have investigated four techniques to handle the feature values in the data set that do not have any representation or meaning. We published our source code on GitHub [32] 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 2). 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 target of the same attacks that were studied in this paper.

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] Eric Byres. The Air Gap: SCADA's Enduring Security Myth. *Communications of the ACM*, 56(8):29–31, August 2013.

[2] Craig S. Wright. SCADA: Air Gaps Do Not Exist, September 2011. Accessed: 2017-12-04.

[3] Ralph Langner. Stuxnet: Dissecting a Cyberwarfare Weapon. *IEEE Security Privacy*, 9(3):49–51, May 2011.

[4] Kim Zetter. Meet 'Flame,' The Massive Spy Malware Infiltrating Iranian Computers, May 2012. Accessed: 2017-12-04.

[5] Vinay M Igure, Sean A Laughter, and Ronald D Williams. Security Issues in SCADA Networks. *Elsevier Computers & Security*, 25(7):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)*, pages 1–821, Oct 2012.

[7] Samuel East, Jonathan Butts, Mauricio Papa, and Sujeet Shenoi. A Taxonomy of Attacks on the DNP3 Protocol. In *International Conference on Critical Infrastructure Protection*, pages 67–81. Springer Berlin Heidelberg, 2009.

[8] Nicholas R. Rodofile, Kenneth Radke, and Ernest Foo. Real-Time and Interactive Attacks on DNP3 Critical Infrastructure Using Scapy. In *Proceedings of the 13th Australasian Information Security Conference (AISC 2015)*, pages 67–70, 2015.

[9] Peter Huitsing, Rodrigo Chandia, Mauricio Papa, and Sujeet Shenoi. Attack Taxonomies for the Modbus Protocols. *International Journal of Critical Infrastructure Protection*, 1:37–44, 2008.

[10] Peter Maynard, Kieran McLaughlin, and Berthold 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)*, September 2014.

[11] Ariana Mirian, Zane Ma, David Adrian, Matthew Tischer, Thasphon Chuenchujit, Tim Yardley, Robin Berthier, Joshua Mason, Zakir Durumeric, Alex. J. Halderman, and Michael Bailey. An Internet-wide view of ICS devices. In *14th IEEE Privacy, Security, and Trust Conference (PST'16)*, 2016.

[12] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *Proceedings of the 22Nd USENIX Conference on Security*, pages 605–620. USENIX Association, 2013.

[13] Robert M. Lee, Michael J. Assante, and Tim Conway. TLP: White Analysis of the Cyber Attack on the Ukrainian Power Grid. Technical report, E-ISAC, March 2016.

[14] Sherenaz W Al-Haj Baddar, Alessio Merlo, and Mauro Migliardi. Anomaly detection in computer networks: A state-of-the-art review. *JoWUA*, 5(4):29–64, 2014.

[15] Zakir Durumeric, David Adrian, Ariana Mirian, Michael Bailey, and J. Alex Halderman. A Search Engine Backed by Internet-Wide Scanning. In *Proceedings of the 22nd ACM Conference on Computer and Communications Security*, October 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] Charu C Aggarwal. *Data Mining: The Textbook*. Springer, 2015.

[18] Florian Adamsky, M Aubigny, F Battisti, M Carli, F Cimorelli, T Cruz, A Giorgio Di, C Foglietta, A Galli, A Giuseppi, F Liberati, A Neri, S Panzieri, F Pascucci, J Proença, P Pucci, L Rosa, and Ridha Soua. Integrated Protection of Industrial Control Systems from Cyber-attacks: the ATENA Approach. *International Journal of Critical Infrastructure Protection*, 21:72–82, 2018.

[19] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, September 1995.

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

[21] Christopher Olah. Understanding LSTM Networks, Aug 2015. Accessed: 2017-12-04.

[22] Andrej Karpathy. The Unreasonable Effectiveness of Recurrent Neural Networks, May 2015. Accessed: 2017-12-04.

[23] Mike Schuster and Kuldip K Paliwal. Bidirectional Recurrent Neural Networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[24] Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. Bidirectional lstm networks for improved phoneme classification and recognition. In Włodzisław Duch, Janusz Kacprzyk, Erkki Oja, and Sławomir Zadrożny, editors, *Artificial Neural Networks: Formal Models and Their Applications – ICANN 2005*, pages 799–804, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[25] Siddique Latif, Muhammad Usman, and Junaid Qadir Rajib Rana. Abnormal heartbeat detection using recurrent neural networks. *arXiv preprint arXiv:1801.08322*, 2018.

[26] Xin Zhang, Weixuan Kou, Eric I Chang, He Gao, Yubo Fan, Yan 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.

[27] Industrial Control System (ICS) Cyber Attack Datasets. `https://sites.google.com/a/uah.edu/`

tommy-morris-uah/ics-data-sets. Accessed: 2017-12-04.
[28] Ian Turnipseed. A New Scada Dataset For Intrusion Detection Research. M. sc., Mississippi State University, August 2015.
[29] Thomas Morris and Wei Gao. Industrial Control System Traffic Data Sets for Intrusion Detection Research. *Advances in Information and Communication Technology Critical Infrastructure Protection VIII*, pages 65—78, 2014.
[30] Zachary C. Lipton, David C. Kale, and Randall Wetzel. Directly Modeling Missing Data in Sequences with RNNs: Improved Classification of Clinical Time Series. In *Proceedings of Machine Learning for Healthcare 2016*, pages 253–270, 2016.
[31] James Bergstra and Yoshua Bengio. Random Search for Hyper-Parameter Optimization. *The Journal of Machine Learning Research*, 13(Feb):281–305, 2012.
[32] Machine learning techniques for Intrusion Detection in SCADA Systems. https://github.com/Rocionightwater/ML-NIDS-for-SCADA.git.
[33] Luis Talavera. Dynamic Feature Selection in Incremental Hierarchical Clustering. In *Proceedings of the European Conference on Machine Learning*. Springer, 2000.
[34] Francisco J Valverde-Albacete and Carmen Peláez-Moreno. 100% Classification Accuracy Considered Harmful: The Normalized Information Transfer Factor Explains the Accuracy Paradox. *PloS one*, 9(1), 2014.
[35] Xingquan Zhu. *Knowledge Discovery and Data Mining: Challenges and Realities: Challenges and Realities*. Igi Global, 2007.
[36] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez. Anomaly-based network intrusion detection: Techniques, systems and challenges. *Comput. Secur.*, 28(1-2):18–28, February 2009.
[37] Jihyun Kim, Jaehyun Kim, Huong Le Thi Thu, and Howon Kim. Long Short Term Memory Recurrent Neural Network Classifier for Intrusion Detection. In *Proceedings of the International Conference on Platform Technology and Service (PlatCon)*, pages 1–5. IEEE, 2016.
[38] Leandros A Maglaras and Jianmin Jiang. Intrusion Detection In SCADA Systems using Machine Learning Techniques. In *Science and Information Conference (SAI), 2014*, pages 626–631, 2014.
[39] Andrés Felipe Sánchez Prisco and M John Freddy Duitama. Intrusion detection system for scada platforms through machine learning algorithms. In *Communications and Computing (COLCOM), 2017 IEEE Colombian Conference on*, pages 1–6. IEEE, 2017.
[40] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita. Network anomaly detection: Methods, systems and tools. *IEEE Communications Surveys Tutorials*, 16(1):303–336, First 2014.
[41] A. L. Buczak and E. Guven. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys Tutorials*, 18(2):1153–1176, Secondquarter 2016.
[42] Bonnie Zhu and Shankar Sastry. Scada-specific intrusion detection/prevention systems: a survey and taxonomy. In *Proceedings of the 1st Workshop on Secure Control Systems (SCS)*, volume 11, 2010.
[43] Annie George. Anomaly Detection Based on Machine Learning: Dimensionality Reduction using PCA and Classification using SVM. *International Journal of Computer Applications*, 47(21), 2012.
[44] Gang Wang, Jinxing Hao, Jian Ma, and Lihua Huang. A new Approach to Intrusion Detection using Artificial Neural Networks and Fuzzy Clustering. *An International Journal of Expert Systems with Applications*, 37(9):6225–6232, 2010.
[45] Jiong Zhang and Mohammad Zulkernine. A Hybrid Network Intrusion Detection Technique using Random Forests. In *The First International Conference on Availability, Reliability and Security*, pages 8–pp. IEEE, 2006.
[46] Yi Yang, Kieran McLaughlin, Tim Littler, Sakir Sezer, Bernardi Pranggono, and HF Wang. Intrusion detection system for iec 60870-5-104 based scada networks. In *Power and Energy Society General Meeting (PES), 2013 IEEE*, pages 1–5. IEEE, 2013.
[47] Steven Cheung, Bruno Dutertre, Martin Fong, Ulf Lindqvist, Keith Skinner, and Alfonso Valdes. Using model-based intrusion detection for scada networks. In *Proceedings of the SCADA security scientific symposium*, volume 46, pages 1–12. Citeseer, 2007.
[48] Y. Cui, P. Bangalore, and L. B. Tjernberg. An anomaly detection approach based on machine learning and scada data for condition monitoring of wind turbines. In *2018 IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, pages 1–6, June 2018.
[49] Patric Nader, Paul Honeine, and Pierre Beauseroy. l^p-norms in one-class classification for intrusion detection in scada systems. *IEEE Transactions on Industrial Informatics*, 10(4):2308–2317, 2014.
[50] Naoum Sayegh, Imad H Elhajj, Ayman Kayssi, and Ali Chehab. Scada intrusion detection system based on temporal behavior of frequent patterns. In *Electrotechnical Conference (MELECON), 2014 17th IEEE Mediterranean*, pages 432–438. IEEE, 2014.
[51] Ondrej Linda, Todd Vollmer, and Milos Manic. Neural network based intrusion detection system for critical infrastructures. In *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*, pages 1827–1834. IEEE, 2009.
[52] 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*, pages 261–272. IEEE, 2017.
[53] F. Schuster, F. M. Kopp, A. Paul, and H. König. Attack and fault detection in process control communication using unsupervised machine learning. In *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*, pages 433–438, July 2018.
[54] Rocio Lopez Perez, Florian Adamsky, Ridha Soua, and Thomas Engel. Machine Learning for Reliable Network Attack Detection in SCADA Systems. In *Proceedings of the 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications (TrustCom)*, pages 633–638, Aug 2018.