Toward Detection and Attribution of Cyber-Attacks in IoT-Enabled Cyber-Physical Systems

Article *in* IEEE Internet of Things Journal · March 2021

DOI: 10.1109/JIOT.2021.3067667

4 authors:

Amir Namavar Jahromi
University of Guelph
13 PUBLICATIONS **112** CITATIONS
SEE PROFILE

Hadis Karimipour
University of Guelph
105 PUBLICATIONS **1,422** CITATIONS
SEE PROFILE

Ali Dehghantanha
University of Guelph
232 PUBLICATIONS **6,061** CITATIONS
SEE PROFILE

Kim-Kwang Raymond Choo
University of Texas at San Antonio
1,035 PUBLICATIONS **25,892** CITATIONS
SEE PROFILE

Toward Detection and Attribution of Cyber-Attacks in IoT-enabled Cyber-physical Systems

Amir Namavar Jahromi, Hadis Karimipour, *Senior Member, IEEE,* Ali Dehghantanha, *Senior Member, IEEE,* and Kim-Kwang Raymond Choo, *Senior Member, IEEE*

***Abstract*—Securing Internet of Things (IoT)-enabled cyber-physical systems (CPS) can be challenging, as security solutions developed for general information / operational technology (IT / OT) systems may not be as effective in a CPS setting. Thus, this paper presents a two-level ensemble attack detection and attribution framework designed for CPS, and more specifically in an industrial control system (ICS). At the first level, a decision tree combined with a novel ensemble deep representation-learning model is developed for detecting attacks imbalanced ICS environments. At the second level, an ensemble deep neural network is designed for attack attribution. The proposed model is evaluated using real-world datasets in gas pipeline and water treatment system. Findings demonstrate that the proposed model outperforms other competing approaches with similar computational complexity.**

***Index Terms*—Cyber-attacks, Deep representation learning, Cyber threat detection, Cyber threat attribution, Industrial Control System, ICS, Cyber-physical systems, Industrial Internet of Things (IIoT)**

I. Introduction

Internet of Things (IoT) devices are increasingly integrated in cyber-physical systems (CPS), including in critical infrastructure sectors such as dams and utility plants. In these settings, IoT devices (also referred to as Industrial IoT or IIoT) are often part of an Industrial Control System (ICS), tasked with the reliable operation of the infrastructure. ICS can be broadly defined to include supervisory control and data acquisition (SCADA) systems, distributed control systems (DCS), and systems that comprise programmable logic controllers (PLC) and Modbus protocols.

The connection between ICS or IIoT-based systems with public networks, however, increases their attack surfaces and risks of being targeted by cyber criminals. One high-profile example is the Stuxnet campaign, which reportedly targeted Iranian centrifuges for nuclear enrichment in 2010, causing severe damage to the equipment [1], [2]. Another example is that of the incident targeting a pump that resulted in the failure of an Illinois water plant in 2011 [3]. BlackEnergy3 was another campaign that targeted Ukraine power grids in 2015, resulting in power outage that affected approximately 230,000 people [4]. In April 2018, there were also reports of successful cyber-attacks affecting three U.S. gas pipeline firms, and resulted in the shutdown of electronic customer communication systems for several days [1]. Although security solutions developed for information technology (IT) and operational technology (OT) systems are relatively mature, they may not be directly applicable to ICSs. For example, this could be the case due to the tight integration between the controlled physical environment and the cyber systems.

Therefore, system-level security methods are necessary to analyze physical behaviour and maintain system operation availability [1]. ICS security goals are prioritized in the order of availability, integrity, and confidentiality, unlike most IT/OT systems (generally prioritized in the order of confidentiality, integrity, and availability) [5]. Due to close coupling between variables of the feedback control loop and physical processes, (successful) cyber-attacks on ICS can result in severe and potentially fatal consequences for the society and our environment. This reinforces the importance of designing extremely robust safety and security measurements to detect and prevent intrusions targeting ICS [1].

Popular attack detection and attribution approaches include those based on signatures and anomalies. To mitigate the known limitations in both signature-based and anomaly-based detection and attribution approaches, there have been attempts to introduce hybrid-based approaches [6]. Although hybrid-based approaches are effective at detecting unusual activates, they are not reliable due to frequent network upgrades, resulting in different Intrusion Detection System (IDS) typologies [7]. Beyond this, conventional attack detection and attribution techniques mainly rely on network metadata analysis (e.g. IP addresses, transmission ports, traffic duration, and packet intervals). Therefore, there has been renewed interest in utilizing attack detection and attribution solutions based on Machine Learning (ML) or Deep Neural Networks (DNN) in recent times.

In addition, attack detection approaches can be categorized into network-based or host-based approaches. Supervised clustering, single-class or multi-class Support Vector Machine (SVM), fuzzy logic, Artificial Neural Network (ANN), and DNN are commonly used techniques for attack detection in network traffic. These techniques analyze real-time traffic data to detect malicious attacks in a timely manner. However, attack detection that considers only network and host data may fail to detect sophisticated attacks or insider attacks.

Amir Namavar Jahromi and Hadis Karimipour are with the School of Engineering, University of Guelph, Ontario, Canada (email: anamavar@uoguelph.ca and hkarimi@uoguelph.ca).

Ali Dehghantanha is with the School of Computer Science, University of Guelph, Ontario, Canada (email: adehghan@uoguelph.ca)

Kim-Kwang Raymond Choo is with the Department of Information Systems and Cyber Security and the Department of Electrical and Computer Engineering, University of Texas at San Antonio (UTSA), San Antonio, TX 78249, USA. He also has courtesy appointments with UTSA's Department of Electrical and Computer Engineering and Department of Computer Science, and UniSA STEM at the University of South Australia, Adelaide, SA 5095, Australia. (email: raymond.choo@fulbrightmail.org)

Unsupervised models that incorporate process/physical data can complement a system's monitoring since they do not rely on detailed knowledge of the cyber-threats. In general, a sophisticated attacker with sufficient knowledge and time, such as a nation state advanced persistent threat actor, can potentially circumvent robust security solutions. Furthermore, most of the existing approaches ignore the imbalanced property of ICS data by modeling only a system's normal behavior and reporting deviations from normal behavior as anomalies. This is, perhaps, due to limited attack samples in existing datasets and real-world scenarios. Although using majority class samples is a good solution to avoid issues due to imbalanced datasets, the trained model will have no view of the attack samples' patterns. In other words, such an approach fails to detect unseen attacks and suffers from a high false-positive rate [8]. Thus, there have been attempts to utilize DL approaches, for example, to facilitate automated feature (representation) learning to model complex concepts from simpler ones [9] without depending on human-crafted features [10].

Motivated by the above observations, this paper presents our proposed novel two-stage ensemble deep learning-based attack detection and attack attribution framework for imbalanced ICS datasets. In the first stage, an ensemble representation learning model combined with a Decision Tree (DT) is designed to detect attacks in an imbalanced environment. Once the attack is detected, several one-vs-all classifiers will ensemble together to form a larger DNN to classify the attack attributes with a confidence interval during the second stage. Moreover, the proposed framework is capable of detecting unseen attack samples. A summary of our approach in this study is as follows:

1) We develop a novel two-phase ensemble ICS attack detection method capable of detecting both previously seen and unseen attacks. We will also demonstrate that the proposed method outperforms other competing approaches in terms of accuracy and f-measure. The proposed deep representation learning results in this method being robust to imbalanced data.
2) We propose a novel self-tuning two-phase attack attribution method that ensembles several deep one-vs-all classifiers using a DNN architecture for reducing false alarm rates. The proposed method can accurately attribute attacks with high similarity. This is the first ML-based attack attribution method in ICS/IIoT at the time of this research.
3) We analyze the computational complexity of the proposed attack detection and attack attribution framework, demonstrating that despite its superior performance, its computational complexity is similar to that of other DNN-based methods in the literature.

The rest of the paper will be organized as follows. Section II will introduce the relevant background and related literature. Section III will describe the proposed framework, followed by the experimental setup in Section IV. In Section V, the evaluation findings based on two real-world ICS datasets demonstrate that the proposed framework outperforms several other systems. Finally, Section VI concludes this paper.

II. Related Work

ML-based attack detection techniques are generally designed to detect moving targets that constantly evolve by learning new vulnerabilities and not relying on known attack signatures or normal network patterns [6]. We will now discuss the related literature as follows.

A. Conventional Machine Learning

In [11], ML algorithms, such as K-Nearest Neighbor (KNN), Random Forest (RF), DT, Logistic Regression (LR), ANN, Naïve Bayes (NB), and SVM were compared in terms of their effectiveness in detecting backdoor, command, and SQL injection attacks in water storage systems. The comparative summary suggested that the RF algorithm has the best attack detection, with a recall of 0.9744; the ANN is the fifth-best algorithm, with a recall of 0.8718; and the LR is the worst-performing algorithm, with a recall of 0.4744. The authors also reported that the ANN could not detect 12.82% of the attacks and considered 0.03% of the normal samples to be attacks. In addition, LR, SVM, and KNN considered many attack samples as normal samples, and these ML algorithms are sensitive to imbalanced data. In other words, they are not suitable for attack detection in ICS. In [12], the authors presented a KNN algorithm to detect cyber-attacks on gas pipelines. To minimize the effect of using an imbalanced dataset in the algorithm, they performed oversampling on the dataset to achieve balance. Using the KNN on the balanced dataset, they reported an accuracy of 97%, a precision of 0.98, a recall of 0.92, and an f-measure of 0.95. In [13], the authors presented a Logical Analysis of Data (LAD) method to extract patterns/rules from the sensor data and use these patterns/rules to design a two-step anomaly detection system. In the first step, a system is classified as stable or unstable, and in the second one, the presence of an attack is determined. They compared the performance of the proposed LAD method with the DNN, SVM, and CNN methods. Based on these experiments, the DNN outperformed the LAD method in the precision metric; however, the LAD performed better in recall and f-measure.

B. Deep Learning

In [14], the authors used the DNN algorithm to detect false data injection attacks in power systems. Findings of their evaluation using two datasets suggested 91.80% accuracy. In [15], the authors proposed an autoencoder-based method to detect false data injection attacks and clean them using denoising autoencoders. Their experiments showed that these methods outperformed the SVM-based method. To handle the effect of imbalanced data on the algorithm, they ignored attack data in training the autoencoder. In [16], the authors presented a technique based on Extreme Learning Machine (ELM) for attack detection in CPS. To address the imbalanced challenge of neural networks, training was conducted using only normal data. Based on these experiments, the proposed ELM-based method outperformed the SVM attack detection method.

Despite promising results in both conventional ML and deep learning-based techniques, most existing ML algorithms suffer from the curse of dimensionality due to the large data volume generated in real-world ICS. Therefore, feature engineering must reduce the number of features or generate a new representation of the features to reduce computational overhead. Moreover, an imbalanced dataset of the ICS is another challenge that should be considered. Researchers have attempted to resolve this issue using oversampling/undersampling, as well as ignoring attack samples and building algorithms using normal samples.

Attack attribution seeks to answer the question of "What kind of attack was it?" and this is generally more challenging to answer in ICS than in typical IT/OT systems due to the different network structures, industry-specific protocols, and so forth [17], [18]. While there have been a small number of ML-based malware attack attributions [19], [20], designing robust and effective ML-based attack attribution for ICS and IIoT systems appears to be understudied. Thus, this paper proposes a two-stage ensemble deep learning-based attack detection and attack attribution framework for ICS. Our approach incorporates both process and physical data to solve the imbalanced data problem without subsampling or oversampling. The proposed framework utilizes an unsupervised ensemble of learned representations from normal and attack instances for attack detection. Next, using an ensemble of several one-vs-all classifiers trained on each attack attribute, it forms a two-part DNN to attribute the samples into their corresponding attack attributes.

III. The Proposed Framework

Figure 1 shows the architecture of the proposed framework. In this framework, the attack detection method detects the attacks by analyzing the ICS input features using the combination of ensembled unsupervised DNNs and a decision tree. If an attack is detected, the sample is passed to several DNNs for detailed analysis. If the attacks were previously unseen/unknown, the unseen attack detection module would detect it and label it as an unseen attack. This will be passed on for detailed security analysis. Otherwise, the attack attribution method detects the attribute of the attack.

A. Proposed Ensemble Attack Detection Method

The proposed attack detection consists of two phases, namely representation learning and detection phase. Using a conventional unsupervised DNN on an imbalanced dataset yielded a DNN model that mainly learned majority class patterns and missed minority class characteristics. Most researchers have tried to address this challenge by generating new samples or removing certain samples to make the dataset balanced and then passing the data to a DNN. However, in ICS/IIoT security applications, generating or removing samples are not reasonable solutions. Due to the ICS/IIoT systems' sensitivity, generated samples should be validated in a real network, which is impossible since the generated attack samples may be harmful to the network and cause severe impacts on the environment or human life. In addition, validation of the generated samples is time-consuming. Moreover, removing the normal data from a dataset is not the right solution since the number of attack samples in ICS/IIoT datasets is usually less than 10% of the dataset, and most of the dataset knowledge is discarded by removing 80% of the dataset.

To avoid the above mentioned problems in handling imbalanced datasets, this study proposed a new deep representation learning method to make the DNN able to handle imbalanced datasets without changing, generating, or removing samples. This model consisted of two unsupervised stacked autoencoders, each responsible for finding patterns from one class. Since each model tries to extract abstract patterns of one class without considering another, the output of that model represented its inputs well. The stacked autoencoders had three decoders and encoders with input and final representation layers. The encoder layers mapped the input representation to a higher, 800-dimensional space, a 400-dimensional space, and the final 16-dimensional space. Equations 1 shows the encoder function of an autoencoder. The decoder layers did the opposite and tried to reconstruct the input representation by starting from the 16-dimensional new representation and mapping it to the 400-dimensional, 800-dimensional, and input representations. Equations 2 shows the decoder function of an autoencoder. These hyperparameters were selected using trial-and-error to have the best performance in f-measure with the lowest architectural complexity.

$$h_i = \sigma(w_i x_i + b_i) \tag{1}$$

In the above equation, σ denotes an activation function, w is the weight matrix of the encoder, x is a vector of sample features, b is encoder's bias, h is the encoded representation, and $i \in \{Normal, Attack\}$.

$$\hat{x}_i = \sigma'(w'_i h_i + b'_i) \tag{2}$$

In the above equation, σ' is the decoder's activation function, w' is the weight matrix of the decoder, h is the encoded representation, b' is decoder's bias, $\hat{x}$ is the reconstruction of input x, and $i \in \{Normal, Attack\}$.

Each autoencoder trained individually using the loss function indicated in Equation 3.

$$\mathcal{L}(x, \hat{x}) = ||x - \hat{x}||^2 = ||x - \sigma'(w'(wx + b) + b')||^2 \tag{3}$$

In the above equation, $\mathcal{L}(x, \hat{x})$ denotes the loss between the input x and its reconstruction $\hat{x}$.

After training the autoencoders, all observations were passed through both autoencoders, and the final representations were fused to form a super-vector for each instance to build a new dataset.

$$X_{new} = [H_{normal}, H_{attack}] \tag{4}$$

In the above equation, X_{new} is the new dataset consists of a super-vector of the learned representations from normal and attack autoencoder models for each sample. The H_{normal} is a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3067667, IEEE Internet of Things Journal

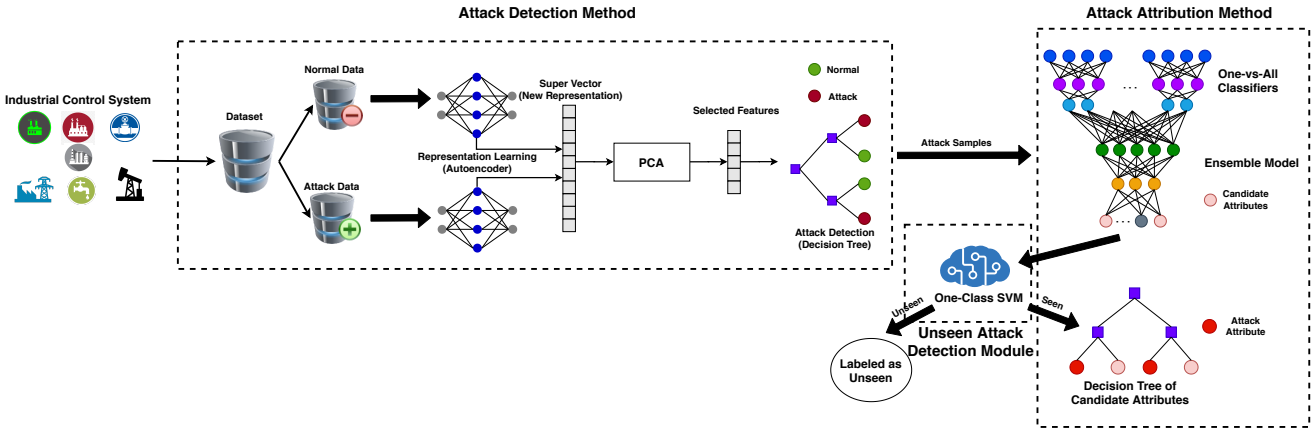

Fig. 1. Proposed attack detection and attribution framework

matrix of h_{normal} which is part of the features show how the sample x can represent a normal sample while the H_{attack} is a matrix of h_{attack} which shows how sample x can represent an attack sample.

In the second phase, to make a decision based on the hybrid representation, the super-vector was passed through the Principal Component Analysis (PCA) algorithm [21], and the extracted features were given to a DT classifier to facilitate detection. Using the PCA increases a DT classifier's speed in training and testing (see equations 5 and 6). Moreover, DT is a simple, powerful model that can be trained faster than more complex models like DNNs, specifically for small feature sets. In addition, our previous experiments [22] and certain other studies [11] have shown that DT works well on ICS and CPS data. Gini function was used to train the DT (see equation 7).

$$X_{new}^{T} X_{new} = PDP^{-1} \tag{5}$$

In the above equation, P is the eigenvector matrix, and D is a diagonal eigenvalue matrix that the eigenvalues are assigned to the main diagonal, and other values are considered zero.

The eigenvectors were sorted based on the eigenvalues and the first k (number of extracted features) vectors was called P^*. Equation 6 shows the process of extracting k features from dataset X_{new}.

$$X^* = X_{new} P^* \tag{6}$$

In the above equation, X^* is the result of dimensionality reduction using PCA.

$$gini = 1 - \sum_{i=1}^{c} p_i^2 \tag{7}$$

In the above equation, c is the number of classes, and p_i is the probability of the class i in the current branch of the tree.

To detect previously unseen attacks, a One-Class SVM (OCSVM) was used to make a boundary around normal samples and to report the others as previously unseen attacks.

Algorithm 1 shows the algorithm of the proposed attack detection component.

Algorithm 1: The proposed two-phase attack detection component

Data: Dataset including *Normal* and *Attack* samples (X) and their labels($y = \{0, 1\}$)

Training Phase:

$X = z(X)$: $z = \dfrac{x - min(x)}{max(x) - min(x)}$;

$X_{attack} = X[y == 1]$;

$X_{normal} = X[y == 0]$;

♯ Training Representation Learning Models:

for *number of epochs* **do**

  for *number of batches in Normal set* **do**

    Train the Normal autoencoder (AE_{normal}):

    $\min \mathcal{L}(X_{normal}, \hat{X}_{normal})$;

  end

  for *number of batches in Attack set* **do**

    Train the Attack autoencoder (AE_{attack}):

    $\min \mathcal{L}(X_{attack}, \hat{X}_{attack})$;

  end

end

♯ Fusion Layer:

$newRep_{normal} = AE_{normal}.predict(X)$;

$newRep_{attak} = AE_{attack}.predict(X)$;

$X_{superVector} = concat(newRep_{normal}, newRep_{attack})$;

♯ Detection Model:

Feature selection using PCA:

$Selected\_Features(X_{superVector}) = PCA(X_{superVector})$;

Train a DT using the new features:

$DT = Train\_DT(Selected\_Features)$

Testing Phase:

$x_{test} = z(x_{text})$;

$newRep_{normal} = AE_{normal}(x_{test})$;

$newRep_{attack} = AE_{attack}(x_{test})$;

$superVector = concat(newRep_{normal}, newRep_{attack})$;

$\hat{x}_{test} = Selected\_Features(superVector)$;

$\hat{y} = DT(\hat{x}_{test})$;

Output: Normal/Attack Label ($\hat{y}$)

B. Proposed Self-Tuning Attack Attribution Method

The proposed self-tuning attack attribution method consists of two phases. In the first phase, a one-vs-all classifier is trained for each attribute. To train these classifiers, a dataset's attack samples are split into several subsets based on their attributes, and one DNN model is trained for every set. The Rectified Linear Unit (ReLU) function is used as an activation function for the hidden layers, and the Sigmoid function is used as the output-layer activation function. Next, the outputs of all of the first phase DNNs are passed to the second phase to attribute the instances based on one-vs-all DNNs.

In the second phase, the one-vs-all classifiers and a DNN ensemble model are combined to compose a more complex DNN. This DNN is constructed from two components: a partially-connected element consisting of several one-vs-all classifiers and a fully-connected element fusing the first part's results and attributes of the samples into different classes . The ReLU activation function is used for the hidden layers of the ensemble DNN, and the softmax function is used as its output activation function (equation 8). The Categorical Cross-Entropy (CE) is performed as the loss function of the final DNN (equation 9). In addition, the outputs of this DNN are the two most probable attributions for the given sample. This model is called the primary attack attribution method. A DT classifier is trained for each pair of attack attributes used for the final attack attribution from the two candidates, and this is referred to as the secondary attack attribution method.

$$\sigma(s)_i = \frac{e^{s_i}}{\sum_{j=1}^{K} e^{s_j}} \tag{8}$$

where K is the number of classes, and $z = (z_1, ..., z_k) \in \mathbb{R}^K$.

$$CE = -\sum_{i=1}^{K} y_i \log(\sigma(s)_i) \tag{9}$$

where y_i is the label of the $i-th$ class, and $\log(\sigma(s)_i)$ is the output of the softmax function.

This method is self-tuning since it can tune itself by changing the attack patterns without needing pre-processing. This results from using the gradient descent technique to simultaneously update the weights of all one-vs-all classifiers and the ensemble model. This feature is useful when a new attack attribute is discovered, and then it is added to the attack attribution method . This work is done by passing the new dataset, including the new attack attribute, through the proposed attack attribution method. Algorithm 2 shows the algorithm od the proposed attack attribution component.

Algorithm 2: The proposed two-phase attack attribution component

Data: Dataset including *Attack* samples from various families (X) and the labels ($y \in [1, c]$)

Training Phase:

$X = z(X)$: $z = \dfrac{x - min(x)}{max(x) - min(x)}$;

foreach *attack type* i **do**
  foreach *sampe* $x \in X$ **do**
    if $y[x] = i$ **then**
      $y_i = 1$
    end
    else
      $y_i = 0$
    end
  end
end

♯ Training the binary DTs:

foreach *two class of attacks* **do**
  Train a DT
end

♯ Training one-vs-all classifiers:

foreach *attack type* i **do**
  for *number of epochs* **do**
    for *number of batches in the Attack type* i **do**
      Train the one-vs-all classifier ($classifier_i$): $\min \mathcal{L}(y_i, \hat{y}_i)$;
    end
  end
end

♯ Ensemble model:

$DNN = new\ neural\ network$;

foreach *classifier* i **do**
  $DNN.add(classifier_i)$;
end

$DNN.add(fully - connected\ neural\ network)$;

for *number of epochs* **do**
  for *number of batches in training data* **do**
    train the whole network: $\min \mathcal{L}(y, \hat{y})$;
  end
end

Testing Phase:

$x_{test} = z(x_{test})$;

$DNN.predict2bests(x_{test})$;

Pass x_{test} to the DT;

Output: Attack type ($\hat{y}$)

IV. Experimental Setup

A. Dataset

As previously discussed, we evaluated the proposed framework using two real-world ICS datasets. The first dataset was collected at the Mississippi State University [23] from a gas pipeline system consisting of sensors and actuators, a communication network, and supervisory control. This dataset consists of normal samples and seven attack types, including Naïve Malicious Response Injection (NMRI), Complex Malicious Response Injection (CMRI), Malicious State Command Injection (MSCI), Malicious Parameter Command Injection (MPCI), Malicious Function Code Injection (MFCI), Denial of Service (DoS), and Reconnaissance (Recon) attacks. It reportedly contained 274,628 observations, in which 214,580 (78.14%) were normal samples, and the remaining 60,048 (21.86%) samples were attack samples. This dataset also consisted of 17 features of network and field states.

The second dataset was the Secure Water Treatment (SWaT)

dataset [24], collected at Singapore University of Technology from a water treatment system, consisting of 449,920 samples. In this dataset, 87.9% and 12.1% were normal and attack samples, respectively. Each dataset sample was formed by 51 features that were the physical measurements of the systems. In addition, this dataset consisted of 31 different attack scenarios that could be used for attack attribution.

B. Pre-Processing

As shown in Figure 1, the proposed framework consists of several DNNs that accept the raw features as input and map them to new representations for attack detection and attack attribution. Similar to some other approaches [25], [26], [27], the data was normalized using the min-max technique before passing it through the methods to make them unbiased against the features. This was the only pre-processing for the proposed framework. Moreover, 10-fold cross-validation was performed to obtain the results.

C. Evaluation Metrics

To ensure fairness in comparison, this study evaluated the performance of the proposed attack attribution method using the DT classifier on the original representation and approaches that used the same dataset(s) in their original articles. However, for the proposed self-tuning attack attribution method, we were not able to find similar approaches. A comparison with the Fuzzy C-Mean (FCM) clustering [25] verified that FCM could detect only four out of eight classes in the gas pipeline dataset (while our model attributes all eight classes). This suggested that the attacks were very similar and hard to classify .

Similar to other approaches, this study used standard metrics to evaluate the performance of machine learning algorithms. Specifically, it used True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) to represent the number of samples correctly classified as attacks, correctly classified as normal, wrongly classified as attacks, and wrongly classified as normal, respectively. Using these metrics, it is possible to define Accuracy (ACC), Precision (Pre), Recall (Rec), F-measure, Receiver Operating Characteristics (ROC) curve, and Area Under Curve (AUC) to quantify the performance of ML algorithms in performing malware detection.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{10}$$

$$Pre = \frac{TP}{TP + FP} \tag{11}$$

$$Rec = \frac{TP}{TP + FN} \tag{12}$$

$$f - measure = \frac{2 \times Pre \times Rec}{Pre + Rec} \tag{13}$$

- Accuracy indicates the number of samples that are correctly classified over the entire dataset. Since ICS datasets are imbalanced, this metric is not a good one for evaluation (see Equation 10).
- Precision indicates the number of samples that are detected correctly as attack over total samples detected as an attack (see Equation 11).
- Recall indicates the number of samples that are detected as attack correctly over the total samples of the attack in the dataset (see Equation 12).
- F-measure is the harmonic value of precision and recall (see Equation 13).

In the detection task, the desired class is the attack one. The attack class is considered as the positive class for precision, recall, and f-measure metrics.

D. Feature Extraction

PCA was chosen for dimensionality reduction and also to extract the best features from super-vectors. It also improve the performance of the DT classifier by extracting independent features in an unsupervised manner.

To extract the best features using the PCA, 10-fold cross-validation was performed on each dataset's possible number of features. The dataset's principal components were extracted in each run, and the model was trained and tested using the principal components. To make the PCA unbiased to the test data, training was performed on the training data . The number of principal components with the best f-measure over ten runs was then selected as the number of PCA components.

V. Discussions

The proposed attack detection and attack attribution methods form a framework that can keep ICS/IIoT systems secure. This framework is proposed to address the challenge of ICS imbalanced data without ignoring the minority class or balancing the dataset. The proposed framework should be deployed on the physical layer to passively monitor the sensor data and give an alert when an attack happens. In such a case, the data is sent to the attribution model to detect the attack attribute. Finally, security experts and incident response teams can handle attacks and prevent potential damages using the proposed framework's efficient, accurate information.

A. The Proposed Attack Detection Method

The proposed attack detection method consists of a deep representation learning model with two unsupervised stacked autoencoders, feature extraction using the PCA, and a DT classification.

Due to the consideration of both attack and normal data in the training step, the proposed attack detection method can detect previously seen attacks with better f-measures than the other methods, as can be seen in Table I. To enhance the method's ability to face the previously unseen attacks, an anomaly detection module was added to the system trained on the normal data to capture the normal data structure and detect anomalies. The OCSVM model was used in this module.

The proposed attack detection component is scalable to larger ICS with more features and larger data sets. The only part of the system that depends on the ICS architecture is the representation-learning step, which needs more training time by increasing the size of the system and/or the data's size. However, it will not affect the performance of the proposed framework in real implementation.

1) General Performance: As observed in Table I, the proposed method outperformed the base DT model on the original representation in all metrics. Moreover, it outperformed other techniques in the f-measure metric (i.e. the harmony between precision and recall and an important metric to evaluate imbalanced datasets). In addition, the proposed attack detection method outperformed all other techniques on the SWaT dataset. In other words, the proposed attack detection method achieved good precision without affecting the recall metric on the data. As discussed earlier, accuracy is not a useful metric by which to evaluate models' performances using imbalanced datasets; in this case, by labeling all of the samples with the majority class label, the model achieved high accuracy (78.14% in gas and 87.9% in the SWaT dataset).

Moreover, as shown in Table II, the proposed attack detection method has a higher recall (true-positive rate) than other techniques for each attack attribute. In other words, the proposed method detects more attacks than the others when trained on only one attack type.

Table I reinforces the importance of the representation learning models to ICS datasets. The proposed deep representation learning step enables the method to develop new features separately for normal and attack data in an unsupervised manner based on their patterns. In turn, these new features allow the DT to perform a more effective classification than was facilitated using the original features.

2) Imbalanced Testing: The reported higher f-measure in Table I shows that the proposed attack detection method achieved better performance on the imbalanced datasets. To evaluate the robustness of the proposed ensemble two-phase attack detection method for imbalanced ICS data, this study generated different sets of data with different imbalance ratios by varying the number of attack samples in the original dataset. These sets were obtained from the original datasets and generated randomly. Next, the new datasets were fed into the proposed attack detection method and compared with several base classifiers, including DT, Logistic Regression (LR), Gradient Boosting (GB), AdaBoost M1 (AB), and Random Forest (RF). The new imbalanced sets were used for training to ensure a fair comparison, and the evaluation was performed using a predefined test set. In addition to achieving better performance for the proposed attack detection method in all metrics, the proposed model resulted in a robust, consistent performance in all metrics for both datasets (see Figure 2). Robustness refers to the low variance of the changes in the performance of the model. It indicates that the proposed attack detection method achieves high accuracy, low false positives, and high f-measures simultaneously, thereby outperforming the competing approaches. More specifically, the high f-measure of the proposed method is significant in performance evaluation for imbalanced datasets.

Beyond this, the findings suggested that the proposed method mitigates the challenge of the imbalanced problem in DNNs by separating the attack and normal samples and running separate, unsupervised stacked autoencoders on each of them. Using this technique, major class samples' effects on the gradient descent algorithm are avoided/omitted, enabling the autoencoders to extract more useful features from the minority set. Furthermore, the fusion layer consists of useful representations from both majority (normal) and minority (attack) data.

3) Previously Unseen Attack Detection: To detect previously unseen attacks, the OCSVM model was added to the proposed framework. OCSVM, a type of SVM, attempts to maximize the decision boundary's margin to yield better generalization. Based on the evaluations, we observed that this method correctly detected 86.14% of previously unseen attacks in the gas pipeline dataset. Moreover, 94.53% of the previously unseen attacks were detected correctly in the SWaT dataset.

4) Execution Time Comparison: Table III compares the proposed attack detection component's execution time with other proposed methods in the literature. As illustrated in Table III, it takes 1200 seconds to train the whole model on the SWaT dataset, while applying the trained model over testing samples takes 2.98 seconds, which means around 0.03 milliseconds for each sample. Moreover, training the proposed method on the Gas Pipeline dataset takes 1115 seconds, while the test takes around 1.1 seconds, which means around 0.02 milliseconds for each sample. As can be seen from Table III, the proposed model is faster than most DNN-based techniques due to its simpler architecture combined with the PCA method, which makes the DT faster. Besides, the proposed attack detection component's execution time illustrates that it can detect attack samples in almost real-time (0.02 milliseconds for the Gas Pipeline dataset and 0.03 milliseconds for the SWaT dataset).

B. The Proposed Attack Attribution Method

In the proposed attack attribution method, a one-vs-all DNN classifier was responsible for extracting each attribute's pattern and assigned belonging confidence to each observation. These confidences from all DNNs were passed to another DNN, which was responsible for attack attribution. Due to the close patterns of the attacks [25], this DNN was not performed well. However, it can detect attributes better than FCM. To improve the attack attribution method performance, this study defined a two-step method. In the first step, the aforementioned DNN determined the two best attribute candidates for the observed sample. In the second step, the observed sample was sent to a DT pre-trained on the samples of two candidate attributes to detect the best attribute.

Using one-vs-all classifiers for each attack attribute guarantees that each classifier passes the best result to the ensemble DNN model that yields better performance, as this paper will show here. These classifiers were connected to a DNN fusion model to pass their extracted features and fuse them into the fusion model to attribute the samples. Each one-vs-all classifier was a supervised DNN that encoded the input features within an 8-dimensional space and then into a 128-dimensional space using the ReLU activation function. Based on the final representation, the output layer classified it. The fusion model is another DNN; its inputs were the outputs of the one-vs-all classifiers. This fusion model decoded the input features in the 128-dimensional space, followed by a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3067667, IEEE Internet of Things Journal

TABLE I
COMPARISON OF THE PROPOSED ATTACK DETECTION METHOD WITH OTHER TECHNIQUES ON THE GAS PIPELINE AND SWAT DATASETS

SWaT Dataset				Pipeline Dataset				
Method	**Pre**	**Rec**	**f-measure**	**Method**	**ACC**	**Pre**	**Rec**	**f-measure**
Proposed method	**0.9999**	**0.9999**	**0.9998**	**Proposed method**	96.20	0.9617	**0.9620**	**0.9618**
DT	0.8411	0.8284	0.8346	**DT**	91.11	0.9092	0.9111	0.9099
LAD-ADS [13]	0.936	0.891	0.914	**SVM [28]**	92.50	0.782	0.936	0.852
DNN [26]	0.9829	0.6785	0.8028	**K-means [25]**	56.80	0.8319	0.5728	0.6751
1D CNN [29]	0.868	0.854	0.861	**NB [25]**	90.36	0.8195	0.7692	0.8595
MADGAN [30]	0.9897	0.6374	0.77	**AllKNN [12]**	**97**	**0.98**	0.92	0.95
Tabor [31]	0.8617	0.7880	0.8232	**LSTM [32]**	92	0.94	0.78	0.85
LSTM [33]	0.951	0.627	0.756					
ST-ED [33]	0.949	0.705	0.809					

TABLE II
COMPARISON BETWEEN THE RECALL OF THE PROPOSED ATTACK DETECTION METHOD AND OTHER TECHNIQUES ON THE GAS PIPELINE DATASET ATTACK ATTRIBUTES

Model	**NMRI**	**CMRI**	**MSCI**	**MPCI**	**MFCI**	**DoS**	**Recon.**
Proposed attack detection method	**0.97**	**0.95**	**0.97**	**0.95**	**1**	**1**	**1**
AllKNN [12]	0.93	0.76	0.68	0.85	**1**	0.98	**1**
LSTM [32]	0.88	0.67	0.62	0.80	**1**	0.94	**1**
K-means [25]	0.19	0.20	0.73	0.66	0.52	0.56	0.75
NB [25]	0.81	0.84	0.73	0.67	0.52	0.79	0.50

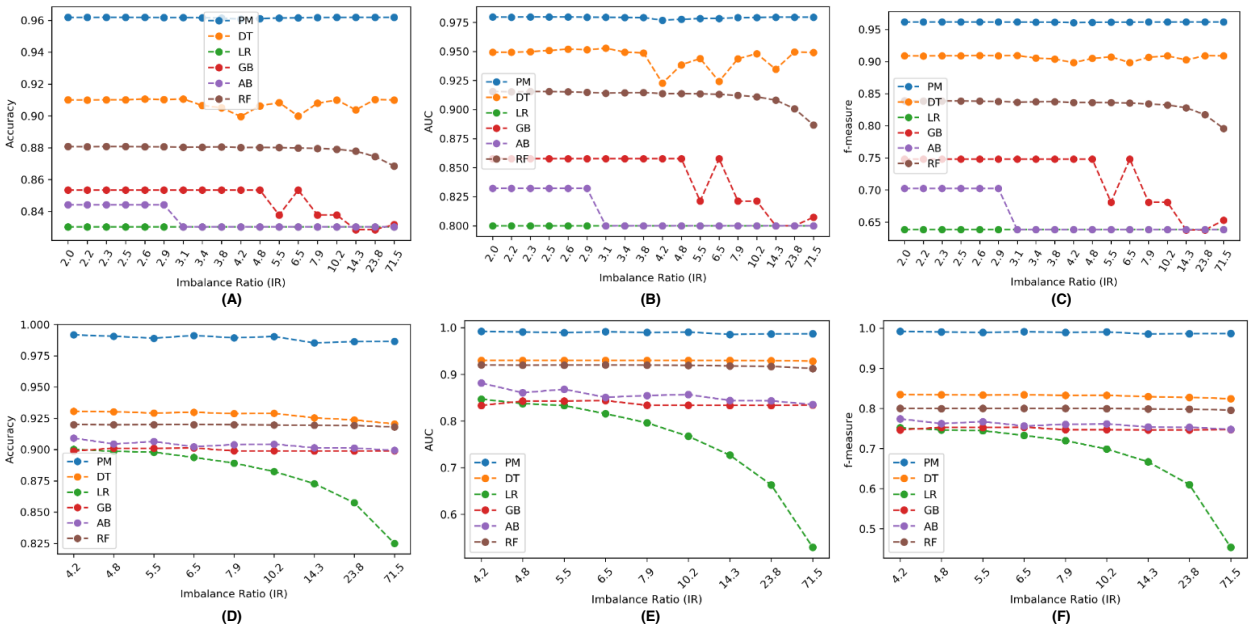

Fig. 2. Comparison of accuracy , AUC, and f-measure of the proposed attack detection method and other basic classifiers on original representation for different attack IR (A), (B), and (C) on the gas pipeline dataset and (D), (E), and (F) on the SWaT dataset. In the figures, PM is the proposed attack detection method, DT is the Decision Tree, LR is the Logistic Regression, GB is the Gradient Boosting, AB is the AdaBoost M1, and RF is the Random Forest.

64-dimensional space using the ReLU activation function. The output layer used the softmax activation function to attribute the observation to the given attributes (31 for the SWaT dataset and seven for the gas pipeline dataset).

As discussed in [25], running the FCM algorithm on the gas pipeline dataset with the eight clusters resulted in four clusters. This implies that the attacks are very similar and share many common features that the FCM algorithm considers them one group. To overcome this problem, this study detected the two most probable attack attributes for each sample using the ensemble model. These samples were fed into the DT classifier, which was trained on the two most probable attributes to obtain the final attack attribute. This was labelled the secondary attack attribution method. As observed in Table IV, all of the metrics improved significantly by using the final DT model (secondary attack attribution) compared with the primary attack attribution method (using the output of DNN model) on both datasets. Thus, the attack attribution method can attribute all attacks with reasonable confidence (as a best or second-best result). Figure 3 compares the confusion matrices for the performance of the proposed primary and secondary attack attribution methods for the gas pipeline dataset. The confusion matrix for the SWaT dataset is not reported due to page limitations since it includes 36 different

TABLE III
COMPARISON OF THE TRAIN AND TEST EXECUTION TIME OF THE PROPOSED ATTACK DETECTION METHOD WITH OTHER TECHNIQUES ON THE GAS PIPELINE AND SWAT DATASETS. IN THIS TABLE, S STANDS FOR SECONDS AND W STANDS FOR WEEKS.

SWaT Dataset			Pipeline Dataset		
Method	**Train**	**Test**	**Method**	**Train**	**Test**
Proposed method	1200s	2.98s	**Proposed method**	1115s	1.10s
LAD-ADS [13]	8820s	2s	**SVM [28]**	11712	-
DNN [26]	2w	28800s	**AllKNN [12]**	-	5.99s
Tabor [31]	214s	33s	**LSTM [32]**	2100s	1.65s
LSTM [33]	57s (per epoch)	13s			
ST-ED [33]	692s	217.50s			

attack attributes. Despite the strong evaluation results of the secondary attack attribution method, it cannot discriminate between DoS and MPCI samples due to the similar impacts of these attacks on its features.

The proposed attack attribution component is scalable to larger ICS with more features and larger data sets. However, its execution time depends on the number of attack classes and almost independent of the system's size (features).

1) Execution Time: Training of the proposed attack attribution component on the Gas Pipeline dataset took 1155 seconds, while the attribution over test data took 0.65 seconds, which means around 0.05 milliseconds for each sample. Moreover, training of the proposed attack attribution component on the SWaT dataset took 3452 seconds, and it classified the test data in 2.87 seconds, which means around 0.27 milliseconds for each sample. The proposed model's training and testing execution time depend on the number of attribute classes (seven classes for the Gas Pipeline dataset vs. 31 classes for the SWaT dataset).

C. Computational Complexity

In this section, the computational complexity of the proposed attack detection and attribution methods will be analyzed.

The computational complexities of training and testing the used algorithms are shown in Table V [34], [35]. In this table, n is the number of training samples, and the computational complexities were calculated for the worst-case scenario, in which the number of input features, number of neurons in each layer, number of selected support vectors, and depth of the DT is considered to be n.

1) The Proposed Attack Detection Method: As mentioned before, the proposed attack detection method consists of a novel form of deep representation learning, PCA feature extraction, and a DT classification. Each deep representation learning model has three encoding and three decoding layers.

Based on Table V, the computational complexity of training the proposed deep representation learning in the worst-case scenario is $O(n^4)$, where n is the number of training samples.

The other parts of this method are the PCA and DT algorithms. As mentioned in Table V, in the worst-case scenario, the PCA and DT algorithms' computational complexity is equal to $O(n^3)$. Equation 14 shows the computational complexity of training of the proposed attack detection method.

$$O(n^4) + O(n^3) + O(n^3) = O(n^4) \tag{14}$$

which is similar to the other DNN-based detection methods in the literature.

Moreover, the testing computational complexity of the proposed attack detection method is shown in Equation 15.

$$O(n^2) + O(n) + O(1) = O(n^2) \tag{15}$$

which is similar to all other DNN-based methods (except the recurrent neural network-based methods) in the literature.

Adding the previously unseen module did not change the computational complexity of training and testing the proposed attack detection technique since the OCSVM's training computational complexity is $O(n^3)$. In addition, its testing computational complexity is $O(n^2)$, which cannot affect the proposed attack detection method's computational complexity.

2) The Proposed Attack Attribution Method: The proposed attack attribution method includes several one-vs-all DNNs connected using another DNN to make a deeper DNN model. The best two attribution candidates were selected using this DNN model, and a pre-trained DT on the candidate attributes was used to detect the final attributes. As a DT should be trained for every two attributes, $\frac{c\times(c-1)}{2}$ DTs should be trained; where c is the number of attributes, each has a computational complexity of $O(n^3)$. Thus, the computational complexity of training all of the DTs is $O(c^2 \times n^3)$, where c is the number of attributes, and n is the number of training samples.

In addition to the DTs, the proposed attack attribution method used DNNs with the training computational complexity of $O(n^4)$. Combining the DTs' and the DNN model's training, the computational complexity of training the proposed attack attribution model is shown in Equation 16.

$$O(c^2 \times n^3) + O(n^4) = O(n^4) \tag{16}$$

where c is the number of attributes, and n is the number of training samples. Since the number of training samples is significantly larger than the number of attributes, the number of attributes is ignored in the computational complexity analysis. As seen in Equation 16, the computational complexity of training the proposed attack attribution method is similar to that of the other DNN methods.

The proposed attack attribution's testing computational complexity is $O(n^2)$, similar to the computational complexities of the other DNN-based techniques in the literature.

TABLE IV
RESULTS OF THE PROPOSED SELF-TUNING TWO-PHASE ATTACK ATTRIBUTION METHOD ON BOTH GAS PIPELINE AND SWAT DATASETS

Model	Accuracy		Precision		Recall		f-measure	
	Gas	SWaT	Gas	SWaT	Gas	SWaT	Gas	SWaT
Proposed primary attack attribution method	78.08	99.53	0.7906	0.9959	0.7808	0.9953	0.7857	0.9956
Proposed secondary attack attribution method	98.14	99.71	0.9822	0.9974	0.9814	0.9971	0.9818	0.9972

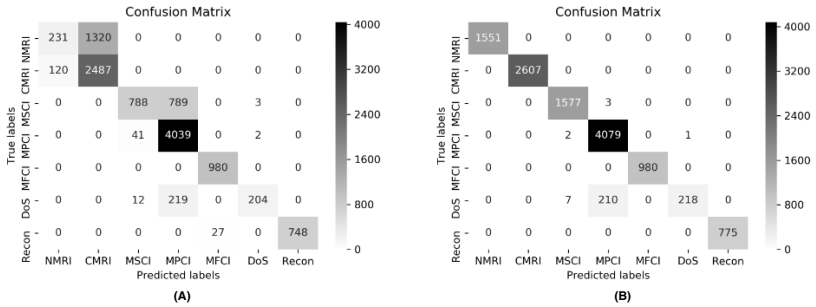

Fig. 3. Confusion matrices of the proposed attack attribution method on the gas pipeline dataset for (A) the proposed primary attack attribution method and (B) the proposed secondary attack attribution method

TABLE V
COMPUTATIONAL COMPLEXITY OF THE USED ALGORITHMS

Algorithm	Training	Testing
DT	$O(n^3)$	$O(n)$
PCA	$O(n^3)$	$O(1)$
OCSVM	$O(n^3)$	$O(n^2)$
DNN	$O(n^4)$	$O(n^2)$

D. Implementation in Real-World Environment

The proposed framework can be implemented in the same network layer as the Human Machine Interface (HMI) to observe the sensor data from field devices and detect and attribute attacks. It also can be connected to the monitoring system in control center to inform the security experts about the presence of the attack and help them choose preventive actions in a timely manner. Moreover, the provided information helps the incident response team understand the attack and its impacts, based on the attribution information, to revive the damaged assets

As shown in Figure 4, at first the input sensor data are fed into the detection component. The detection components classify it as normal or attack based on its previous experience (training data). If the entered sample is detected as normal, it will pass to an OCSVM module for further investigation by comparing it to normal samples' profiles. However, if the detection component detects the sample as an attack, it will go to the attribution component to extract its attribution. All the outputs are then passed to a monitoring system.

VI. CONCLUSION

This paper proposed a novel two-stage ensemble deep learning-based attack detection and attack attribution framework for imbalanced ICS data. The attack detection stage uses deep representation learning to map the samples to the new higher dimensional space and applies a DT to detect the attack samples. This stage is robust to imbalanced datasets and capable of detecting previously unseen attacks. The attack attribution stage is an ensemble of several one-vs-all classifiers, each trained on a specific attack attribute. The entire model forms a complex DNN with a partially connected and fully connected component that can accurately attribute cyber-attacks, as demonstrated. Despite the complex architecture of the proposed framework, the computational complexity of the training and testing phases are respectively $O(n^4)$ and $O(n^2)$, (n is the number of training samples), which are similar to those of other DNN-based techniques in the literature. Moreover, the proposed framework can detect and attribute the samples timely with a better recall and f-measure than previous works.

Future extension includes the design of a cyber-threat hunting component to facilitate the identification of anomalies invisible to the detection component for example by building a normal profile over the entire system and the assets.

REFERENCES

[1] F. Zhang, H. A. D. E. Kodituwakku, J. W. Hines, and J. Coble, "Multilayer Data-Driven Cyber-Attack Detection System for Industrial Control Systems Based on Network, System, and Process Data," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4362–4369, 2019.

[2] R. Ma, P. Cheng, Z. Zhang, W. Liu, Q. Wang, and Q. Wei, "Stealthy Attack Against Redundant Controller Architecture of Industrial Cyber-Physical System," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9783–9793, 2019.

[3] E. Nakashima, "Foreign hackers targeted U.S. water plant in apparent malicious cyber attack, expert says." [Online]. Available: https://www.washingtonpost.com/blogs/checkpoint-washington/post/foreign-hackers-broke-into-illinois-water-plant-control-system-industry-expert-says/2011/11/18/gIQAgmTZYN_blog.html

[4] G. Falco, C. Caldera, and H. Shrobe, "IIoT Cybersecurity Risk Modeling for SCADA Systems," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4486–4495, 2018.

[5] J. Yang, C. Zhou, S. Yang, H. Xu, and B. Hu, "Anomaly Detection Based on Zone Partition for Security Protection of Industrial Cyber-Physical Systems," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 5, pp. 4257–4267, 2018.

[6] S. Ponomarev and T. Atkison, "Industrial control system network intrusion detection by telemetry analysis," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 2, pp. 252–260, 2016.

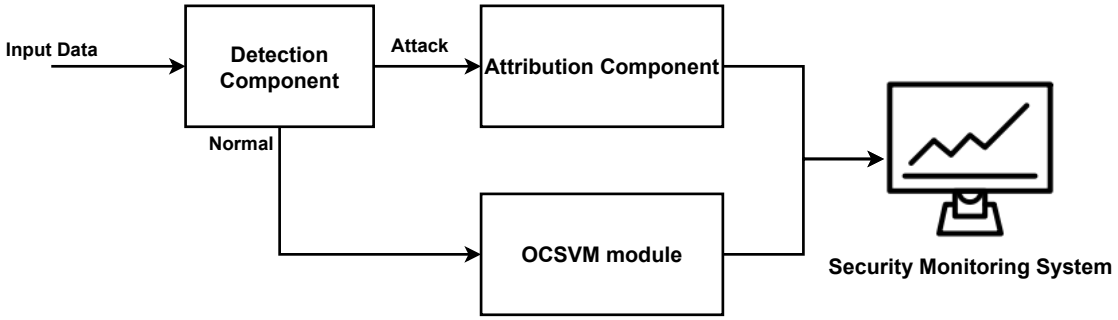

Fig. 4. Interaction of the proposed framework in real environment.

[7] J. F. Clemente, "No cyber security for critical energy infrastructure," Ph.D. dissertation, Naval Postgraduate School, 2018.

[8] C. Bellinger, S. Sharma, and N. Japkowicz, "One-class versus binary classification: Which and when?" in *2012 11th International Conference on Machine Learning and Applications*, vol. 2, 2012, pp. 102–106.

[9] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[10] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[11] M. Zolanvari, M. A. Teixeira, L. Gupta, K. M. Khan, and R. Jain, "Machine Learning-Based Network Vulnerability Analysis of Industrial Internet of Things," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6822–6834, 2019.

[12] I. A. Khan, D. Pi, Z. U. Khan, Y. Hussain, and A. Nawaz, "HML-IDS: A hybrid-multilevel anomaly prediction approach for intrusion detection in SCADA systems," *IEEE Access*, vol. 7, pp. 89 507–89 521, 2019.

[13] T. K. Das, S. Adepu, and J. Zhou, "Anomaly detection in industrial control systems using logical analysis of data," *Computers & Security*, vol. 96, p. 101935, 2020.

[14] J. J. Q. Yu, Y. Hou, and V. O. K. Li, "Online False Data Injection Attack Detection With Wavelet Transform and Deep Neural Networks," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 3271–3280, 2018.

[15] M. M. N. Aboelwafa, K. G. Seddik, M. H. Eldefrawy, Y. Gadallah, and M. Gidlund, "A machine-learning-based technique for false data injection attacks detection in industrial iot," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8462–8471, 2020.

[16] W. Yan, L. K. Mestha, and M. Abbaszadeh, "Attack detection for securing cyber physical systems," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8471–8481, 2019.

[17] A. Cook, A. Nicholson, H. Janicke, L. Maglaras, and R. Smith, "Attribution of Cyber Attacks on Industrial Control Systems," *EAI Endorsed Transactions on Industrial Networks and Intelligent Systems*, vol. 3, no. 7, p. 151158, 2016.

[18] L. Maglaras, M. Ferrag, A. Derhab, M. Mukherjee, H. Janicke, and S. Rallis, "Threats, Countermeasures and Attribution of Cyber Attacks on Critical Infrastructures," *ICST Transactions on Security and Safety*, vol. 5, no. 16, p. 155856, 2018.

[19] M. Alaeiyan, A. Dehghantanha, T. Dargahi, M. Conti, and S. Parsa, "A Multilabel Fuzzy Relevance Clustering System for Malware Attack Attribution in the Edge Layer of Cyber-Physical Networks," *ACM Transactions on Cyber-Physical Systems*, vol. 4, no. 3, pp. 1–22, 2020.

[20] U. Noor, Z. Anwar, T. Amjad, and K.-K. R. Choo, "A machine learning-based FinTech cyber threat attribution framework using high-level indicators of compromise," *Future Generation Computer Systems*, vol. 96, pp. 227–242, 2019.

[21] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1, pp. 37 – 52, 1987, proceedings of the Multivariate Statistical Workshop for Geologists and Geochemists.

[22] A. N. Jahromi, J. Sakhnini, H. Karimpour, and A. Dehghantanha, "A deep unsupervised representation learning approach for effective cyber-physical attack detection and identification on highly imbalanced data," in *Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering*, ser. CASCON '19. USA: IBM Corp., 2019, p. 14–23.

[23] T. Morris, Z. Thornton, and I. Tunipseed, "Industrial control system simulation and data logging for intrusion detection system research," in *7th Annual Southeastern Cyber Security Summit*, 2015.

[24] J. Goh, S. Adepu, K. N. Junejo, and A. Mathur, "A dataset to support research in the design of secure water treatment systems," in *Critical Information Infrastructures Security*, G. Havarneanu, R. Setola, H. Nassopoulos, and S. Wolthusen, Eds. Cham: Springer International Publishing, 2017, pp. 88–99.

[25] S. N. Shirazi, A. Gouglidis, K. N. Syeda, S. Simpson, A. Mauthe, I. M. Stephanakis, and D. Hutchison, "Evaluation of anomaly detection techniques for scada communication resilience," in *2016 Resilience Week (RWS)*, 2016, pp. 140–145.

[26] J. Inoue, Y. Yamagata, Y. Chen, C. M. Poskitt, and J. Sun, "Anomaly detection for a water treatment system using unsupervised machine learning," *IEEE International Conference on Data Mining Workshops, ICDMW*, vol. 2017-November, pp. 1058–1065, 2017.

[27] M. Kravchik and A. Shabtai, "Detecting cyber attacks in industrial control systems using convolutional neural networks," *Proceedings of the ACM Conference on Computer and Communications Security*, no. 1, pp. 72–83, 2018.

[28] S. D. Anton, A. Hafner, S. Sinha, and H. Schotten, "Anomaly-based intrusion detection in industrial aata with SVM and random forests," in *the 27th International Conference on Software, Telecommunicationsand Computer Networks (SoftCOM)*. IEEE, 2019.

[29] M. Kravchik and A. Shabtai, "Efficient cyber attack detection in industrial control systems using lightweight neural networks and pca," *IEEE transactions on dependable and secure computing*, pp. 1–1, 2021.

[30] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S. K. Ng, "MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11730 LNCS, pp. 703–716, 2019.

[31] Q. Lin, S. Verwer, S. Adepu, and A. Mathur, "TABOR: A graphical model-based approach for anomaly detection in industrial control systems," *ASIACCS 2018 - Proceedings of the 2018 ACM Asia Conference on Computer and Communications Security*, pp. 525–536, 2018.

[32] C. Feng, T. Li, and 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)*, 2017, pp. 261–272.

[33] M. Macas and W. Chunming, "Enhanced cyber-physical security through deep learning techniques," *CEUR Workshop Proceedings*, vol. 2457, no. 38, 2019.

[34] C.-t. Chu, S. Kim, Y.-a. Lin, Y. Yu, G. Bradski, K. Olukotun, and A. Ng, "Map-reduce for machine learning on multicore," in *Advances in Neural Information Processing Systems*, B. Schölkopf, J. Platt, and T. Hoffman, Eds., vol. 19. MIT Press, 2007, pp. 281–288.

[35] J. Su and H. Zhang, "A fast decision tree learning algorithm," in *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, ser. AAAI'06. AAAI Press, 2006, p. 500–505.