international journal of critical infrastructure protection 38 (2022) 100542

Contents lists available at ScienceDirect

International Journal of Critical Infrastructure Protection

journal homepage: www.elsevier.com/locate/ijcip

Cyber-attacks detection in industrial systems using artificial intelligence-driven methods

Wu Wang [a], Fouzi Harrou [b,*], Benamar Bouyeddou [c,d], Sidi-Mohammed Senouci [e], Ying Sun [b]

[a] Center for Applied Statistics and School of Statistics, Renmin University of China, Beijing 100872, China
[b] King Abdullah University of Science and Technology (KAUST) Computer, Electrical and Mathematical Sciences and Engineering (CEMSE) Division, Thuwal 23955-6900, Saudi Arabia
[c] STIC Lab, Department of Telecommunications, Abou Bekr Belkaid University, Tlemcen, Algeria
[d] LESM Lab., University of Saida-Dr Moulay Tahar, Faculty of Technology, Saida, Algeria
[e] DRIVE Laboratory, University of Burgundy, Nevers, France

ARTICLE INFO

Keywords:
XGBoost
Stacked deep learning
SCADA system
Intrusion detection
Critical infrastructure protection
Cyber-attacks

ABSTRACT

Modern industrial systems and critical infrastructures are constantly exposed to malicious cyber-attacks that are challenging and difficult to identify. Cyber-attacks can cause severe economic losses and damage the attacked system if not detected accurately and timely. Therefore, designing an accurate and sensitive intrusion detection system is undoubtedly necessary to ensure the productivity and safety of industrial systems against cyber-attacks. This paper first introduces a stacked deep learning method to detect malicious attacks in SCADA systems. We also consider eleven machine learning models, including the Xtreme Gradient Boosting (XGBoost), Random forest, Bagging, support vector machines with different kernels, classification tree pruned by the minimum cross-validation and by 1-standard error rule, linear discriminate analysis, conditional inference tree, and the C5.0 tree. Real data sets with different kinds of cyber-attacks from two laboratory-scale SCADA systems, gas pipeline and water storage tank systems, are employed to evaluate the performance of the investigated methods. Seven evaluation metrics have been used to compare the investigated models (accuracy, sensitivity, specificity, precision, recall, F1-score, and area under curve, or AUC). Overall, results show that the XGBoost approach achieved superior detection performance than all other investigated methods. This could be due to its desirable characteristics to avoid overfitting, decreases the complexity of individual trees, robustness to outliers, and invariance to scaling and monotonic transformations of the features. Unexpectedly, the deep learning models are not providing the best performance in this case study, even with their extended capacity to capture complex features interactions.

1. Introduction

The latest (Fourth) generation of SCADA (Supervisory Control and Data Acquisition) systems integrate different internet and networks technologies, such as Cloud computing, Internet of things (IOT), big data, and web services. Such systems can provide real-time notifications, allowing administrators to use a variety of platforms to access the system and execute many new complex control algorithms [1]. Despite these advantages, this growing connectivity has made them more vulnerable to different forms of cyber-attacks. For instance, several severe attacks that targeted critical infrastructures were reported in recent years, including nuclear plants, power systems, water distribution, and transportation systems [2–4]. Therefore, cybersecurity becomes a fundamental concern for researchers and operators who strive to develop high-performance detection procedures for SCADA-based systems [5,6]. Several security mechanisms, including intrusion detection systems and cryptography, are adopted to secure and protect industrial systems from security threats [7].

Numerous intentional cyberattacks targeting industrial systems have occurred in the last two decades. These attacks highlight the lack of a security-driven approach to building up and maintaining industrial systems. Hackers exploit vulnerabilities in these critical systems to penetrate and gain access to SCADA networks that monitor physical processes, collect, manipulate and destroy critical data exchanged between facilities and operators, and implant malicious malware to disrupt the normal operating conditions of systems. Key targets include petroleum pipeline operators, chemical and petrochemical plants, smart

* Corresponding author.
E-mail addresses: wu.wang@ruc.edu.cn (W. Wang), fouzi.harrou@kaust.edu.sa (F. Harrou), benamar.bouyeddou@univ-saida.dz (B. Bouyeddou), Sidi-Mohammed.Senouci@u-bourgogne.fr (S.-M. Senouci).

https://doi.org/10.1016/j.ijcip.2022.100542
Received 12 January 2022; Received in revised form 5 June 2022; Accepted 11 June 2022
Available online 18 June 2022
1874-5482/© 2022 Elsevier B.V. All rights reserved.

power grids, and water treatment plants. For example, in May 2021, a ransomware cyberattack disrupted all operations at the Colonial Pipeline, the largest natural gas pipeline in the United States. The company declared that it had paid hackers 4.4 million dollars to provide it with decryption tools to restore its business and reactivate the crippled computer network [8]. In 2006, a water filtering plant in Pennsylvania, USA, was attacked by installing malicious software that can affect the normal operation of the water treatment plant [9]. Another attack on SCADA systems occurred in 2009 when cyberspies infiltrated the US power grids by leaving programs that could be used to disrupt the system [10]. Spies from Russia and other countries were trying to take control of the US electrical grid. In 2012, oil and chemical companies in the Middle East, in particular, suffered from a cyber-attack campaign by unknown parties using malware called Shamoon [11]. In this cyber-espionage campaign, unknown attackers deployed sophisticated Shamoon malware to delete data from computers with Microsoft Windows and then tamper with the master boot record of storage media, rendering the computer to become inaccessible. Unfortunately, Shamoon-based cyberattacks destroyed the content of roughly 30 000 workstations. In 2016, the Kemuri water plant was compromised by attackers who hacked into its utility control systems and altered the levels of chemicals used to treat tap water, according to Verizon Security Solutions [12]. More specifically, the hackers penetrated some computers of the Kemuri water plant via unpatched web in its defenseless internet-facing customer payment portal. They disrupted the system by negatively affecting water treatment and production capacities, increasing water supply recovery times. The damage was minimized due to the hackers' limited knowledge of ICS/SCADA systems or the intention to do any harm, but it could have severe consequences. In December 2015, a cyber-attack on the Ukraine power grid caused circuit breakers at 30 substations to trip, cutting power to around 225,000 customers [13]. A denial of service (DoS) targeted the telephone system and communication network, making the call center unavailable to customers. In addition, the malware implanted on the human-machine interface (HMI) was employed for deleting software on the system, which prevented the operators from characterizing the extent of the power outage and hampered repair actions. SCADA equipment was rendered inoperable, and power restoration had to be completed manually. It could have resulted in severe damage to the power grid. In 2017, industrial safety systems in the Middle East were targeted by cyber-attack campaigns known as Triton, Trisis, or Hatman. The attackers managed to penetrate Schneider Electric's Triconex safety instrumented system (SIS) and alter in-memory firmware by adding malicious functions, enabling them to modify memory contents and execute code on demand by receiving specially crafted network packets from a Command and Control server [12].

1.1. Contribution

This work leverages machine learning and deep learning techniques to support IDS in detecting cyber attacks. In other words, we explore the increasing important role of shallow and deep learning classification mechanisms to detect malicious cyberattacks in critical industrial infrastructures. Firstly, we introduced an innovative anomaly detection method that uses an ensemble of deep learning models to detect and identify cyber-attacks in SCADA systems. The deep ensemble model amalgamates five deep learning models to deeply learn the features of the malicious activities and distinguish them from nominal features. The deep learning models require neither feature engineering nor assumption on the data distribution, making them very attractive for cyber-attack detection. Additionally, in this paper, we investigated eleven machine learning models for malicious attack detection, including the Xtreme Gradient Boosting (XGBoost), Random forest, Bagging, support vector machines with different kernels, classification tree pruned by the minimum cross-validation and by 1-standard error rule, linear discriminate analysis, conditional inference tree, and the C5.0 tree. To the best of the authors' knowledge, some potential methods, such as XGBoost, have not been employed to detect attacks in SCADA systems. To address this research gap, we investigate the performance of shallow machine learning and deep learning methods to detect intrusion in SCADA systems. Importantly, this study evaluates different machine learning/deep learning methods using real datasets with different kinds of cyberattacks from two laboratory-scale SCADA systems: gas pipeline and water storage tank systems. We use four common performance metrics to compare and check the detection quality of different models: accuracy, precision, recall, and F-measure. Results revealed that the XGBoost model yields higher detection performance than other investigated models and outperforms the state-of-the-art methods. We found that cyber-attack data constantly exhibits outliers, e.g., a huge value for some features, even when the system is operating in normal conditions. The success behind machine learning methods such as random forest and XGBoost is largely due to their robustness to outliers. In contrast, stacked deep learning methods are more sensitive to outliers, because outlier-sensitive linear transformations are fundamental operations in neural networks.

Section 2 highlights literature reviews on the related works and Section 3 introduces the proposed deep learning-based malicious attack detector and the considered machine learning methods. Section 4 assesses the proposed method and compare its performance using datasets for the test-beds: gas pipeline system, and Water storage tank systems. Finally, Section 5 concludes this study and sheds light on potential future research lines.

2. Related works

Cyber security in industrial control systems is becoming an indispensable component in designing modern industrial systems, such as water treatment plants, power grids, and gas pipeline systems. Cyber-attack detection in industrial systems and critical infrastructures gained much consideration in the last two decades to protect industrial control systems from cyber-attacks. In [14], Mathur and Tippenhauer presented a Secure Water Treatment testbed to assess the power of attack detection algorithm. Recently, Faramondi et al. provided an industrial control system dataset collected from a hardware-in-the-loop Water Distribution Testbed (WDT) that contains physical, and network datasets, which support researchers in evaluating intrusion detection methods dedicated to deal with threats in Cyber-Physical Systems (CPS) [15]. In [16], Poojitha at al. proposed attacks classification technique based on Feedforword neural network (FNN). In [17], the ANN (Artificial neural network) was used to deal with external malicious data in SCADA systems. Unfortunately, both aforementioned techniques are not appropriate to handle attacks with temporally dependent data. In [18], Huan et al. used the convolutional neural network (CNN) to discriminate prominent temporal patterns and isolate samples that contain specific and non specific SCADA attacks. Unfortunately, the proposed detector requires that attackers inject bad packets when it inspects only the headers ignoring completely their payload. As it supposes the packets are not encrypted as well. In [19], Barbosa et al. developed a flow-based whitelisting technique to identify legitimate data in SCADA networks based on TCP and UDP transport protocols. The whitelists are constructed using clients and servers address, server's port and the used transport protocol. Nevertheless, detection results show high rate of false positives that are caused by non-considered legitimate flows. In [20], Chaojoun et al. addressed false data injection attacks in power systems via the Kullback–Leibler Divergence (KLD). Accurately, KLD metric tends to take high values in the presence of such attacks. However, detection performance depends principally on the preestablished detection threshold. In [21] Maglaras merged the one class support vector machine to K-means algorithm using five attributes namely: packets size, rate, same destination, same source and destination and ARP (Address resolution protocol) packets. Besides that these features are manually chosen, to set them, the detection

method consists to control only last ten packets each time. In [22], Farah et al. developed the PPFSCADA (Privacy preserving framework for SCADA) which is a permutation method to preserve privacy. In [23], Shitharth et al. introduced a two stages intrusion detection algorithm in SCADA network. The first stage uses the IWP-CSO (Intrusion weighted particle based Cuckoo search optimization) technique to select the most relevant features. Then in the second stage, normal and anomalous events are labeled and classified through a Hierarchical neuron architecture based neural network (HNA-NN) procedure. In [24], Xue et al. targeted the false data injection attacks using a detection structure named ELM-OCON (extreme learning machine ELM-based one-class-one-network). To attain high classification rate, the proposed solution uses bad data built from alternating current state estimation to learn ELM in the subnet of OCON state identification layer and separate them from normal data. In [25], Al Zaki Khan et al. assessed the performance of three machine learning classifiers namely naïve bayes, PART and random forest using a gas pipeline dataset. Simulation results show that PART and random forest presented in general good performance in opposition to naïve bayes. In [26], Wang et al. implemented a server–client intrusion detection system called HOIDS (hierarchical online intrusion detection system. The main server is installed at the control center and manages various IDS clients that are placed in all modules of the SCADA network. Each IDS client performs a logistic regression-based technique to classify and build de detection patterns. In [27], Erez et al. presented a domain-aware anomaly detection system to reveal potential abnormal deviations in SCADA control registers. A distribution-based automatic classification windows technique was applied to learn the system, determine the existing registers and establish the normal profile for each class. Then the detection step (i.e. enforcement phase) uncover deviations from the created profiles. In [28], a combination between autoassociative kernel regression (AAKR) model and the statistical probability ratio test (SPRT) was used to prevent denial of services (DOS) attacks. The detection rules consider several parameters (criteria) including CPU load, idle time, link utilization and log failure. In [29], an intrusion detection system using the Hidden Markov model (HMM) is presented. The system integrates two subsystems depending on targeted attacks. The data subsystem is charged to track data-based attacks, while the header subsystem controls the header's fields to reveal modifications, eventually manipulated by attackers. On each subsystem, multiple HMM models are trained using the Baum–Welch algorithm. Then, the forward–Backward algorithm is performed in the detection step, and a decision threshold is fixed accordingly. In practice, both subsystems should be implemented as preprocessors on appropriate IDS platforms. The classification accuracy achieved by the HMM-based approach is around 93.36% in discriminating normal data from attacks in the Water storage tank testbed. In [30], Demertzis et al. presented a new implementation of one-class classification, called SOCCADF (Spiking one-class anomaly detection framework), to recognize different forms of advanced persistent threats. SOCCADF relies on free-forward and multilayer architecture of evolving spiking neural networks (eSNN). Spikes mapping and parameters optimization within eSNN are performed using ROPE encoder (Rank order population encoding) and vQEA (Versatile quantum-inspired evolutionary) techniques. Rule classification is totally based on attacks-free data and calculated using a heuristic selection method. Compared to one-class SVM and one-class Combining Density and Class Probability Estimation (OCC-CD/CPE) detectors, the SOCCADF algorithm showed better detection performance.

3. Methodology

In this section, we introduce the details of the implemented machine learning models. In summary, the classification tree and the classification tree-based methods, e.g., random forest, boosting, and bagging, are robust to outliers and invariant to scaling and monotonic transformations of the features. The deep learning model is able to capture complex interactions of features but it is sensitive to outliers.

3.1. Classification trees

The classification tree [31] is a directed acyclic decision tree that predicts the class membership for subjects with observed features. The classification tree consists of internal nodes and terminal nodes. The internal nodes each have two descendants and a decision rule. The terminal node corresponds to a target value. We predict the target value of a new subject by moving the subject through the decision tree until it reaches one of the terminal nodes. In an internal node, including the root node, the subject is assigned to one of its two descendants by testing the corresponding decision rule. When the new subject reaches a terminal node, we predict the target value of the subject by the value attached to the terminal node.

Denote the training data as $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, where the features are $\mathbf{x}_j = (x_{j1}, \ldots, x_{jp})$, y_j is either *attack* or *normal* which labels the observation, p is the number of features and n is the number of training samples. To train a classification tree, we need to determine the structure of the tree and the decision rule of every internal node. The decision rules are of the type $\{x > c\}$ for a feature x and a constant c. In a internal node, the decision rule is chosen according to the Gini impurity. The Gini impurity is defined as

$$Gini(\mathbf{p}) = 1 - \sum_{i=1}^{C} p_i^2, \tag{1}$$

where $\mathbf{p} = (p_1, \ldots, p_C)$ are the proportion of classes in the training data that reach this internal node, and C is the number of possible classes taken by y. In a terminal node, the class label with the maximum proportion is assigned to this terminal node.

Based on the Gini impurity, the classification tree is first grown to the maximum depth such that each terminal node is pure or the maximum number of training samples in every terminal node is less than a small number. The classification tree is then pruned in a way such that the classification error and the number of terminal nodes in the tree is balanced. Specifically, we minimize

$$R(T) + \alpha|T|, \tag{2}$$

where the empirical classification error $R(T)$ of a tree T is defined as the frequency of falsely classified events which is estimated by cross-validation, that is if $\hat{y}_i, i = 1, \ldots, n$ are the predicted events by the tree T, the empirical classification error is defined as $n^{-1}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$, $I(.)$ is the indicator function, $|T|$ is the number of terminal nodes in the tree T, and the parameter α is a trade-off between estimation error and the tree-size.

The classification tree naturally handles both continuous and discrete features. With some modifications, the classification tree can also deal with missing values in the features. It is invariant to scaling and monotonic transformations of the features. For example, log-transform the continuous features in the training data will not affect the resulting tree. Most importantly, it is robust to outliers. For example, if the maximum value of a feature is increased to 10^{30}, either by mistake or on purpose, the constructed tree or the prediction will not be distorted because the decision rule $\{x > c\}$ in the internal node is robust to large values of features. Some of the attacks attempt to inject features with huge values to distort the cyber-attack detection system in the application. The classification tree and the tree-based methods, e.g., random forest, boosting and bagging, have the advantage that it is immune to this kind of attacks. In contrast, the deep neural networks, the linear discriminate analysis, and the support vector machines are sensitive to outliers, and they are not able to detect these attacks.

3.2. Bagging

Sometimes a single classification tree cannot capture complex interactions between features and does not generalize well to new samples. The vital weakness of a classification tree is its structural instability,

that is, if we built two trees based on different yet identically distributed data, the structure of the resulting trees will be significantly different. In other words, a classification tree has a large variance and a small bias. To solve this problem, Breiman [32] proposed the bootstrap aggregating, or the bagging method that trains multiple classification trees using bootstrap samples of the training data. When predicting the class label of a new subject, bagging follows a plurality vote. Bagging has a smaller variance compared to classification trees.

3.3. Random forest

Bagging generates multiple classification trees using bootstrap samples of the training data. Because the classification tree algorithm minimizes the impurity measure at every internal node, the resulting trees in the bagging method repeatedly select some significant features, and the trees are highly correlated with each other. The high correlation between trees hinders the prediction accuracy of the bagging method. Similar to bagging, random forest [33] also trains a cluster of classification trees by bootstrapping the training data. To de-correlate the trained trees, a further randomization step is adopted in the tree training algorithm. That is, the splitting feature is selected from a random subset of all features. Let $\mathbf{x}_s = \{x_{i_1}, \dots, x_{i_k}\}$ be a random subset of $\mathbf{x} = \{x_1, \dots, x_p\}$, where $k \leq p$ denotes the number of candidate features and is a tuning parameter of random forest. Random forest restricts the features to be $\mathbf{x}_s$ when minimizing the Gini impurity (1). The number k is selected by cross-validation in our implementation.

Because random forest aggregates a cluster of classification trees, it inherits the advantages of classification trees. That is, random forest is also robust to outliers, it is invariant to scaling and monotonic transformations of the features, and it handles both discrete and continuous features. [34] recommends random forest as the best off-the-shelf machine learning methods. In practice, we implement the random forest algorithm by the R package ranger [35].

3.4. Xgboost

Xgboost [36] is an open-source software library that implements the regularized gradient boosting method. The gradient boosting algorithm [37–39] is an ensemble algorithm that boosts the performance of weak base learners, such as shallow classification trees. When the response variable is continuous, and the loss function is the squared loss, the gradient boosting algorithm repeatedly fits shallow regression trees to the residuals of the response, and the resulting prediction rule is a weighted average of the fitted trees. The xgboost package further generalizes the gradient boosting method and includes strategies such as penalization of tree parameters, proportional shrinking of terminal nodes, extra randomization, and automatic feature selection. Xgboost is flexible and efficient and is adopted in many winning data mining competitions [36].

Specifically, xgboost trains a cluster of additive classification trees to predict the output. Let K denote the number of trees, $f(.)$ denotes a tree as a function of the input features, and $l(.)$ denotes a loss function such as the entropy loss, the objective function of xgboost can be written as

$$\sum_{i=1}^{n} l\left(y_i, \sum_{k=1}^{K} f_k(\mathbf{x}_i)\right) + \sum_{k=1}^{K} \Omega(f_k) \tag{3}$$

where $\Omega(f) = \gamma|T| + \lambda\|w\|^2$, γ and λ are tuning parameters, $|T|$ is the number of terminal nodes of the tree f, the parameter w denotes the vector of probabilities of the tree f predicting an attack on each of its leaf nodes, y is the label of the data. By penalizing the number of terminal nodes and the parameters of the tree, xgboost avoids overfitting and decreases complexity of individual trees. In practice, the objective function (3) is optimized in an iterative and greedy manner.

Xgboost is a well-designed software package, it scales to billions of samples in distributed and memory-limited settings. Xgboost accelerates training by utilizing sparseness of features, by designing a quantile sketch algorithm for internal nodes splitting, and by exploiting out-of-core computation resources. Similar to classification trees and random forests, xgboost is robust to outliers of features, it is invariant to scaling and monotonic transformations of features, thus saves efforts for feature engineering. By penalizing the complexity of individual trees and injecting extra randomization through sampling subjects and features, xgboost is more robust to overfitting compared to random forest.

3.5. The stacked deep learning-driven method

Deep learning methods is successful in various fields such as speech recognition [40], natural language processing [41], and computer vision [42]. Deep neural networks consist of multiple layers of artificial neurons. It automatically recovers complex interactions between features from the training data. Detecting cyber-attacks is a pattern recognition problem, a field in which deep learning has been shown to be one of the most powerful methods.

While a single deep neural network may overfit the training data and does not have enough power for discriminating cyber-attacks, we adopt an ensemble learning strategy and develop a stacked deep learning method. Stack generalization [43] is a popular strategy widely adopted in the machine learning community to boost the performance of machine learning algorithms. Stack generalization learns an additional model based on the outputs from the base learners. In this work, we use model averaging, which is a simple stacked generalization method, to boost the performance of deep learning methods in cyber-attack discrimination in SCADA systems. Algorithm 1 illustrates the steps to train and predict with the stacked deep learning neural network.

We will first show theoretically that a stacked deep neural network can promote the prediction accuracy compared to a single deep learning model. In a binary cyber-attack detection problem, denote $y = 1$ as an attack, $y = 0$ as a natural event. Assume that we already trained N deep learning models $m_l(x), l = 1, \dots, N$, which are all better than guessing, that is $\mathbb{P}(m_l(x) = y) = p > 0.5$. The stacked deep learning model can be defined as by Eq. (4),

$$\begin{cases} m(x) = 1 & \text{if } \sum_{l=1}^{N} m_l(x)/N > 0.5 \\ m(x) = 0 & \text{otherwise.} \end{cases} \tag{4}$$

Assume that the trained deep learning models are independent for simplicity. Also assume that the new feature x_{new} corresponds to an attack, then the probability of the stacked deep learning model correctly predicts the attack is

$$\begin{aligned} \mathbb{P}(m(x_{new}) = 1) &= \mathbb{P}\left(\frac{1}{N}\sum_{l=1}^{N} m_l(x_{new}) > \frac{1}{2}\right) \\ &\geq 1 - \exp\left(-2\left(p - \frac{1}{2}\right)^2 N\right), \end{aligned} \tag{5}$$

where the last inequality follows from Hoeffding's inequality. As the number of stacked models N increases, the probability in (5) approaches 1, which shows that the stacked deep learning model can be arbitrarily accurate as the number of averaged models increases.

The theoretical analysis shows that the stacked deep learning model indeed increases classification accuracy. In practice, the individual models are trained with the same data, possibly with cross-validation, therefore individual models are correlated. A full analysis of the theoretical property of stacked deep learning model is beyond the scope of the current paper. Practically, the stacked deep learning model indeed shows higher ac- curacy compared to an individual deep learning model in our empirical study.

Limited by the available computing resources, we train five deep neural networks using the training data. The stacked deep learning

Algorithm 1: The algorithm for the stacked deep learning model.

Input: Training data: $(x_i, y_i), i = 1, \ldots, n$; New data: x_{new}
Output: The predictions by the stacked deep learning model: y_{pred}
for $l = 1 : 5$ **do**
1. Train individual deep learning model m_l using the training data;
2. Obtain the output of the trained model for the new data: $\hat{y}_l = m_l(x_{new})$.

end
Calculate the prediction of the stacked model: $y_{pred} = (\hat{y}_1 + \ldots, \hat{y}_5)/5$.

model predicts cyber-attacks by the majority votes of the outcomes of the five trained neural networks. The five trained neural networks each have three layers, and the number of hidden neurons in the five networks are $(80, 60, 60)$, $(80, 80, 60)$, $(100, 80, 80)$, $(120, 100, 80)$ and $(180, 120, 80)$, respectively. The number of hidden neurons are selected to cover small (Network 1) to moderately large (Network 5) neural networks. The activation functions of the hidden layers in the neural network are the rectified linear units function, and the activation functions of the output layers are the softmax function for the multiple-class cyber-attack detection problem. The networks are all implemented by Keras 2.4.3 and Tensorflow 2.3.0 with the RMSprop optimizer. All the networks are trained with a batch size of 256 for 5000 epochs.

The structure of individual neural networks is motivated by the available data and general neural network building techniques. Neural networks with more hidden layers will generally achieve better prediction accuracy in classification problems [44]. For example, successful designs for the ImageNet dataset all take advantage of huge networks with more than 30 hidden layers and millions of parameters [44]. On the other hand, the depth of the constructed neural network is restricted by the number of samples. Otherwise, an arbitrarily large network will eventually overfit the available data. In our experiment, a three-layer network can discriminate malicious attacks in a good accuracy without overfitting the training data. Large networks with more than four layers are more time-consuming to train and will generally overfit the data. We implement networks with different number of hidden neurons to cover small to large neural networks and compare their prediction accuracy in the experiment.

3.6. The algorithm for training an individual deep learning model

In this section, we describe the details for training a single deep learning neural network. First, we introduce some notations. Denote the output vector of the lth layer, including the input layer, by $\mathbf{a}^l = (a_1^l, \ldots, a_{n_l}^l)^\top$, where n_l is the number of neurons in the lth layer. The neural network has in total L layers. The output of the $(l+1)$th layer can be represented as

$$\mathbf{a}^{l+1} = \sigma\left(\mathbf{W}^{l+1}\mathbf{a}^l + \mathbf{b}^l\right),$$

where $\mathbf{W}^{l+1}$ is an $n_{l+1} \times n_l$ coefficient matrix corresponding to the mapping from the lth layer to the $(l+1)th$ layer, $\mathbf{b}^l$ is a bias vector, and $\sigma(x)$ is the activation function. For hidden layers, the activation function is $\sigma(x) = \max(0, x)$, which is the ReLU function. For the output layer, we use $\sigma(x) = 1/(1 + \exp(x))$, which is the sigmoid function, for binary cyber-attack detection problems, and $\sigma(x) = \exp(x_i)/(\exp(x_1) + \cdots + \exp(x_C)), i = 1, \ldots, C$, which is the softmax function, for multi-class cyber-attack detection problems where C is the number of events. Next, we introduce the objective function for the deep learning model. Denote (x, y) as a generic training data, where x is the feature, $y = (y_1, \ldots, y_C)$ is the one-hot encoding of the observed response. The loss function is

$$L(\mathbf{W}, \mathbf{b}) = \sum_{j=1}^{C} y_j \log(a_j^L). \tag{6}$$

We rely on the stochastic gradient descent (SGD) algorithm to minimize the objective function (6) and estimate the coefficients $\mathbf{W}, \mathbf{b}$ of the deep learning model. The core of SGD is the backpropagation (BP) algorithm, which calculates the gradients of the objective function. The BP algorithm calculates the derivative of the loss function (6) recursively from the last layer back to the first layer. Denote $\mathbf{z}^l = \mathbf{W}^{l+1}\mathbf{a}^l + \mathbf{b}^l$, and $\delta^l = \partial L/\partial \mathbf{z}^l$. Let $\odot$ denotes the element-wise product of two vectors. The BP algorithm is illustrated in Algorithm 2, and the SGD algorithm is shown in Algorithm 3.

In the literature, the technique of dropouts is widely adopted to prevent overfitting [45], and batch normalization [46] is used to accelerate training. We also implemented these techniques in our experiment. Fig. 1 shows the history of training a single neural network for discriminating the cyber-attacks in SCADA systems. As expected, the loss decreases, and the accuracy increases as the training progresses. When the training terminates, the loss and the accuracy stabilize, and the deep learning model learns the structure of the data.

3.7. Machine learning-based cyber-attacks detection framework

Fig. 2 illustrates the flowchart of the proposed cyber-attack detection procedure. The experiment is divided into the training phase and the detection phase. In more detail, in the data preprocessing step, we remove missing values and standardize numeric features. In the training step, we train the neural network and machine learning methods using the SGD algorithm. Finally, the trained model is verified using the testing data.

Algorithm 2: The backpropagation algorithm.

Input: Data: (x, y); The number of layers: L.
Output: The partial derivatives:$\partial L/\partial \mathbf{W}$ and $\partial L/\partial \mathbf{b}$.
Calculates the activations $\mathbf{a}^2, \ldots, \mathbf{a}^L$.
for $l = L : 2$ **do**
1. $\delta^L = \frac{\partial L}{\partial \mathbf{a}^L} \odot \sigma'(\mathbf{z}^L)$.
2. $\delta^l = ((\mathbf{W}^{l+1})^\top \delta^{l+1}) \odot \sigma'(\mathbf{z}^l)$ for $l < L$.
3. $\frac{\partial L}{\partial \mathbf{b}^l} = \delta^l$.
4. $\frac{\partial L}{\partial \mathbf{W}^l} = \delta^l (\mathbf{a}^{l-1})^\top$.

end

The machine learning methods implemented in the training and testing process are the following, neural networks with five different shapes (Net1-Net5), stacked neural networks (NetStack), XGBoost, random forest (RF), linear discriminant analysis (SLDA), decision trees pruned by the minimum cross validation error rule (RpartCV), decision trees pruned by the one standard error rule (Rpart1SE), conditional inference trees (CTree), a generalized conditional tree (C50Tree), Bagging, support vector machines with linear kernels (SVMLinear), support vector machines with polynomial kernels (SVMPoly), support vector machines with radial kernels (SVMRadial).

3.8. Metrics of effectiveness

In this study, five statistical scores are employed to quantify the performance of the studied methods computed using a 2×2 confusion

Algorithm 3: The stochastic gradient descent algorithm.

Input: Data: $\{(x_1, y_1), \ldots, (x_n, y_n)\}$; The number of epochs: E; The size of mini-batches: m.
Output: Network coefficients:**W** and **b**.
Randomly initialize the coefficients **W** and **b**.
for $i = 1 : E$ **do**
1. Randomly shuffle the input data and split it into blocks of size m.
2. For each mini-batch, calculate the partial derivatives by the BP algorithm and update the coefficients by

$$\mathbf{W} \to \mathbf{W} - \eta \frac{\partial L}{\partial \mathbf{W}}$$
$$\mathbf{b} \to \mathbf{b} - \eta \frac{\partial L}{\partial \mathbf{b}}$$

end

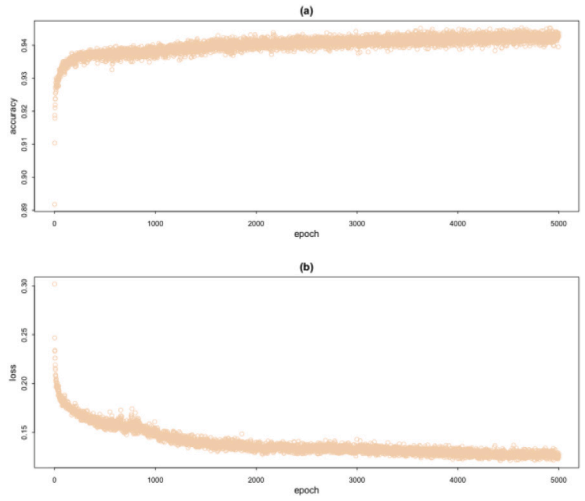

Fig. 1. The history for training a neural network to discriminate the cyber-attacks. Upper panel (a), the history of the accuracy measure. Lower panel (b), the history of the loss function.

Table 1
Confusion matrix associated to intrusion detection.

		IDS decision	
		Attack	**No attack**
Actual condition	Attack	True Positive (TP)	False Negative (FN)
	No attack	False Positive (FP)	True Negative (TN)

matrix (Table 1): Accuracy, Precision, Recall, F1-Score, and Area under curve (AUC). For a binary detection problem, the number of true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN) are used to compute the evaluation metrics.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}. \tag{7}$$

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{8}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}. \tag{9}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{10}$$

$$\text{F1} = 2\frac{\text{Precision.Sensitivity}}{\text{Precision} + \text{Sensitivity}} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}. \tag{11}$$

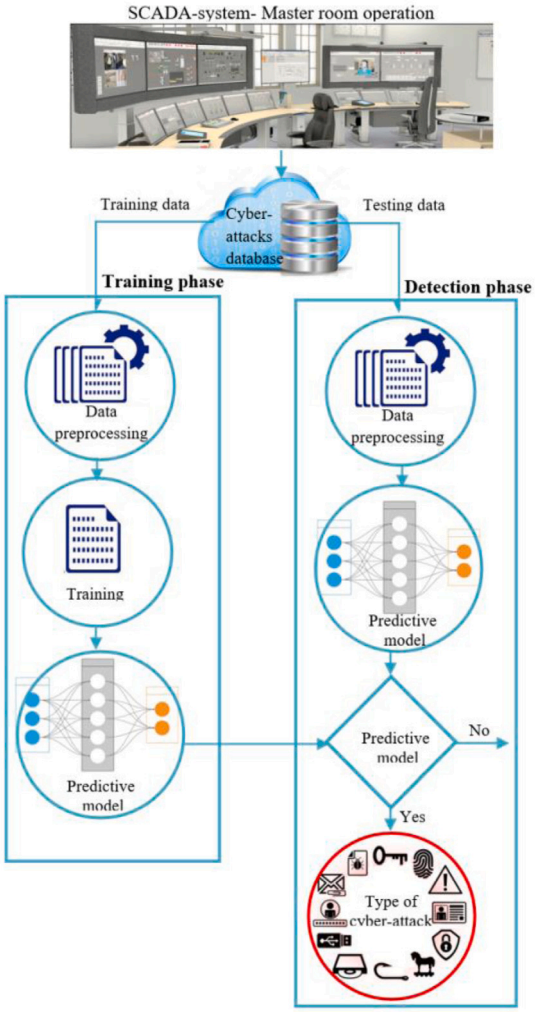

Fig. 2. Flowchart of the detection framework.

4. Results and discussion

This section is dedicated to verify the performance of the considered models using datasets from two testbeds: the gas pipeline and the storage tank testbeds from the Mississippi State University SCADA Laboratory [47]. The deployed physical process and control systems

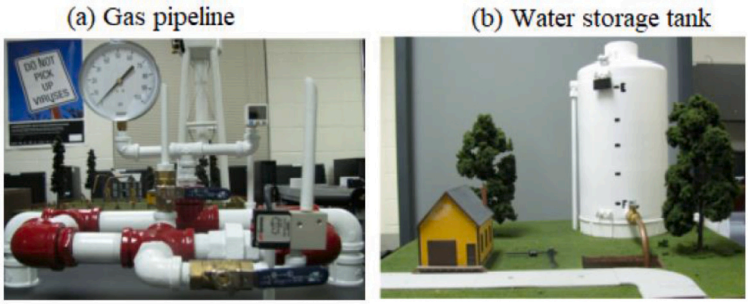

Fig. 3. Gas pipeline and water storage tank systems [47].

include gas pipeline and water tank storage in which normal operations are altered by different type of attacks (Fig. 3).

4.1. Gas pipeline dataset

In the first experiment, the effectiveness of the considered cyber-attack detectors to uncover cyber-attacks is investigated using SCADA datasets collected from the testbeds of SCADA Laboratory of Mississippi State University. The gas pipeline testbed mimics the behavior of a gas pipeline in transporting petroleum products to the market, while the storage tank testbed reproduces the oil storage tanks found in the petrochemical industry. This testbed includes principally sensors and actuators, a communication network, and supervisory control systems. The actuators are used to monitor the system state and maintain the pressure level. The internal and external communications are established via a Modbus RTU (Remote terminal unit) serial stack interface. In the supervisory control, MTU (Master Control Unit) is used to collect data from the associated RTU and forward their control commands, and connected to the HMI (Human-machine interface), thus offering the necessary information to the operator to monitor the system. Each sample collected from gas pipeline testbed data comprises 27 attributes describing heterogeneous variables, such as gas pressure, the pump state, PID controller's parameters, and the command packet length. The generated datasets represent the network traffic in both testbeds during their normal operations and when they face many types of threats and cyber-attacks. Precisely, about 28 scenarios were created, including reconnaissance attacks, response injection attacks, command injection attacks, and denial of Service attacks [47,48].

- *The reconnaissance attacks* aim to collect information about ModBus servers and the SCADA system (e.g., mapping network architecture, operational IP address, supported protocols, implemented services, open ports, operating system, devices in use, and their characteristics). Such information is then commonly used to uncover potential vulnerabilities or be useful for future attacks. The dataset comprises five reconnaissance attacks: address scan attack, function code scan attack, device identification attack, points scan attack, and memory dump attack. They are performed to identify the server IP address, detect the supported function code (public function code (PFC), user function code (UFC), reserved function code RFC and error function code (EFC)), to find device specifications (e.g., vendor and product names, and current code, release), for mapping data blocks addresses and memory, respectively.
- *The response injection attacks* are principally performed by exploiting the absence of authentication mechanism of exchanges within SCADA systems. Specifically, these attacks can seriously compromise the monitoring process by capturing, falsifying server's responses, forwarding falsified responses, and creating (i.e., injecting) new responses packets. This data contains two types of response injection attacks: naïve malicious response injection (NMRI) and complex malicious response injection (CMRI). The NMRI attacks consist only of injecting spoofed response packets. Many scenarios were conducted, including naïve read payload size, invalid read payload size, and naïve false error response. On the other hand, CMRI attacks penetrated to control information and changed them accordingly. Among CMRI scenarios, the dataset integrates calculated sensor measurement injection attacks, replayed measurement injection attacks, and high-frequency measurement injection attacks.
- *Command injection attacks* target essentially systems where most control operations are associated with the SCADA system, with minor human assistance. They create false administration commands to interrupt monitoring operations, suspend communications, and realize illegitimate settings. The generated attacks belong to the following types: malicious state command injection (MSCI), malicious parameter command injection (MPCI), and malicious function code command injection (MFCI).
- *Denial of Service attacks (DOS)* attempt to render the SCADA system inoperable and make the related monitoring process inaccessible. DOS attacks are generally launched using a large volume of traffics or malformed packets. This data considers two DOS attacks: Invalid Cyclic Redundancy Code (CRC) and Modbus Slave Traffic Jamming. The first attack is flooding based on an invalid CRC, while the second prevents salves to get access to the wireless link and pushes them to rest in the idle mode permanently.

The detailed description of all attacks and their concepts are well-reviewed in [47–50].

The classification metric of the seventeen considered classifiers for the gas pipeline dataset are tabulated in Table 2. Note that ten fold cross-validation has been implemented for the gas pipeline dataset. Accuracy measures the overall capability of a procedure for discriminating attacks. A high accuracy value indicates better detecting rate. From Table 2, XGBoost, RF, RpartCV, CTree, C50Tree, and Bagging have accuracy greater than 0.990, in particular, RF achieved the highest accuracy of 0.993. All these methods are related to classification trees which consist of multiple layers of simple decision rules.

One notable feature of the gas pipeline dataset is outlier. For example, the gas pipeline pressure measurements are distributed above 0 and below 32 in 90% of times, but in about 0.1% of the times, the gas pipeline pressure measurements are extremely large, with a value greater than 10^{20}, in the normal state without attacks. The naive malicious response injection attack also injects huge values of pressure measurements into the system. Theses records cannot be simply dropped in the training and testing process, because it reflects both normal states of the gas pipeline system in extreme conditions and features of attacks. Fortunately, classification trees are robust to outliers because it uses simple rules such as $\{x > c\}$ where x is a feature and c is a constant for discriminating attacks and normal states. Ensemble methods based on classification trees such as RF, Bagging and XGBoost inherit the robustness of trees, and therefore have superior performance in discriminating attacks. In contrast, neural networks and support

Table 2
Summary of classification metric based on testing the gas pipeline dataset.

Method	Accuracy	Sensitivity	Specificity	AUC	Precision	Recall	F1-score
Net1	0.850	0.732	0.941	0.836	0.879	0.732	0.799
Net2	0.915	0.889	0.946	0.917	0.907	0.889	0.898
Net3	0.967	0.923	0.994	0.958	0.989	0.923	0.955
Net4	0.930	0.866	0.993	0.930	0.986	0.866	0.922
Net5	0.946	0.880	0.986	0.933	0.974	0.880	0.924
NetStack	0.968	0.920	0.997	0.959	0.995	0.920	0.956
XGBoost	0.990	0.980	0.997	0.989	0.995	0.980	0.987
RF	0.993	0.987	0.997	0.992	0.995	0.987	0.991
SLDA	0.941	0.916	0.958	0.937	0.928	0.916	0.922
RpartCV	0.992	0.985	0.997	0.991	0.995	0.985	0.990
Rpart1SE	0.982	0.955	0.998	0.976	0.996	0.955	0.975
CTree	0.813	0.500	0.998	0.749	0.993	0.500	0.665
C50Tree	0.991	0.982	0.997	0.989	0.995	0.982	0.988
Bagging	0.992	0.987	0.996	0.991	0.993	0.987	0.990
SVMLinear	0.967	0.947	0.980	0.963	0.965	0.947	0.956
SVMPoly	0.972	0.957	0.981	0.969	0.967	0.957	0.962
SVMRadial	0.970	0.958	0.979	0.968	0.964	0.958	0.961

Table 3
The p-values testing the hypothesis that the stacked deep learning method is superior to the individual network for the gas-pipeline data.

Net1	Net2	Net3	Net4	Net5
0.013	0.130	0.481	0.013	0.058

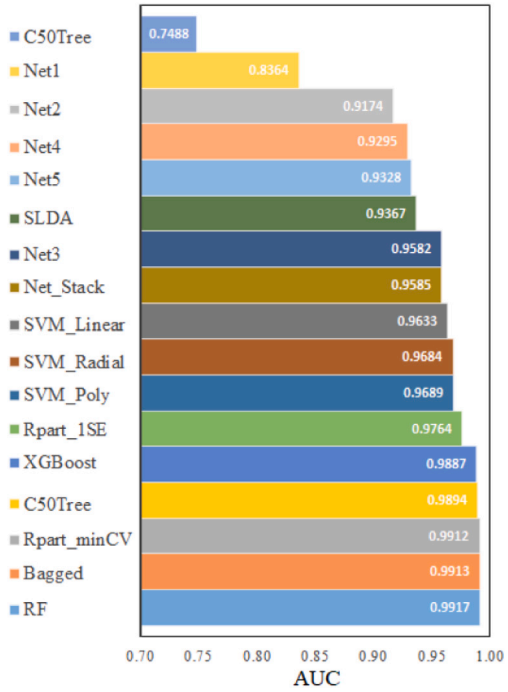

Fig. 4. Barplot of AUC values obtained by each detection method.

vector machines are sensitive to outliers. They have a lower accuracy compared to classification tree based methods.

In contrast, the deep learning and stacked deep learning methods, and support vector machine based methods are inferior to the tree based ensemble methods in discriminating cyber-attacks in this study. For example, the accuracy and F1-score are 0.968 and 0.956, respectively, for the stacked deep learning method, 0.972 and 0.962, respectively, for the support vector machine using the polynomial kernel, which are smaller than those of the XGBoost method. The main reason for the unsatisfactory performance of the deep learning and support vector machine methods are that they are not robust to outliers.

Overall, the stacked deep learning methods has higher accuracy than single deep learning methods. For example, the accuracy of the stacked deep learning method is 0.968 for the gas pipeline data, whereas the accuracy is 0.85, 0.915, 0.967, 0.93 and 0.946 for Net1 to Net5, respectively. We use a paired t-test to test whether the difference is statistically significant. The null hypothesis is that the accuracy are equal, the alternative is that the accuracy of a single deep learning model is less than that of the stacked deep learning model. From Table 3, we can see the p-values for comparing Net1, Net4, and Net5 with the stacked deep learning model are small, and the test is rejected under a significance level of 0.1, meaning that the stacked deep learning model performs significantly better than individual deep learning models in terms of accuracy.

While accuracy is a global measure of classifier performance, precision, recall, and F-measure are more comprehensive indicators of classifier errors. Precision measures the rate of true attacks among all the detected attacks, whereas recall measures the rate of true attacks among all the true attacks, the F-measure is a harmonic mean of precision and recall. Among all the methods, NetStack, XGBoost, RF, RpartCV, Rpart1SE, and C50Tree have precision values greater than 0.995, indicating a low number of false positives for these methods. On the other hand, RF, RpartCV, and Bagging show a recall value greater than 0.985, signifying a low number of false positives for these methods. Overall, RF, RpartCV, and Bagging have a F1-score greater than 0.990, which means that these three methods have the best capacity to lower both the false positive and false negative numbers.

Fig. 4 depicts the AUC values for all the classifiers. Similar as the F1-score, AUC measures the capability of a classifier to distinguish both attacks and normal states. RF, RpartCV, and Bagging have a AUC value greater than 0.991, which means that these three classifiers have a better capacity to discriminate attacks and normal states.

The precision, recall and F1-score for each type of cyber-attack are shown in Fig. 5. Overall, the ensemble methods, XGBoost, RF, and Bagging, demonstrate the best capability in discriminating each type of attacks, the precision, recall, and F1-score of the ensemble methods are all among the highest scores in all methods. Among the seven types of attacks, MSCI and NMRI are the most difficult attacks to detect, with the highest F1-score equal to 0.945 and 0.963, respectively, achieved both by the Bagging method. The MFCI and Reconnaissance attacks are the easiest to detect, with the highest F1-score equal to 1 achieved by multiple methods.

Now, to convincingly illustrate the effectiveness of the investigated cyber-attack detection methods, we compare the performance of the RF, XGBoost, and the designed stacked deep model (NetStack) based cyber-attack detectors with state-of-the-art (SOTA) techniques applied to the same data from the gas pipeline testbed. Numerous data-driven methods have been applied to address the problem of cyber-attack detection in the gas pipeline system, including K-means-Convolutional Autoencoder(CAE) [51], Bloom-LSTM [52], and Gaussian Mixture Model (GMM) [53]. Table 4 compares the achieved average accuracy of the investigated approaches (i.e., RF, XGBoost, and NetStack) with those of the SOTA detection methods.

The authors in [51] proposed an approach to detect anomalies in industrial systems by applying the k-means procedure and convolutional autoencoder (CAE) model. Specifically, k-means is constructed to model the normal behavior for each attribute at any time point. At the same time, the CAE evaluates the evolution of each attribute in consecutive instants of time. A data point is declared as normal if the predicted outputs of both modules are normal. This k-means-CAE anomaly detection scheme achieved an accuracy of 0.9553 and an F1-score of 0.8908 when applied to a gas pipeline dataset accuracy of 0.9659 and an F1-score of 0.9373 when applied to a water storage tank dataset. However, this approach requires building a k-means model for each attribute, which is time-consuming. In [52], a combined Bloom filter(BF)-LSMT anomaly detection approach is introduced by combining a BF-driven

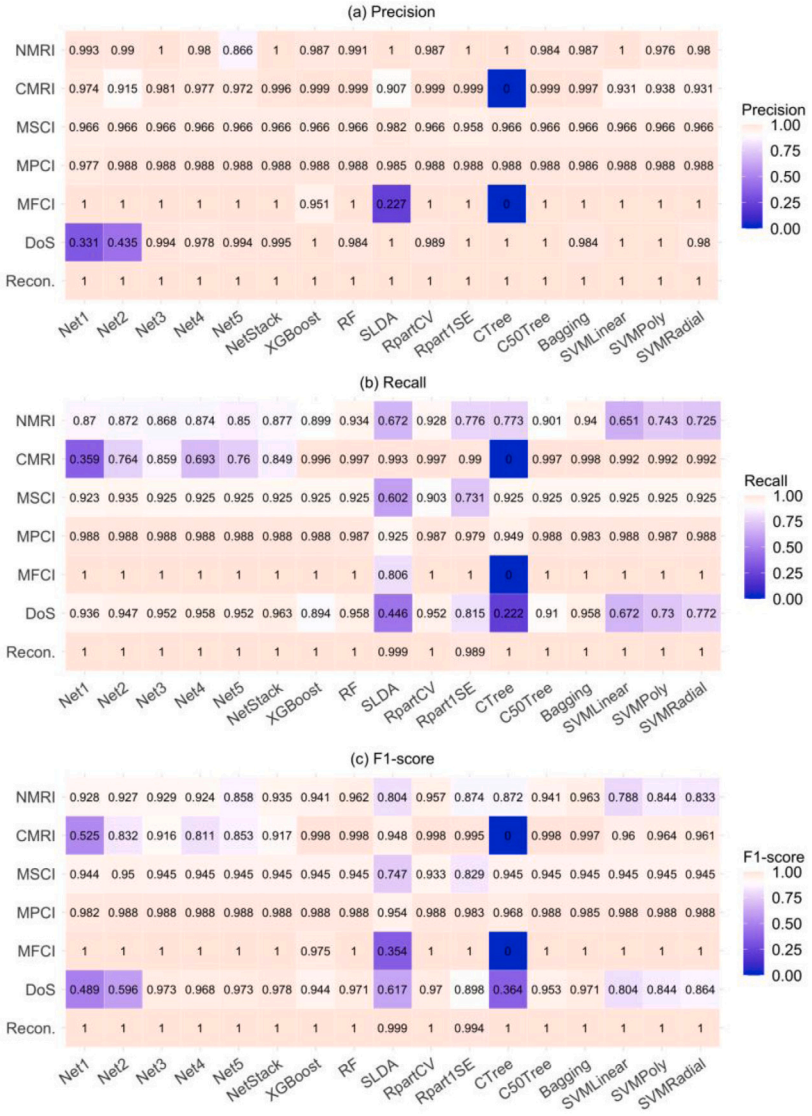

Fig. 5. The precision, recall and F1-score of each attack type for the gas pipeline dataset. (a) Precision, (b) Recall, and (c) F1-score.

anomaly detector and an LSTM network-driven detector. A BF is employed to detect anomalies within package content level, and a stacked LSTM-based softmax classifier is used to identify anomalies in package traffic given previously seen package traffic. Importantly, this approach learns normal evolution using anomaly-free data, and then it is applied to detect unseen attacks. This approach achieved a detection accuracy of 0.92 and an F1-score of 0.85 and showed better performance compared to BF, Bayesian Network (BN), Isolation Forest (IF), a Support Vector Data Description (SVDD) (Table 4). The authors in [54] have proposed a three-stage scheme to detect and identify intrusions for a gas pipeline system. At first, the detector discriminates normal data points from attacks. If the observation is declared an attack, it is classified into one of seven attack classes in the second stage. After that, another sub-attack classification is accomplished to find the sub-attack type. This three-stage-based intrusion detection scheme achieved an accuracy of 92.06%. In [53], four traditional anomaly detection methods, K-means, Naïve Bayesian (NB), Principal Component Analysis (PCA), and Gaussian Mixture Model (GMM), are applied to detect attacks in gas pipeline data. From Table 4, it can be seen that these traditional methods are not efficient in discriminating normal data points from attacks in gas pipeline data.

To broaden the analysis, we compared the performance of the considered attack detection techniques with the state-of-the-art methods in discriminating the individual attacks. It is crucial to differentiate between different detected attacks in industrial systems to effectively thwart these attacks. Specifically, accurate identification of attack types

Table 4
Comparison with the state-of-the-art methods based on the gas pipeline dataset.

Technique	Accuracy	Precision	Recall	F1-score
K-means-CAE [51]	0.9553	0.9543	0.8352	0.8908
Bloom-LSMT [52]	0.92	0.94	0.78	0.85
BF [52]	0.87	0.97	0.59	0.73
Isolation Forest [52]	0.70	0.51	0.13	0.20
SVDD [52]	0.76	0.95	0.21	0.34
K-means [53]	0.5680	0.8319	0.5728	0.6751
BN [52]	0.87	0.97	0.59	0.73
NB [53]	0.9036	0.8195	0.7692	0.8595
PCA [53]	0.1714	0.6472	0.2796	0.2710
GMM [53]	0.4516	0.7309	0.4416	0.5583
Three-stages [54]	92.06	0.920	0.921	–
GMM [55]	45.16	0.731	0.442	–
OCC-eSNN [30]	98.82	0.988	0.988	–
OCC-SVM [30]	97.98	0.980	0.980	–
NetStack	0.968	0.995	0.920	0.956
XGBoost	0.990	0.995	0.980	0.987
RF	0.993	0.995	0.987	0.991

Table 5
Discrimination performance (recall) comparison for each attack type with SOTA methods.

Model	NRMI	CMRI	MSCI	MPCI	MFCI	DOS	Reconnaissance
LSTM [52]	0.88	0.670	0.620	0.800	1	0.940	1
BF [52]	0.77	0.530	0.180	0.490	1	0.930	1
BN [52]	0.77	0.530	0.530	0.340	1	0.930	1
SVDD [52]	0.01	0.020	0.190	0.260	1	0.400	1
Kmeans [53]	0.184	0.201	0.728	0.664	0.515	1	0.753
NB [53]	0.810	0.837	0.727	0.669	0.519	0.792	0.503
Isolation Forest [52]	0.13	0.080	0.460	0.080	0	0.120	0.120
GMM [53]	0.31	0.330	0.660	0.640	0.320	0.150	0.720
PCA-SVD [53]	0.45	0.190	0.620	0.660	0.540	0.580	0.540
NetStack	0.877	0.849	0.925	0.988	1	0.963	1
XGBoost	0.899	0.996	0.925	0.988	1	0.894	1
RF	0.934	0.997	0.925	0.987	1	0.958	1

enables getting pertinent information about the nature of the detected attack, which helps to take an efficient countermeasure to handle these attacks. The Recall values of the different types of cyberattacks for the investigated models are listed in Table 5. The results shown in Table 5 indicate that the considered ensemble models (i.e., NetStack, XGBoost, and RF) dominate the state-of-the-art methods in discriminating the type of attacks in almost every scenario. Essentially, this could be attributed to the flexibility of ensemble models, which merge several individual weak models to achieve high classification accuracy. We also note that the RF model outperformed all the considered models for cyber-attacks identification. This could be attributed to its design based on bagging for reducing the variation in the classification by merging the output of several decision trees on distinct samples of the dataset. From Table 5, we noticed that the state-of-the-art methods are providing low classification performance for MSCI, CMRI, and MPCI attack types compared to the other attacks. This is mainly because of the noisiness of data related to these attacks since they are related to the physical processes that show generally noisy behavior [52]. The limited classification performances of the NB classier are mainly due to its strong assumption that two features are independent given the output class, which is often violated in real applications. Moreover, PCA and SVD cannot capture process nonlinearities and assume that data follow Gaussian distribution, making it unsuitable for cyber-attack detection in industrial systems. On the other hand, the three ensemble models (NetStack, XGBoost, and RF) provided significantly improved results than the other machine learning models in identifying these three attacks. The RF and XGBoost classifiers performed the best due to their robustness to noise and outliers [56]. Generally speaking, noisy measurements from physical-process-related variables could decrease the classification performance of machine learning methods to identify some related attacks. This difficulty could be mitigated by considering more training data to increase the sensitivity to physical-process-related variable changes when training the models.

4.2. Water storage tank systems

The second experiment is conducted using data from the storage tank testbeds from the Mississippi State University SCADA Laboratory [47]. This storage tank testbed is designed to mimic tanks of oil storage in petrochemical plants. Usually, this kind of storage tank, which stores oil coming from the sea, is employed to consistently supply the refinery with oil. The water storage system consists of two tanks, a pump to move water from the primary to the secondary tank via a swap valve, and a sensor that measures the primary tank's water level. Like the gas pipeline testbed, the supervisory control module includes an MTU, a set of RTUs, and HMI in which communications are guaranteed using a Modbus RTU serial interface. A schematic representation of the control system of the storage tank is shown in Fig. 6 [57]. The operator can manually or automatically control this system and turn off and

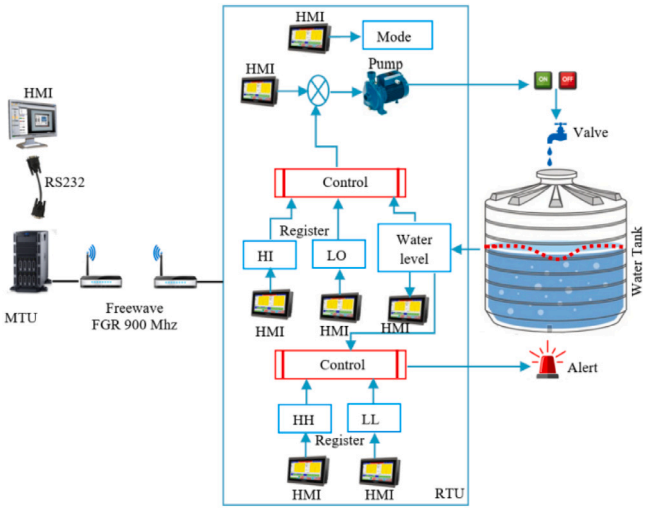

Fig. 6. Schematic representation of the water tank system.

on the pump to maintain the water level within the high and the low levels. An alarm is declared if the water level is outside the zone limited by the high and the low levels. Each input sample has 24 heterogeneous variables, such as the pump state, the manual pump setting, water level in the tank, and water levels. If the intruders can penetrate the control systems of a storage tank, cyber-attacks can render this system inoperable or cause significant financial and human life losses. Similar to the first dataset, the same 28 types of attacks have been introduced in this system to negatively disturb its normal operating conditions.

Table 6 quantifies the performance of the seventeen investigated classifiers based on the testing dataset. In terms of all metrics calculated, the XGBoost outperformed the other machine learning and deep learning models. It achieved an accuracy of 0.989 and an F1-score of 0.982. It is followed by RF and Bagging models with an accuracy of 0.984 and an F1-score of 0.973. We observe that ensemble learning models (i.e., XGBoost, RF, and Bagging) outperformed the other machine learning models and the deep learning model. Similar to the gas pipeline data, the water storage tank dataset has outliers. The water level measurements are huge (greater than 10^{15}) in about 0.2% of the times in normal states without attacks, whereas in most of the times, the water level measurements are within 0 and 100. The ensemble learning models, such as RF and XGBoost, are capable of discriminating attacks in the presence of outliers, which explains the superior performance of these models compared to support vector machines and neural networks. The stacked deep learning network

Table 6
Summary of classification metric based on testing the water storage tank dataset.

Method	Accuracy	Sensitivity	Specificity	AUC	Precision	Recall	F1-Score
Net1	0.929	0.786	0.986	0.886	0.957	0.786	0.863
Net2	0.920	0.797	0.969	0.883	0.911	0.797	0.850
Net3	0.900	0.806	0.947	0.876	0.856	0.806	0.831
Net4	0.892	0.803	0.928	0.866	0.815	0.803	0.809
Net5	0.907	0.795	0.952	0.873	0.868	0.795	0.830
NetStack	0.940	0.788	1.000	0.894	1.000	0.788	0.881
XGBoost	0.989	0.986	0.991	0.989	0.978	0.986	0.982
RF	0.984	0.975	0.989	0.982	0.971	0.975	0.973
SLDA	0.894	0.637	0.996	0.817	0.986	0.637	0.774
RpartCV	0.979	0.985	0.978	0.981	0.946	0.985	0.965
Rpart1SE	0.966	0.981	0.960	0.971	0.907	0.981	0.942
CTree	0.877	0.567	1	0.78	1	0.567	0.723
C50Tree	0.980	0.986	0.978	0.982	0.947	0.986	0.966
Bagging	0.984	0.974	0.989	0.981	0.972	0.974	0.973
SVMLinear	0.922	0.726	1.000	0.863	0.999	0.726	0.841
SVMPoly	0.941	0.793	1.000	0.897	1.000	0.793	0.885
SVMRadial	0.935	0.775	0.999	0.887	0.997	0.775	0.872

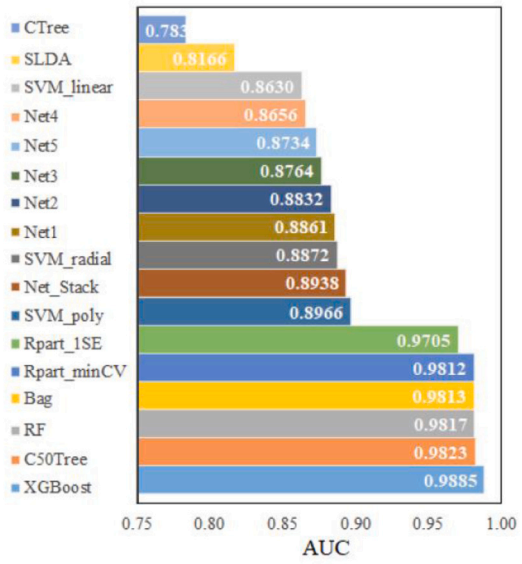

Fig. 7. Barplot of AUC values obtained by each detection method.

Table 7
The p-values testing the hypothesis that the stacked deep learning method is superior to the individual network for the water storage tank data.

Net1	Net2	Net3	Net4	Net5
0.100	0.072	0.131	0.030	0.045

can detect the different cyber-attacks that occurred in the water tank system with an accuracy of 0.940 and an F1-score of 0.881. Fig. 7 displays the barplot of AUC values of the considered models to visually support comparing the obtained results in Table 6. It can be observed that results testify the superior discrimination capacity of the ensemble learning models (i.e., XGBoost, RF, and Bagging) compared to the other models, including deep learning models. Moreover, results in this experiment revealed that the XGBoost has better detection performance by reaching more accurate and reliable classification results compared to the other investigated models.

Similar to the performance for the gas pipeline data, the stacked deep learning model shows a better performance than single deep learning models. The p-values for testing whether the difference in accuracy is significant using a paired t-test is listed in Table 7. Based on a significance level of 0.1, the accuracy of the stacked deep learning model is significantly better than that of Net2, Net4 and Net5.

Table 8
Comparison with the state-of-the-art methods based on the water storage tank dataset.

Technique	Accuracy	Precision	Recall	F1-score
OCC-eSNN [30]	0.9808	0.981	0.981	0.994
OCC-SVM [30]	0.9801	0.980	0.980	0.995
OCC-CDCPE [30]	0.9675	0.975	0.975	0.980
K-means-CAE [51]	0.9659	0.9300	0.9447	0.9373
HMM-based [29]	0.9336	0.9869	0.7775	0.8686
NetStack	0.940	1.000	0.788	0.881
XGBoost	0.989	0.978	0.986	0.982
RF	0.984	0.971	0.975	0.973

The precision, recall and F1-score for each type of cyber-attack are shown in Fig. 8. Similar to the gas pipeline data, the ensemble methods, XGBoost, RF, and Bagging, demonstrate the best capability in discriminating each type of attacks, the precision, recall, and F1-score of the ensemble methods are all among the highest scores in all methods. Among the seven types of attacks, CMRI is the most difficult one to detect, with the highest F1-score equal to 0.919 achieved by the XGBoost method. The MFCI, DoS, and Reconnaissance attacks are the easiest to detect, with the highest F1-score equal to 1 achieved by multiple methods.

Here, we compare the effectiveness of the investigated approaches with the SOTA methods. Several studies using the same water storage tank dataset are reported in the literature, as listed in Table 8 . Results in Table 8 indicate that ensemble learning models (i.e., Xgboost and RF) display higher detection performance compared to the other models. Specifically, the overall accuracy values obtained by XGBoost and RF are 0.989 and 0.984, respectively. The one-class detection methods, including OCC-eSNN, OCC-SVM, OCC-CDCPE [30] exhibit the closest performance to ensemble models (i.e., XGBoost and RF) by obtaining a detection accuracy of 98.08%, 98.01%, 96.75%, respectively. In [30], the detection approaches are constructed using only anomaly-free data and then employed to detect any deviation from the normal data. However, their ability to detect attacks in this application is still lower than ensemble models. For the study reported in [51], detection results using the combined K-means-CAE approach are relatively below than one-class-based and ensemble models by reaching an accuracy of 96.59%. The HMM-based detector [29] have relatively lower performance compared to the other models (an accuracy of 93.36%) mainly because it is not efficient in dealing with such complicated multivariate data. Besides, the detection accuracy obtained by using the NetStack approach is 94%, which means that the cyber-attacks detection has not been improved even by using powerful deep learning models (Table 8). Overall, the XGBoost algorithm outperformed all the other methods by reaching a satisfactory discrimination performance with an accuracy of 0.989.

Fig. 9 illustrates the barplot of averaged AUC values of those shown in Tables 2 and 6. Overall, results demonstrated that ensemble learning-based detectors could effectively identify cyber-attacks in gas pipeline and water tank testbeds. Specifically, the detection result obtained using the ensemble models has been satisfying because almost all attacks are correctly separated from the normal situations (AUC values greater than 0.98). We could say that the XGBoost classifier provides the highest detection accuracy among all of the sixteen models.

5. Conclusion

With the extended use of information and communication technologies in SCADA systems, cyberattacks against industrial systems have significantly increased. The task of conventional IDSs becomes very challenging due to the several vulnerabilities in SCADA systems and the variety of cyberattacks. Miss-detection of cyberattacks in critical systems can lead to severe economic and safety consequences. This work explores the detection potential of machine learning and deep learning to support IDSs in detecting cyberattacks. At first, this study

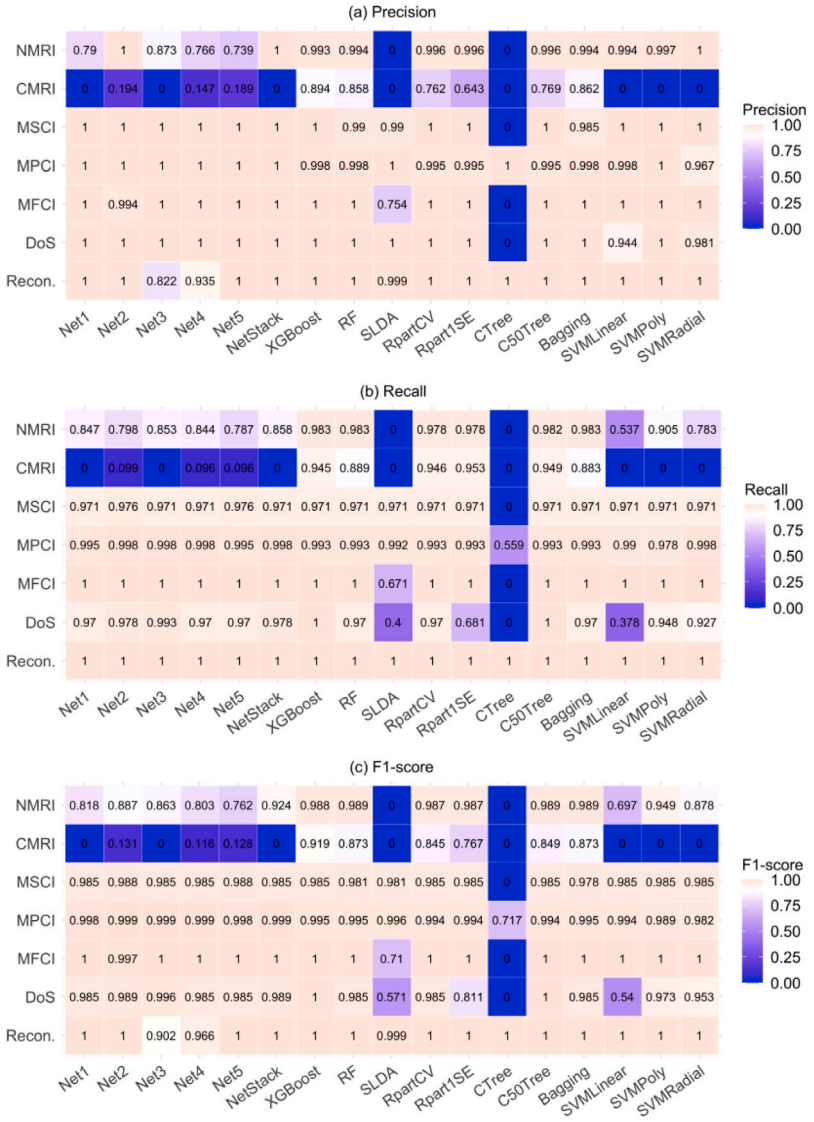

Fig. 8. The precision, recall and F1-score of each attack type for the water storage tank dataset. (a) Precision, (b) Recall, and (c) F1-score.

investigated the feasibility of a stacked deep learning scheme to uncover malicious attacks in SCADA systems. Moreover, the performances of eleven machine learning models, including the Xtreme Gradient Boosting (XGBoost), Random forest, Bagging, support vector machines with different kernels, classification tree pruned by the minimum cross-validation and by 1-standard error rule, linear discriminate analysis, conditional inference tree, and the C5.0 tree, are investigated in detecting attacks in SCADA systems. To this end, measurements from a gas pipeline and water storage tank testbeds are used to verify the performance of the considered machine learning and deep learning models. This comparison study generates that the XGBoost-anomaly detection approach achieved superior detection performance than all other investigated models. The RF scheme also achieves comparable performances. Results suggested that using ensemble learning models (XGBoost and RF) and exploiting their capacity to provide precise detection significantly reducing the number of missed detections and false alarms to reach high cyber-attack detection performance.

Cyber-attack data constantly exhibits outliers, e.g., a huge value for some features, even when the system is operating in normal conditions. The success behind ensemble methods such as random forest and XGBoost is largely due to their robustness to outliers. In contrast, stacked deep learning methods are more sensitive to outliers, because outlier-sensitive linear transformations are fundamental operations in neural networks. Although XGboost and RF models have superior detection performance than the stacked deep learning model in this work; it is hard to say that ensemble learning models dominate the deep learning models. This is because we expect that deep learning can achieve improved performance once larger datasets are used. However, how to design and implement a stacked deep learning model for cyber-attack detection that is robust to outliers is a challenge that we leave to future research.

In this study, the ensemble learning models, particularly XGBoost and RF, showed promising performance in discriminating normal data from attacks in the laboratory-scale SCADA systems: gas pipeline and water storage tank systems. However, the principal disadvantage of these supervised classifiers is that prior knowledge of different types of attacks is needed to provide an efficient detection. These machine learning models are constructed using labeled training data, while obtaining this prior knowledge on all the existing types of attacks is not an easy task, particularly with the generation of advanced and sophisticated attacks every day. An interesting direction for future work is the design of unsupervised anomaly detection approaches by amalgamating unsupervised deep learning models as features extractors, such as deep generative models, with the sensitivity of statistical monitoring charts, such as Generalized Likelihood Ratio Test [58].

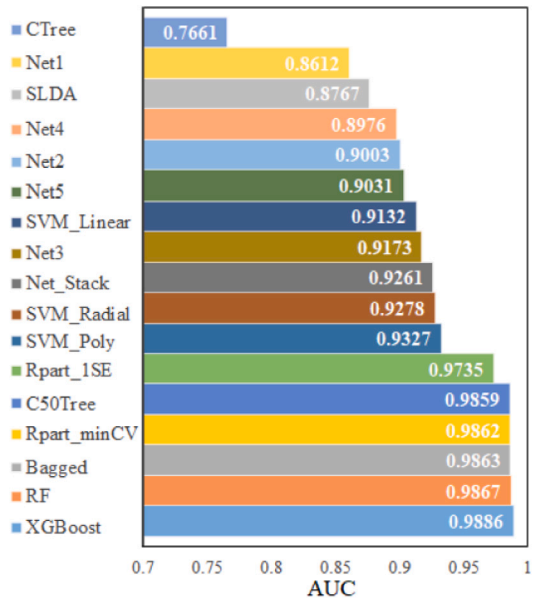

Fig. 9. Barplot of averaged AUC values obtained by each detection method.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.ijcip.2022.100542.

Acknowledgments

Wu Wang's research is supported by the Fundamental Research Funds for the Central Universities, China and the Research Funds of Renmin University of China. This publication is based upon work supported by King Abdullah University of Science and Technology (KAUST), Saudi Arabia, Office of Sponsored Research (OSR) under Award No: OSR-2019-CRG7-3800.

References

[1] S.V.B. Rakas, M.D. Stojanović, J.D. Marković-Petrović, A review of research work on network-based scada intrusion detection systems, IEEE Access 8 (2020) 93083–93108.
[2] D.U. Case, Analysis of the Cyber Attack on the Ukrainian Power Grid, Vol. 388, Electricity Information Sharing and Analysis Center (E-ISAC), 2016.
[3] S. Adepu, V.R. Palleti, G. Mishra, A. Mathur, Investigation of cyber attacks on a water distribution system, in: International Conference on Applied Cryptography and Network Security, Springer, 2020, pp. 274–291.
[4] I. Figueiredo, P. Esteves, P. Cabrita, Water wise–a digital water solution for smart cities and water management entities, Procedia Comput. Sci. 181 (2021) 897–904.
[5] B. Bouyeddou, F. Harrou, Y. Sun, Detecting cyber-attacks in modern power systems using an unsupervised monitoring technique, in: 2021 IEEE 3rd Eurasia Conference on Biomedical Engineering, Healthcare and Sustainability, ECBIOS, IEEE, 2021, pp. 259–263.
[6] B. Bouyeddou, B. Kadri, F. Harrou, Y. Sun, DDOS-attacks detection using an efficient measurement-based statistical mechanism, Eng. Sci. Technol. Int. J. 23 (4) (2020) 870–878.
[7] A. Abou el Kalam, Securing SCADA and critical industrial systems: From needs to security mechanisms, Int. J. Crit. Infrastruct. Prot. 32 (2021) 100394.
[8] S. Veronica, S. Geneva, S. Arlette, Cyberattack forces major US fuel pipeline to shut down, [Online]. Available: https://edition.cnn.com/2021/05/08/politics/colonial-pipeline-cybersecurity-attack/index.html.
[9] A.A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, S. Sastry, Attacks against process control systems: risk assessment, detection, and response, in: Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, 2011, pp. 355–366.
[10] S. Gorman, Electricity grid in US penetrated by spies, Wall Str. J. 8 (2009).
[11] S. Zhioua, The middle east under malware attack dissecting cyber weapons, in: 2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops, IEEE, 2013, pp. 11–16.
[12] L. Visaggio, Hacking the infrastructure Cyber-attack, physical damage.
[13] C.-C. Sun, A. Hahn, C.-C. Liu, Cyber security of a power grid: State-of-the-art, Int. J. Electr. Power Energy Syst. 99 (2018) 45–56.
[14] A.P. Mathur, N.O. Tippenhauer, SWaT: A water treatment testbed for research and training on ICS security, in: 2016 International Workshop on Cyber-Physical Systems for Smart Water Networks, CySWater, IEEE, 2016, pp. 31–36.
[15] L. Faramondi, F. Flammini, S. Guarino, R. Setola, A hardware-in-the-loop water distribution testbed dataset for cyber-physical security testing, IEEE Access 9 (2021) 122385–122396.
[16] G. Poojitha, K.N. Kumar, P.J. Reddy, Intrusion detection using artificial neural network, in: 2010 Second International Conference on Computing, Communication and Networking Technologies, IEEE, 2010, pp. 1–7.
[17] O. Linda, T. Vollmer, M. Manic, Neural network based intrusion detection system for critical infrastructures, in: 2009 International Joint Conference on Neural Networks, IEEE, 2009, pp. 1827–1834.
[18] H. Yang, L. Cheng, M.C. Chuah, Deep-learning-based network intrusion detection for SCADA systems, in: 2019 IEEE Conference on Communications and Network Security, CNS, IEEE, 2019, pp. 1–7.
[19] R.R.R. Barbosa, R. Sadre, A. Pras, Flow whitelisting in SCADA networks, Int. J. Crit. Infrastruct. Prot. 6 (3–4) (2013) 150–158.
[20] G. Chaojun, P. Jirutitijaroen, M. Motani, Detecting false data injection attacks in ac state estimation, IEEE Trans. Smart Grid 6 (5) (2015) 2476–2483.
[21] L.A. Maglaras, J. Jiang, T. Cruz, Integrated OCSVM mechanism for intrusion detection in SCADA systems, Electron. Lett. 50 (25) (2014) 1935–1936.
[22] A. Fahad, Z. Tari, A. Almalawi, A. Goscinski, I. Khalil, A. Mahmood, PPFSCADA: Privacy preserving framework for SCADA data publishing, Future Gener. Comput. Syst. 37 (2014) 496–511.
[23] S. Shitharth, et al., An enhanced optimization based algorithm for intrusion detection in SCADA network, Comput. Secur. 70 (2017) 16–26.
[24] D. Xue, X. Jing, H. Liu, Detection of false data injection attacks in smart grid utilizing ELM-based OCON framework, IEEE Access 7 (2019) 31762–31773.
[25] A.A.Z. Khan, Misuse intrusion detection using machine learning for gas pipeline SCADA networks, in: Proceedings of the International Conference on Security and Management, SAM, The Steering Committee of The World Congress in Computer Science, Computer, 2019, pp. 84–90.
[26] H. Wang, T. Lu, X. Dong, P. Li, M. Xie, Hierarchical online intrusion detection for scada networks, 2016, arXiv preprint arXiv:1611.09418.
[27] N. Erez, A. Wool, Control variable classification, modeling and anomaly detection in modbus/TCP SCADA systems, Int. J. Crit. Infrastruct. Prot. 10 (2015) 59–70.
[28] D. Yang, A. Usynin, J.W. Hines, Anomaly-based intrusion detection for SCADA systems, in: 5th Intl. Topical Meeting on Nuclear Plant Instrumentation, Control and Human Machine Interface Technologies, Npic&Hmit 05, 2006, pp. 12–16.
[29] K. Stefanidis, A.G. Voyiatzis, An HMM-based anomaly detection approach for SCADA systems, in: IFIP International Conference on Information Security Theory and Practice, Springer, 2016, pp. 85–99.
[30] K. Demertzis, L. Iliadis, S. Spartalis, A spiking one-class anomaly detection framework for cyber-security on industrial control systems, in: International Conference on Engineering Applications of Neural Networks, Springer, 2017, pp. 122–134.
[31] L. Breiman, J. Friedman, C.J. Stone, R.A. Olshen, Classification and Regression Trees, CRC Press, 1984.
[32] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.
[33] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[34] M. Fernández-Delgado, E. Cernadas, S. Barro, D. Amorim, Do we need hundreds of classifiers to solve real world classification problems? J. Mach. Learn. Res. 15 (1) (2014) 3133–3181.
[35] A.Z. Marvin Wright, ranger: A fast implementation of random forests for high dimensional data in C++ and R, J. Stat. Softw. 77 (1) (2017).
[36] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785–794.
[37] R.E. Schapire, Y. Freund, P. Bartlett, W.S. Lee, et al., Boosting the margin: A new explanation for the effectiveness of voting methods, Ann. Statist. 26 (5) (1998) 1651–1686.
[38] J. Friedman, T. Hastie, R. Tibshirani, et al., Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors), Ann. Statist. 28 (2) (2000) 337–407.
[39] J.H. Friedman, Greedy function approximation: a gradient boosting machine, Ann. Statist. (2001) 1189–1232.
[40] G. Hinton, L. Deng, D. Yu, G.E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, et al., Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups, IEEE Signal Process. Mag. 29 (6) (2012) 82–97.
[41] R. Collobert, J. Weston, A unified architecture for natural language processing: Deep neural networks with multitask learning, in: Proceedings of the 25th International Conference on Machine Learning, 2008, pp. 160–167.

[42] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: Towards real-time object detection with region proposal networks, in: Advances in Neural Information Processing Systems, 2015, pp. 91–99.
[43] D.H. Wolpert, Stacked generalization, Neural Netw. 5 (2) (1992) 241–259.
[44] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2016, pp. 770–778.
[45] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
[46] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: International Conference on Machine Learning, 2015, pp. 448–456.
[47] T. Morris, A. Srivastava, B. Reaves, W. Gao, K. Pavurapu, R. Reddi, A control system testbed to validate critical infrastructure protection concepts, Int. J. Crit. Infrastruct. Prot. 4 (2) (2011) 88–103.
[48] T. Morris, W. Gao, Industrial control system traffic data sets for intrusion detection research, in: International Conference on Critical Infrastructure Protection, Springer, 2014, pp. 65–78.
[49] T.H. Morris, W. Gao, Industrial control system cyber attacks, in: 1st International Symposium for ICS & SCADA Cyber Security Research 2013, ICS-CSR 2013 1, 2013, pp. 22–29.
[50] W. Gao, T.H. Morris, On cyber attacks and signature based intrusion detection for modbus based industrial control systems, J. Digit. Forensics Secur. Law 9 (1) (2014) 3.
[51] C.-P. Chang, W.-C. Hsu, I.-E. Liao, Anomaly detection for industrial control systems using k-means and convolutional autoencoder, in: 2019 International Conference on Software, Telecommunications and Computer Networks, SoftCOM, IEEE, 2019, pp. 1–6.
[52] C. Feng, T. Li, D. Chana, Multi-level anomaly detection in industrial control systems via package signatures and LSTM networks, in: 2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN, IEEE, 2017, pp. 261–272.
[53] S.N. Shirazi, A. Gouglidis, K.N. Syeda, S. Simpson, A. Mauthe, I.M. Stephanakis, D. Hutchison, Evaluation of anomaly detection techniques for scada communication resilience, in: 2016 Resilience Week, RWS, IEEE, 2016, pp. 140–145.
[54] A.A.Z. Khan, G. Serpen, Intrusion detection and identification system design and performance evaluation for industrial SCADA networks, 2020, arXiv preprint arXiv:2012.09707.
[55] R.L. Perez, F. Adamsky, R. Soua, T. Engel, Machine learning for reliable network attack detection in SCADA systems, in: 2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering, TrustCom/BigDataSE, IEEE, 2018, pp. 633–638.
[56] S. Ishii, D. Ljunggren, A comparative analysis of robustness to noise in machine learning classifiers, 2021.
[57] W. Gao, T. Morris, B. Reaves, D. Richey, On SCADA control system command and response injection and intrusion detection, in: 2010 ECrime Researchers Summit, IEEE, 2010, pp. 1–9.
[58] F. Harrou, Y. Sun, A.S. Hering, M. Madakyaru, A. Dairi, Unsupervised deep learning-based process monitoring methods, in: Statistical Process Monitoring using Advanced Data-Driven and Deep Learning Approaches, Elsevier, 2021, pp. 193–223.