Journal of Information Security and Applications 41 (2018) 1–11

Contents lists available at ScienceDirect

Journal of Information Security and Applications

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

# Identification of malicious activities in industrial internet of things based on deep learning models

Muna AL-Hawawreh, Nour Moustafa*, Elena Sitnikova

*School of Engineering and Information Technology, University of New South Wales, Australian Defence Force Academy (ADFA), Canberra, Australia*

ARTICLE INFO

*Article history:*
Available online 22 May 2018

*Keywords:*
Industrial internet of things (IIoT)
Internet industrial control systems (IICSs)
Deep learning
Auto-encoder

ABSTRACT

Internet Industrial Control Systems (IICSs) that connect technological appliances and services with physical systems have become a new direction of research as they face different types of cyber-attacks that threaten their success in providing continuous services to organizations. Such threats cause firms to suffer financial and reputational losses and the stealing of important information. Although Network Intrusion Detection Systems (NIDSs) have been proposed to protect against them, they have the difficult task of collecting information for use in developing an intelligent NIDS which can proficiently detect existing and new attacks. In order to address this challenge, this paper proposes an anomaly detection technique for IICSs based on deep learning models that can learn and validate using information collected from TCP/IP packets. It includes a consecutive training process executed using a deep auto-encoder and deep feedforward neural network architecture which is evaluated using two well-known network datasets, namely, the NSL-KDD and UNSW-NB15. As the experimental results demonstrate that this technique can achieve a higher detection rate and lower false positive rate than eight recently developed techniques, it could be implemented in real IICS environments.



## 1. Introduction

Cyberspace plays a key role in contemporary societies and economies as the internet has changed the ways in which people and organizations communicate and conduct business electronically [1]. Therefore, different devices, applications, and services, which link the virtual and physical worlds, are included in the new term the 'Industrial Internet of Things' (IIoT) [2]. The interoperability of Information Technology (IT) and Operational Technology (OT) exposes industrial environments that depend on closed and proprietary communication protocols to diverse types of anomalous activities [3]. IIoTs are connected to the internet via the TCP/IP protocol in the forms of Machine-to-machine (M2M) and Machine-to-people (M2P) using specific IIoT protocols, for example, Message Queue Telemetry Transport (MQTT) and Advanced Message Queuing Transport (AMQT) [4]. There have been substantial increases in the numbers of loopholes and vulnerabilities in IICSs that can be breached using several sophisticated attack techniques, whereby attackers attempt to exploit these systems in order to achieve their aims of stealing valuable information and/or financial funds, and/or corrupting device resources [5]. It is expected that cyber threats to the IIoT/IICSs will cost up to $90 trillion by 2030 if the cyber-security domain cannot find promising control solutions for preventing them [6].

With the number of IIoT devices and applications rapidly increasing, protecting critical infrastructures (i.e., IICSs) is becoming a more critical issue for business [7]. In IIoT environments, malware, which leverages zero-day vulnerabilities, is one of the most common threats, whereby attackers infect critical devices in order to control and modify their operations using different methodologies, such as Advanced Persistent Threat (APT), Denial of Service (DoS) and Distributed DoS (DDoS). For example, the Stuxnet worm attacked the Iranian nuclear program in 2010, Iranian hackers penetrated the ICS of New York's dam in 2013, black-energy malware was directly responsible for power outages for at least 80.000 customers in Ukraine in 2015 and, more recently, a SFG malware attack targeted European energy companies [10,11]. These malicious activities have proven that the 'security by obscurity' or traditional cyber-security mechanisms, including security policies, authentication, firewalls and signature-based Intrusion Detection Systems (IDSs), are no longer appropriate schemes for achieving efficient protection for critical infrastructure.

To detect IIoT attacks, a Network IDS (NIDS), which is the second line of defense after firewall, antivirus and access control systems, has to be deployed [8]. It is defined as a software and/or hardware mechanism used to monitor and detect suspicious events throughout networked systems [9], with its method-

* Corresponding author.
*E-mail address:* nour.moustafa@unsw.edu.au (N. Moustafa).

https://doi.org/10.1016/j.jisa.2018.05.002
2214-2126/

ology categorized as either signature- or anomaly-based detection. The former identifies existing intrusions by comparing upcoming rules/signatures against a blacklist of their known rules but cannot detect new attacks while the latter can detect known and new attacks but also creates a huge number of errors [8]. An anomaly-based IDS could be a powerful technique if its methodology could successfully detect known and unknown attacks that attempt to breach IIoTs [8,9]. According to the literature, IDSs have been built based on classical machine-learning and data-mining techniques [12,13], rules-based models [14], artificial intelligence approaches [15] and statistical models [40]. However, these methods often produce high False Positive Rates (FPRs) due to overlapping between legitimate and anomalous observations.

In this study, we propose an effective Anomaly Detection System (ADS) for IICSs using deep-learning models to address the drawback of FPRs as much as possible because these models can automatically analyze raw network data to discover abnormal patterns efficiently. A deep-learning technique is very effective, as it can deal with high dimensionality and determine the latent structure from unlabeled data [34]. More importantly, in the training phase, it conducts a consecutive training process using the unsupervised Deep Auto-Encoder (DAE) algorithm to learn normal network behaviors and produce the optimal parameters (i.e., weights and biases). Then, a standard supervised deep neural network model uses the estimated parameters of the ADE models for effectively tuning its parameters and classifying network observations. The proposed technique is evaluated on two benchmark datasets, the NSL-KDD [38] and UNSW-NB15 [39,44,45], due to their widely and recently use for assessing ADSs. The experimental results reveal the superiority of the proposed technique compared with different network intrusion detection mechanisms which clarify its effectiveness for deployment in real-world IICS environments.

The remainder of this paper is organized as follows. Section 2 provides an overview of the most relevant literature concerning ADSs in industrial control systems and the IoT. Section 3 discusses the use of deep learning as an ADS. In Section 4, details of the design of the proposed model and its deployment in an IIoT environment are presented. Descriptions of the datasets and evaluation metrics used are presented in Section 5, and the experimental results discussed in Section 6. Finally, the conclusion is presented in Section 7.

## 2. Background and related work

This section explains the ADS technology and its approaches in IoT and industrial environments. Moreover, we focus on the approaches of shallow and deep networks, which are related to our proposed technique, demonstrating their capability of identifying suspicious activity.

### 2.1. ADS technology

An ADS is a fundamental security control mechanism which acts as a sniffer and decision engine for monitoring network traffic and identifying abnormal activities [18]. We focus on one that establishes a profile from normal data and considers any variation from it an attack because it can detect both known and unknown (zero-day) attacks [8,19]. Some ADSs have been introduced in IIoTs; for example, Shang et al. [20] proposed a Particle Swarm Optimization (PSO) technique-based ADS for improving the efficiency of the One Class Support Vector Machine (OCSVM) model by extracting packets of the Modbus/TCP communication protocol for training and validating the model. Similarly, Maglaras and Jiang [21] developed an IDS/ADS based on this model which was trained on offline data using the network traces collected from a SCADA environment. Silva and Schukat [22] used a K-NN classifier to build an IDS based on the context of the Modbus/TCP protocol. Although the above mechanisms achieved reasonable performances to some extent, they were dedicated to specific protocols with high FPRs.

Stewart et al. [23] proposed an adaptive IDS for fitting the dynamic architectures of SCADA systems using different OCSVM models to choose the most appropriate one for effectively detecting different attacks. However, this system consumed a high amount of computational resources while executing and produced a high false alarm rate for detection. Shang et al. [24] proposed an ADS for discovering attacks that penetrated the Modbus/TCP protocol by extracting different features of communication activities from SCADA systems which were used by a SVM algorithm to classify attacks. However, the detection process was not sufficiently effective for detecting abnormal behaviors.

Meglaras and Jiang [25] combined the OCSVM model and recursive K-means clustering algorithm to avoid the influence of kernel parameters on the OCSVM for effectively detecting network attacks. An IDS for a critical infrastructure based on an Artificial Neural Network (ANN) mechanism, which used error back-propagation and Levenberg-Marquard functions to train a multi-perceptron ANN to detect abnormal network traffic, was presented by Linda et al. [26]. Hodo et al. [27] adopted an ANN to detect DoS/DDoS attacks in IoTs using a simulated network, and Chen et al. [28] proposed an artificial immune-based distributed IDS for IoT systems. More recently, Marsden et al. [56] proposed a Probability Risk Identification based Intrusion Detection System (PRI-IDS) mechanism by inspecting network traffic of the Modbus TCP/IP protocol for detecting replay attacks. However, these schemes produced high false alarm rates and had a difficulty of recognizing some new attacks.

### 2.2. Deep networks for IDS

IDSs have been studied using shallow and deep networks for detecting abnormal observations from the host- and network-based systems [47–50,54]. A shallow network is an ANN that consists of often one/two hidden layer(s), whilst a deep network comprises many hidden layers with different architectures. Deep learning is one of the most popular machine-learning techniques that academic and industrial researchers use due to its capability of learning a computational process in depth that mimic the natural behaviors of a human's brain [29].

Deep learning can be categorized into different types depending on its architectural design which consists of hierarchical layers of non-linear processing levels [17]. According to Hodo et al [54], Deep networks are classified based on its architecture into generative and discriminative; the generative architecture models a joint probability distribution for observed data with their classes. There are four types of generative models, which are Recurrent Neural Networks (RNN), Dee Belief Network (DBN), Auto-Encoder (AE), and Deep Boltzmann Machine (DBM). The discriminative architecture models the posterior distributions of classes conditioned on the observed data comprises RNN and Convolutional Neural Network (CNN) [54]. These models are described as follows.

- **Generative deep architectures**
  - ***RNN*** is considered as a supervised or unsupervised learning model. The core theory behind it is that information is linked in long sequences via a layer-by-layer connection with a feedback loop. There is a directed cycle between its layers that increase its reliability, with the capability of creating an internal memory for storing data of the previous input. RNN has two types: Elman and Jordan, based on the way of layer connections. Elman consists of three layers (i.e., input, hidden, and output) in addition to the context layer. The hidden layer is connected to the context layer after each

feed-forward and learning rules are applied, a copy of the previously hidden units is saved at the context units. Jordan is like Elman networks but the context units are fed from the output units.

- ***DAE*** is used for learning efficient coding in an unsupervised manner. The simplest architecture of DAE involves an input layer, more than one hidden layer and an output layer that has the same number of neurons in the input layer for reconstruction.
- ***DBM*** is an undirected probabilistic generative model. It consists of energy and stochastic units for the overall network for producing binary results. A Restricted Boltzmann Machine (RBM) is applied to reduce hidden layers, which does not allow intra-layer connections between hidden units. Training a stack of DBM on unlabeled data as the input of the next layer and inserting a layer for discrimination can lead to constructing an architecture of DBN.
- ***DBN*** consists of multiple hidden layers, where a connection is between layers not between units within each layer. It is a composition of unsupervised and supervised learning networks. The unsupervised model is learned by a greedy layer-by-layer connection at a time, whereas the supervised network is one or more layers linked for classifying tasks.

- **Discriminative deep architectures**
  - ***RNN*** utilizes discriminative power for a classification task, and this occurs when the output of the model is labeled data in a sequence with the input.
  - ***CNN*** is a space invariant multi-perceptron ANN, which is biologically inspired by the organization of the animal visual cortex. It has many hidden layers, which typically consists of convolutional layers, pooling layers, fully connected layers and normalization layer. The convolutional layers share many weights that have a few parameters and this makes the CNN is easier in the training process compared with other models with the same number of hidden units.

Many recent research studies [46–53] have applied deep learning techniques for IDSs. A study by Alom et al. [46] used DBN which adopted the greedy layer-by-layer learning algorithm to learn each stack of RBM at a time to identify intrusion activities. Similarly, Gao et al [47] suggested the use of the DBN technique to build an IDS. In the training phase, the greedy layer-by-layer algorithm was used for pre-training and fine-tuning the model. In [48], a deep auto-encoder algorithm was utilized to reduce the data dimensions and a pre-stage of classifying network data. The ANN mechanism was adopted as a classifier to evaluate the efficiency of the auto-encoder compared with the Principle Component Analysis (PCA), kernel- PCA and factor analysis algorithms. The results revealed the technique's efficiency for detecting network attacks.

Li et al. [49] presented a hybrid malicious code detector based on deep learning. In the first step, the auto-encoder was used for decreasing the data dimensions, and the unsupervised DBN model was applied to discover network attacks. Chuan-Long et al. [50] proposed an IDS based on RNN to classify the collected data. The experiments were conducted on a different number of hidden nodes and learning rate values. The output of the techniques accomplished a reasonable performance with the parameters' setting of 80 hidden nodes and learning rate 0.1, but its computational processing was high. A flexible NIDA using a Self-Taught Learning (STL) algorithm was presented by Niyaz et al. [43]. The sparse auto-encoder was applied to represent a good feature representation while the soft-max regression technique was utilized for classifying the network data. The proposed model performed well performance in the evaluation process.

Tang et al. [51] built an IDS using a simple DFN which consists of three hidden layers, the model was trained and tested the best six features selected from the NSL-KDD dataset. Seok et al. [52] utilized the CNN technique-based IDS for recognizing malware. In [53], the author proposed an ensemble method for IDS using different DFN architectures that contain shallow auto-encoder networks, DBN, DNN, and an extreme learning machine. The method was evaluated on the NSL-KDD dataset, and the experiment results showed a good performance of detecting abnormal observations from network data.

From the discussion above, it is observed that deep learning techniques could considerably improve the performance of designing a reliable IDS for IICSs with higher detection accuracy and low false alarm rates. This is the motivation of utilizing deep learning models in this study, due to their ability of the automatic feature extracting with a depth analysis to network data and detecting outlier patterns from data as suspicious vectors. Our proposed DAE-DFFNN-based ADS technique contains a DAE algorithm to pre-train the DFFNN model that classifies network observations by ranking the parameter values of the ADE. It has the capability of discovering a good representation for network data and converting the high dimensional data to low dimensional using the decreased layer in the DAE-DFFNN model, as detailed in the following section.

## 3. Proposed ADS-based deep learning for IICSs

This study applies different architectures of deep-learning models to develop an efficient ADS for IIoT environments. In the training phase, a DAE algorithm learns using normal network observations to create the initialization parameters (i.e., weights and biases) and learn a deep representation of normal behaviors. These parameters are used as an initialization stage for training a standard Deep Feed Forward Neural Network (DFFNN) to discover existing and new attack instances. In the testing phase, the DFFNN is used to recognize malicious vectors. Different hidden nodes in the technique can professionally learn a deep feature representation and capture the most important features by converting the high dimensions of data to low dimensions based on the decreased hidden layer. The details of the proposed ADS technique are explained in the following three subsections.

### 3.1. Deep feed-forward neural network (DFFNN)

Typically, a DFFNN is defined as an ANN technique that has an input layer, more than one hidden layers and an output layer with direct connections without a cycle between them [30]. Each hidden layer of the nodes represents abstracted features based on the previous level's output which are automatically determined and collected in several layers to generate the outputs. To train this technique, a stochastic gradient descent back-propagation algorithm [42] is used.

In this deep-learning algorithm, the input data feeds into an input layer and is then propagated to the hidden layer, the output from which is a non-linear transformation of the data that passes to the output layer. A loss function or back-propagation error [31], which is the difference between the predicted and actual output, is calculated to evaluate the model's performance and its value propagated backward through the hidden layers to update the weights. Calculations of the loss function are conducted based on single or mini-batch samples of, rather than all, the training data, with the weights updated after each sample is processed in order to properly fit the model.

The supervised training process in this algorithm depends on the randomness of the initialization of the neural network's parameters which tends to place the model at local minima solutions with poor regularization [33]. To have better convergence properties and improve the results of supervised learning, pre-training

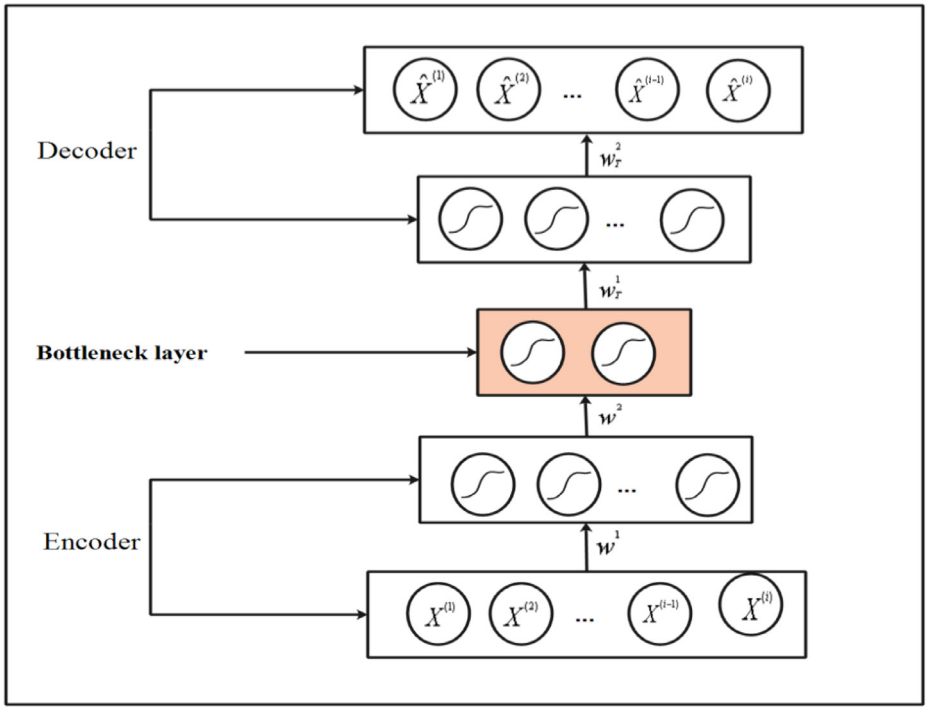


**Fig. 1.** Architecture of DAE network.

unsupervised techniques, in particular, an AE, can be used to create the initialization parameters.

### 3.2. Deep auto-encoder (DAE)

A DAE is a feed-forward neural network algorithm for learning efficient coding using an unsupervised technique [34]. It creates a representation of a set of data ($x$) by learning the approximation of an identity function, where the output ($\hat{x}$) is similar to the input ($x$), that is, $x \rightarrow \hat{x}$. Its schematic structure consists of vectors ($x^{(i)}$) in the input layer and more than one hidden layer with a non-linear activation function. The hidden layers are used to learn a compressed representation of the input data via fewer neurons than the input layer. As a result, it learns the most important features, reduces the dimensionality and represents an abstraction of the input data. Ultimately, the output layer ($\hat{x}^{(i)}$) is displayed as an approximate representation of the input layer.

The simplest architecture of an AE consists of an input layer, hidden layer, and output layer. Assuming that the training data ($x^{(i)}$) has $n$ samples, where each $x^{(i)}$ ($i \in (1, ...., n)$ has many dimensions, and there is a dimensional feature vector ($d^0$), the Tanh activation function is used [34] and computed by

$$\mathrm{T(t)} = \frac{1 - \mathrm{e}^{-2\mathrm{t}}}{1 + \mathrm{e}^{-2\mathrm{t}}}, \tag{1}$$

The AE algorithm has two main parts, an encoder and decoder [16,35]. To map the input vector ($x^{(i)}$) into a hidden layer representation ($z^{(i)}$), a deterministic mapping called an encoder process ($f_\theta$) is used [16] and the dimensionality of $x^{(i)}$ reduced to provide the correct number of codes as

$$f_\theta\left(x^{(i)}\right) = T\left(W_{x^{(i)}} + b\right) \tag{2}$$

where $W$ is a weight matrix of size $d^0 \times d^h$, $d^h$ a number of neurons in a hidden layer ($d^h < d^0$), $b$ the bias vector, $T$ a *Tanh* activation function and $\theta$ the mapping parameters [$W$, $b$].

To reconstruct the input as an approximation ($\hat{x}^{(i)}$), the result of the hidden layer's representation is mapped and the decoder process computed by the deterministic mapping ($g_{\theta'}$) as

$$g_{\theta'}\left(x^{(i)}\right) = T\left(W'_{z^{(i)}} + b'\right), \tag{3}$$

where $W'$ is a $d^h \times d^0$ weight matrix, $b'$ a bias vector and $\theta'$ the mapping parameters [$W'$, $b'$].

The input is formed in a compressed representation to fit the hidden layer, the data in which is then used as input to reconstruct the original data. The training process minimizes the reconstruction error (i.e., the difference between the original data and its low-dimensional reconstruction) and is calculated for a single or mini-batch training sample (s) by

$$E\left(x, \hat{x}\right) = \frac{1}{2}\sum_{i}^{s} ||x^{(i)} - \hat{x}^{(i)}||^{2}, \tag{4}$$

$$\theta = \{W, b\} = argmin_\theta E\left(x, \hat{x}\right), \tag{5}$$

The DAE architecture depicted in Fig. 1 that contains three hidden layers, an encoder, bottleneck (which consists of fewer nodes than the previous layers and is used to represent the input data with a non-linear dimensionality reduction, where the number of nodes represents the number of dimensions) and decoder. This model potentially operates using a non-linear Principal Component Analysis (PCA) technique for reducing dimensionality [36]. In order to identify legitimate and suspicious activities in IICSs, we propose an ADS based on deep learning that includes training and testing phases, as discussed below.

### 3.3. Training and testing phases of ADS-based deep learning

The DFFNN and DAE approaches discussed above are the fundamental mechanisms used to build the proposed ADS-based deep-learning technique; the structures of networks in DAE-DFFFN model are depicted in Fig. 2. In the training phase, given an unlabeled normal training dataset $A$, and labeled training dataset $B$, where $A \subsetneq B$, a DAE with a bottleneck layer is trained based on only normal records ($A$) without any anomalous vectors to learn and discover the most important feature representations for normal

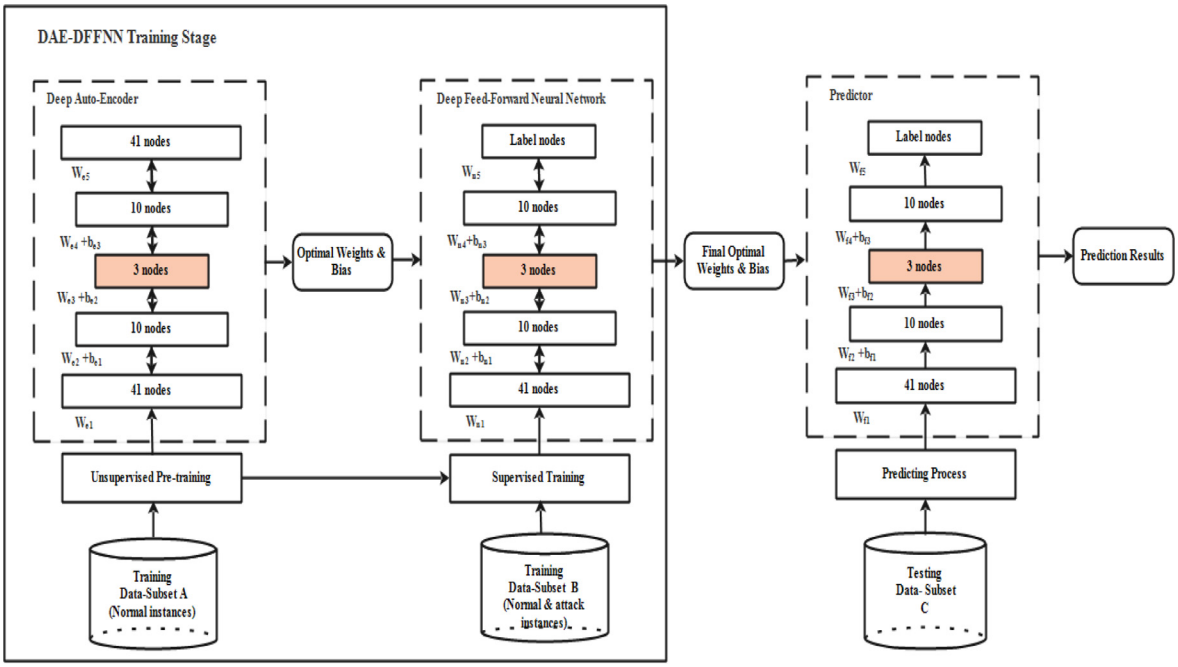


**Fig. 2.** Proposed architecture of DAE-DFFNN model based ADS for IICs.

**Algorithm 1:**
Unsupervised training phase for proposed ADS-based deep learning.

**Input**: training dataset ($A$) with n umber of samples ($n$) of ($x^{(i)}$), where $i \in (1, ...., n)$.
**Output**: parameters $\theta = \{W, b\}$
Begin
  Initialize $\{W, b\}$;
    repeat
      For each record ($x^{(i)}$), do
        compute the activation ( $z^{(i)}$) in/at hidden layer and
        give output ($\hat{x}^{(i)}$) to outhe tput layer.
        compute the training error ($E(x^{(i)}, \hat{x}^{(i)})$.
        Back-propagate $E$ and update parameters
        $\theta = \{W, b\}$;
    End
until converged
end

patterns. It is trained using all the data, where the input to the network ($x^{(i)}$) is passed through three hidden layers, including the bottleneck one, to reconstruct it ($\hat{x}^{(i)}$), where $W_e$, $W_n$ and $W_f$ are the weights of the DAE, DFFNN and final prediction model, as depicted in Fig. 2.

In the encoding step, the input layer in the first hidden layer is processed using Eqs. (1) and (2). In the bottleneck step, a low-dimensional non-linear transformation of the input features is executed to extract important representative features from the network data. Then, in the decoding step, the last hidden layer in the bottleneck feature is used to approximately replicate the input using Eqs. (1) and (3), and stochastic gradient descent back-propagation to reduce the loss function, that is, the mean square error between ($x^{(i)}$) and ($\hat{x}^{(i)}$) using Eqs. (4) and (5). The key procedures for unsupervised training of the proposed ADS-based deep learning are presented in Algorithm 1.

Then, the trained model is used as the starting point and initial parameters of weights and biases for training the supervised deep network and the process for tuning the network model conducted using the labeled training dataset ($B(x^{(i)}, y^{(i)})$). The same steps as previously followed are applied to learn and validate the proposed ADS technique on dataset B which includes normal and anomalous vectors that test the accuracy of the method. In more detail, the network model is trained based on the stochastic gradient descent back-propagation mechanism to minimize the loss function, with the mean square error calculated from the difference between the values of the target output ($y^{(i)}$) and predicted output ( $g_{\theta'}(x^{(i)})$), where $g_{\theta'}$ is the hypothesis function that yields an estimated output.

In the testing phase, after the parameters are automatically learned in the training phase, the new dataset sample ($C \subsetneq \{A, B\}$) is tested based on the final constructed network model. Each input record ($\hat{x}^{(i)}$) is passed to the input layer with the initialization weights and bias ($\theta_f = \{W_f, b_f\}$) adopted and then the input data is processed through the hidden layers. Finally, the output layer predicts the class of the input data as either normal or attack based on the estimated value of the loss function of each class.

## 4. Suggested framework for applying proposed ADS in IICSs

This study proposes an efficient anomaly detection model for protecting IICS environments against malicious activities. As illustrated in Fig. 3, its architecture consists of training and testing steps.

Data pre-processing, this includes feature transformation and normalization, and is the first stage in the proposed ADS-based deep-learning mechanism, inspects and selects important information from the large-scale data in an IIoT environment.

- **Feature transformation**—as the proposed model accepts only numerical features, each symbolic feature value is converted into a numerical one; for example, the NSL-KDD dataset has many symbolic attributes such as protocol types with nominal values like ICMP, TCP, and UDP which are mapped into 1, 2 and 3, respectively.
- **Feature normalization—**since deep learning depends on weights, the different feature scales can bias data into particular layers which may cause certain weights to update faster than others [32]. Consequently, it is necessary to handle this issue using a statistical normalization whereby the Z-score func-

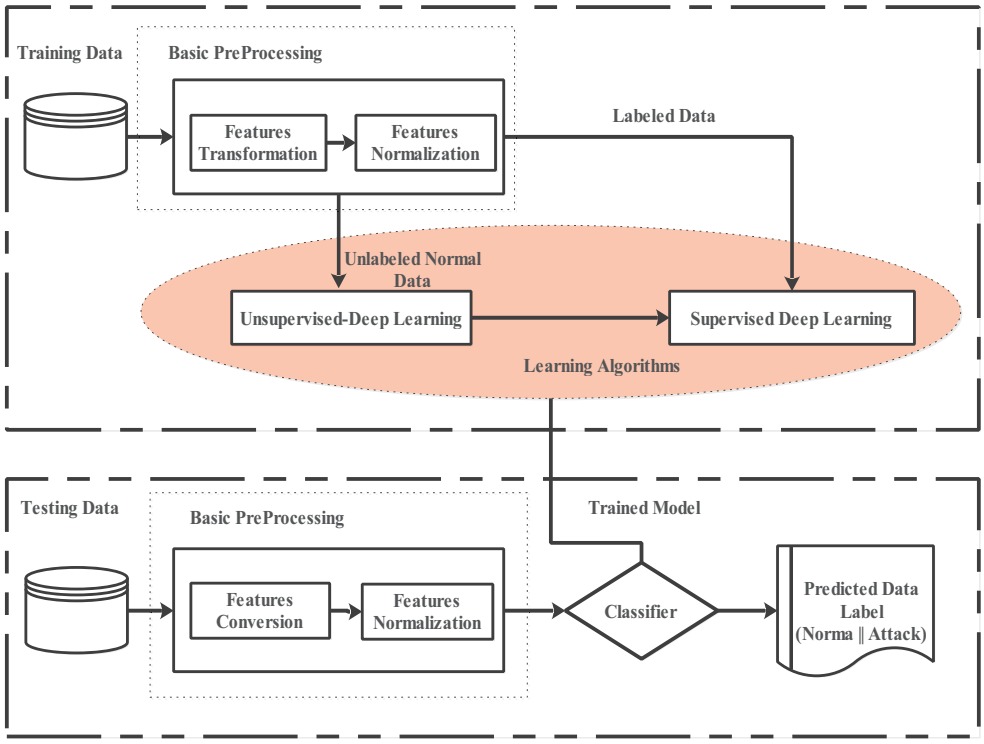


**Fig. 3.** Architecture of proposed ADS-based Deep Learning.

tion for each feature value ($v^{(i)}$) is performed using

$$Z^{(i)} = \frac{v^{(i)} - \mu}{\sigma} \tag{6}$$

where $\mu$ is the mean of the $n$ values for a given feature ($v^{(i)}$ ($i \in 1, 2, 3, \ldots n$)) and $\sigma$ the standard deviation.

As network data contains a high-dimensional space, it is essential reducing its dimensionality for improving the computational resources to design a lightweight and scalable ADS technique [36]. Consequently, the proposed DAE-DFFNN model is utilized to reduce the high dimensions into low ones using a central decreased layer. In more detail, there is a non-linear function in the Model that encodes a large number of features into the lower feature set in the decreased hidden layer, so feature reduction is applied without the need for human knowledge. The target of the DAE-DFFNN feature reduction is to eliminate from the ambiguous structure in the input distribution and find out well-designed representations in terms of higher-level learned, as well as importantly filtered and reduced features.

Since a suspicious activity is determined by any change in the normal state of the network, analyzing normal behaviors are so significant for facilitating the detection process. Therefore, an unsupervised learning process is applied to the normal data observations to estimate the initialization parameters of the weights and biases as a given input of the standard DFFNN for decreasing the processing time of building this model. The parameters are also retuned in the supervised deep-learning method using the labeled data (i.e., normal and malicious), with the final training model evaluated based on new data samples obtained during the testing phase, as explained in the aforementioned sections.

The placement of an IDS in the IIoT environment is critical for ensuring that the environment is secured against any malicious activities. As industrial systems transmit data for end-users and/or cloud storage through the IoT gateway which includes an internet–protocol connectivity (e.g., IP, TCP, UDP or HTTP), this gateway is a crucial location in which to deploy the proposed anomaly-IDS based on the deep-learning algorithm in an IIoT environment. Like other IDSs, the proposed system uses a set of cooperative and major components: a sniffing and monitoring unit; databases; and analyzer and response units. A general view of the structure of this IDS is presented in Fig. 4.

The main components of the proposed ADS are described below.

- **Sniffing and monitoring unit—**a sniffer, which is implemented in the gateway to monitor and collect the traffic exchanged between it and the external network via the internet can be embedded in either the software or hardware to obtain the sent and received packets which it stores in files that are then passed to the raw traffic database. Three databases, raw traffic, behavior, and log, implemented in our IDS are installed and retained in the cloud storage at the network's edge (i.e., fog computing storage). The first stores the raw network traffic collected by the sniffing unit, the second contains a list of previous datasets which is considered a profile history of the network while the third stores the new signatures of detected attacks and is used to continuously feed the behavior database.
- **Analyzer unit**—this is an important component in an IDS which consists of data processing and detection models.
  - ➢ **Data processing model**—since an enormous amount of data can be extracted from packets and processing it is extremely challenging in terms of the processing power, resources and time required, the raw data should be passed to this model to convert it to useful information. In it, the collected network traffic is analyzed and gathered according to the size of the time window or flow, such as the source and destination IPs, source and destination ports, and protocol type. Moreover, the basic features are extracted for the data flow and the data converted to a uniform format, a process handled directly by the proposed ADS based on the deep learning algorithm in order to reduce the data's dimensionality. Therefore, this processing model assists in the decision-

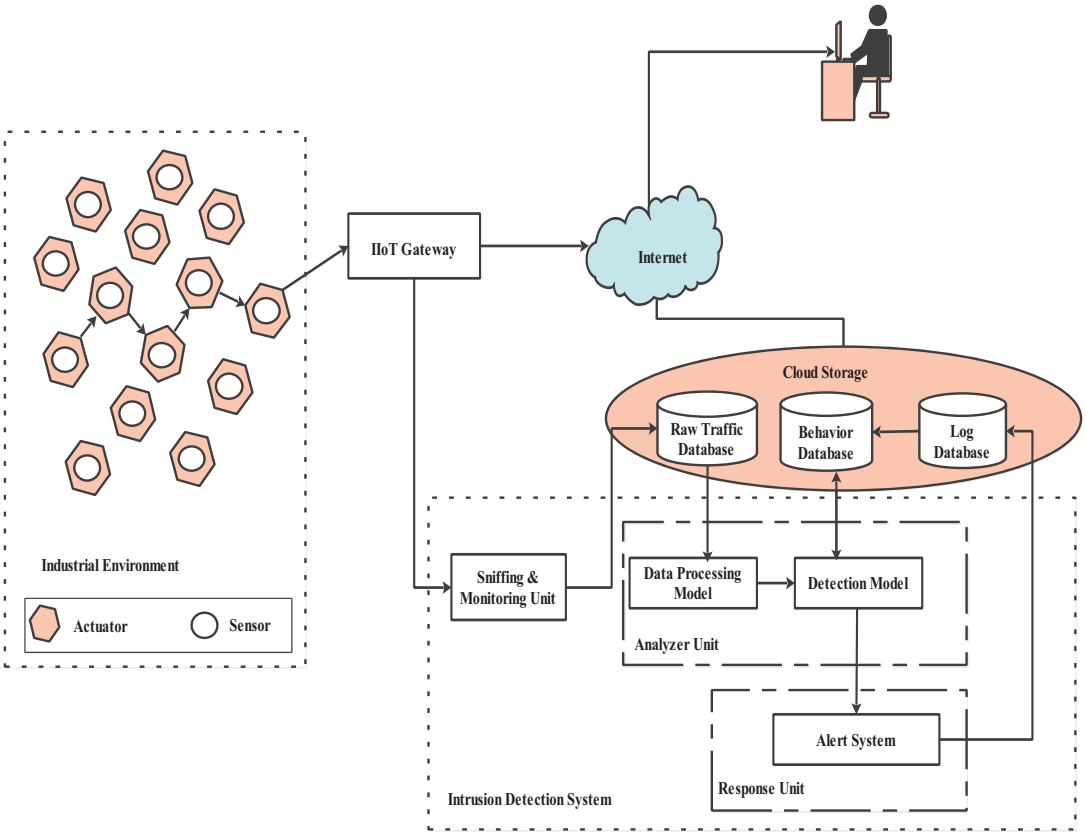


**Fig. 4.** Structure of deployment proposed ADS.

making process and prevents any bias and confusion for the anomaly detector.

➢ **Detection Model**—the result obtained from the processing model is sent to the detection model for a decision to be made for each data group, that is, the flow or window size. This model is used to detect known and unknown attacks by learning from the behavior database whereby, if any of the input data does not match normal network behavior, it is classified as an attack. Details of the detection process are discussed in Section 3.

- **Response unit—**this contains the alert system model and log database. The ADS alerts the system administrator to take the appropriate action when any abnormal activity is detected in the network, with the new signature for a specific attack type stored in the log database which, in turn, is used to feed the behavior database.

## 5. Descriptions of datasets and evaluation metrics

### 5.1. Datasets

Since a dataset plays a vital role in testing, analyzing and evaluating the behavior of a detection system, a good-quality one not only produces efficient results for an offline system but is also potentially effective when deployed in a real environment. Most researchers have used the popular NSL-KDD dataset, an amended version of the KDD CUP 99 one which solved the main problems of KDD CUP 99 by eliminating its redundant records and selecting numbers of records from it in proportion to their percentages. After pre-processing, it consists of 148,517 records (i.e., 77,054 normal and 71,460 attacks), each of which contains 41 features and a class label. There are five classes, namely, Probing, DoS, User to Root (U2R), Remote to Local (R2L) and Normal [37,38]. However, although having been used widely in IDSs, it is outdated [40].

Therefore, to effectively evaluate our proposed work, a new dataset called UNSW-NB15 is used. It reflects real modern normal behaviors and contains contemporary synthesized attack activities [39,44,45]. It has 257,673 records (i.e., 93,000 normal and 164,673 attacks), each with 41 features and a class label. There are ten different class labels, one normal and nine attacks, namely, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms.

### 5.2. Evaluation metrics

The proposed ADS is evaluated on the two datasets in terms of the accuracy, detection rates, and FPRs extracted from the confusion matrix terms; True Positive (TP) and False Negative (FN), which indicate the numbers of attack observations correctly identified as anomalous and incorrectly identified as normal, respectively, and True Negative (TN) and False Positive (FP) which show the numbers of normal observations correctly identified as normal and incorrectly identified as attack, respectively. The main evaluation metrics are calculated as follows.

- Accuracy identifies the total number of observations correctly identified with respect to the total number of observations and is calculated by

$$Accuarcy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{7}$$

- The Detection Rate (DR) is the ratio of attack to normal observations correctly classified and defined as

$$Detection\ Rate = \frac{TP}{TP + FN}, \tag{8}$$

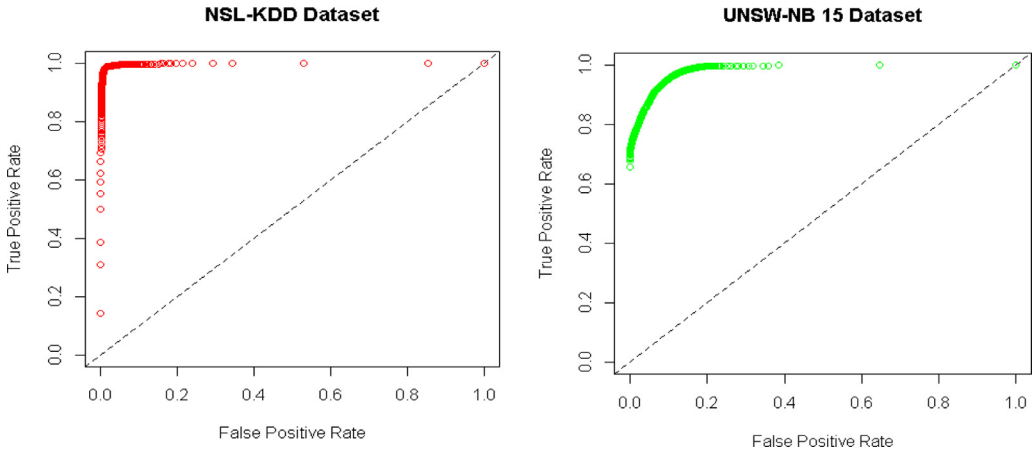


**Fig. 5.** (a) NSL-KDD dataset ROC curve; (b) UNSW-NB15 dataset ROC curve.

**Table 1**
Evaluation of performances for two datasets.

| Dataset | Accuracy | Detection rate | FPR |
|---|---|---|---|
| NSL-KDD | 98.6% | 99% | 1.8% |
| UNSW-NB15 | 92.4% | 93% | 8.2% |

- The False Positive Rate (FPR) is the ratio of attack to normal observations incorrectly classified and calculated by

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

## 6. Experimental results and discussion

### 6.1. Performance assessment and comparison

The proposed ADS technique is implemented and evaluated using the R programming language. The experiment is conducted on both datasets with all features and the important features are automatically adopted using the decreased layer of the DAE-DFFNN model. We use the full NSL-KDD dataset (i.e., 77,054 normal and 71,460 attack records) and different samples from the UNSW-NB15 one (i.e., 93,000 normal and 92,000 attack records), with 20% (which represents 40% of the normal records), 60% and 20% of the samples used for unsupervised learning, supervised learning and testing, respectively.

Based on the experiments, we adopted the networks structures and parameters that produce the highest DR and lowest FPR. The best network structures of the proposed model were used for both datasets as follows; one input layer (41 nodes), three hidden layers (10, 3, 10 nodes) and output layer (41 nodes) for the DAE model while the output layer with 2 nodes for the DFFNN model. The Tanh activation function was used, 100 epochs, a 0.002 learning rate, 2e-6 annealing rate, 0.2 momentum start, 0.4 momentum stable, momentum 1e7 ramp, L1 and L2 regularizations of L1 = L2 = 1e-6 for the UNSW-NB15 dataset, and a learning rate of 0.0015 and momentum start of 0.2 for the NSL-KDD dataset.

The results obtained from the proposed ADS model based on deep learning presented in Table 1 show that it performs better for the NSL-KDD than the UNSW-NB15 dataset.

In Fig. 5, the ROC curves indicate the performances of the proposed ADS for the two datasets in terms of the TPR (i.e., detection rate or sensitivity) and FPR (i.e., fall-out), with the areas under the red and green curves showing that its levels of accuracy for the NSL-KDD and UNSW-NB 15 datasets are 98.4% and approximately 92.5%, respectively.

The detection rates for the types of records in the NSL-KDD and UNSW-NB15 datasets using the proposed ADS based on the deep-learning technique are shown in Fig. 6. In the left-hand chart, the results for the NSL-KDD dataset demonstrate that the proposed model can determine the record types DoS, Normal, Probe, R2L and U2R with detection rates of 99.8%, 99.5%, 98.7%, 93.6% and 71.4%, respectively. The results for the UNSW-NB15 dataset in the right-hand chart show that the detection rates for the record types Analysis, Backdoor, DoS, Exploits, Fuzzer, Generic, Normal, Reconnaissance, Shellcode, and Worms are 83.3%, 91.8%, 95.1%, 96%, 60%, 99.5%, 98.9%, 96.8%, 81.1% and 76%, respectively. Although some results, such as for U2R in the NSL-KDD dataset and Fuzzer and Worms in the UNSW-NB15 one, are not high, the proposed model demonstrates overall good performances for detecting record types in both datasets.

### 6.2. Time cost for proposed ADS model

We analyze the proposed ADS model based on deep learning in terms of the total elapsed and CPU times it requires. As explained in Section 3, it consists of two main phases, training and testing. In the former, it is passed through two levels: firstly, unsupervised learning is used to obtain the initialization parameters (i.e., weights and biases) which, for the NSL-KDD dataset, takes approximately 194.37 seconds to process 30,783 records over 100 epochs, with the CPU requiring approximately 0.31 s and, for the UNSW-NB15 dataset, 227.30 seconds to process 37,280 records over 100 epochs, with the total CPU time 0.25 s; and secondly, the supervised learning takes approximately 194.67 seconds to pass 88,119 records 100 times, with the CPU requiring approximately 0.13 s for the NSL-KDD dataset and, for the UNSW-NB15 one, 119.03 seconds to process 110,776 records over 100 epochs and 0.14 s of CPU time.

In the testing phase, for the NSL-KDD dataset, the proposed model takes 2.25 seconds to predict 29,615 records, that is, approximately 13,162 records in one second and 0.06 s of CPU time and, for the UNSW-NB15 dataset, 55 seconds to predict 36,944 records, that is, approximately 6657 records in one second and 0.01 s of CPU time. It is worth mentioning that this time includes the task of reducing the data dimensions, learning and extracting the most crucial features which make our proposed model a practical solution for detecting intrusive activities in a real IIoT environment. The total elapsed and CPU times for the NSL-KDD and UNSW-NB15 datasets are presented in Table 2.

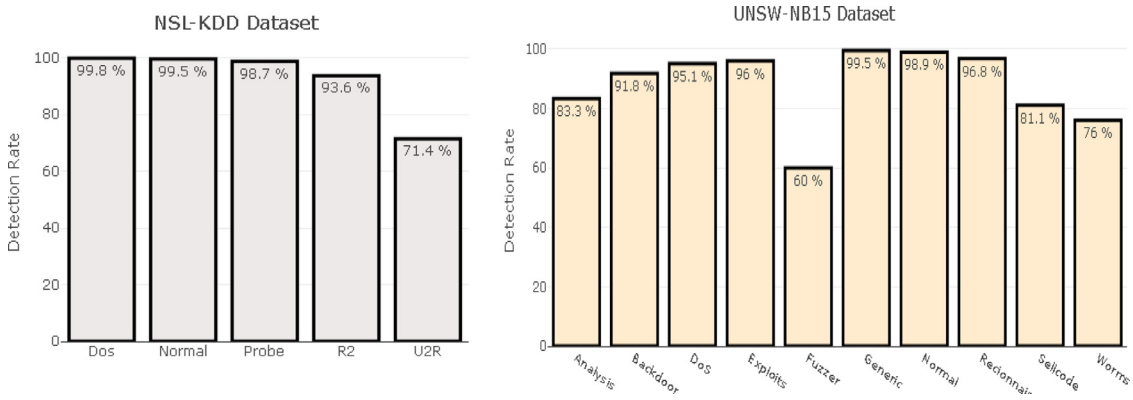


**Fig. 6.** (a) Detection rates for NSL-KDD dataset classes; (b) Detection rates for UNSW-NB15 dataset classes.

**Table 2**
Time cost (Elapsed and CPU time in seconds) for both datasets.

| Dataset | Unsupervised training phase (100 epochs) | | Supervised training phase (100 epochs) | | Testing phase | |
|---|---|---|---|---|---|---|
| | Elapsed time | CPU time | Elapsed time | CPU time | Elapsed time | CPU time |
| NSL-KDD | 194.37 | 0.31 | 194.67 | 0.13 | 2.25 | 0.06 |
| UNSW-NB 15 | 227.30 | 0.25 | 119.03 | 0.14 | 5.55 | 0.01 |

**Table 3**
Comparison of performances of five techniques for NSL-KDD dataset.

| Technique | Detection rate | FPR |
|---|---|---|
| F-SVM [13] | 92.2% | 8.7% |
| CVT [41] | 95.3% | 5.6% |
| DMM [40] | 97.2% | 2.4% |
| TANN [12] | 91.1% | 9.4% |
| DBN [46] | 95.1% | 4.5% |
| RNN [50] | 73% | 3.6% |
| DNN [51] | 76% | 15% |
| Ensemble-DNN [53] | 98% | 14.7% |
| Proposed Model | 99% | 1.8% |

### 6.3. Comparative study

To illustrate the effectiveness of our proposed ADS model, we compare its performance with those of eight recently developed anomaly detection techniques, namely, the Filter-based Support Vector Machine (F-SVM) [13], Computer Vision Technique (CVT) [41], Dirichlet Mixture Model (DMM) [40], Triangle Area Nearest Neighbors (TANN) [12], DBN [46], RNN [50], DNN [51], and Ensemble-DNN [53]. Table 3 demonstrates the result achieved by our proposed model compared with other models tested on NSL-KDD dataset in term of detection rate and false positive rate. It is very clear that our proposed model gets the best results with 99% DR and 1.8% FPR. The first four models achieved a rational performance in detecting the intrusive activities after a feature selection process. F-SVM adopted mutual information to address the linear and non-linear data features, and then they were used with the SVM for recognizing attacks. Nevertheless, this model still needs to optimize the search strategy to improve the efficiency of IDS. CVT and TANN used the PCA technique to reduce the data dimensions.

TANN used the K-means cluster based on only attack data to obtain good feature representations of attack behaviors. Both models rely on estimating the correlations between normal and abnormal instances, and this will face a problem in dealing with sophisticated attacks, which mimic the normal behavior, as they overlap with normal patterns. DMM utilized some statistical properties to specify the boundaries between normal and abnormal behavior. In this model, the PCA technique selected the best features, and the model achieved a high detection rate. However, this method needs to further statistical analysis for ensuring fitting all possible normal events that could happen in network production systems.

The last four models adopted different deep learning architectures in designing IDSs. The DBN and Ensemble-DNN techniques utilized the unsupervised pre-training model based on a greedy layer-by-layer learning algorithm. In DDBN model, to learn the weights between layers, each stack of RBM was trained at a time. After pre-training the two RBM, the parameters were tuned using the discriminative layer, which is trained on the labeled data for classification.

The ensemble–DNN model used the auto-encoder and extreme learning machine RBM to the DBN for learning data representations accurately. These models learned the hierarchical representation of the network data with high performance in terms of DR and FPR. However, they have the drawbacks of the computational costs related to training DBN and the optimization way of network structure based on the maximum-likelihood training approximation that is ambiguous. The RNN model which used a loop in its architecture performed less because of gradient and exploding gradient problems, but it works better in other domains, like noise cancellation and character recognition [55]. The final model is IDS-DNN which used only the basic concept of deep learning to detect intrusive activities after selecting the best six features in NSL-KDD dataset, here building the model based on the initial random weights and bias achieved fewer convergence properties and performance.

For discussing why our model performs better the above techniques, it automatically provides dimensionality reduction and feature extraction without human intervention, which takes a great deal of time and effort using most classical machine-learning algorithms, and has the capability to infer the unknown structure of normal network behavior through unsupervised and supervised training phases, thereby learning and obtaining good representations. Also, as previously discussed, it takes a suitable amount of processing time to train and predict data records. Furthermore, its

performance continually improves when the structure of the deep learning model is trained with more data.

The proposed model is different from the previous IDSs based on deep learning that it used the simple mathematical algorithm (DAE) for unsupervised learning which estimates the parameters in a suitable range that is the input of the supervised-DFFNN for building it effectively and efficiently. In addition, the model, through the decreased hidden layer, learns and explores the high-level features, reduce the dimensionality of data automatically, and represent the crucial features well. Therefore, these traits ensure that our proposed model is appropriate for deployment in a real industrial environment containing a massive amount of unlabeled and unstructured data.

### 6.4. Pros and cons of proposed anomaly-IDS based on deep learning

The proposed ADS has several advantages. Firstly, it can easily detect normal and attack behaviors in an IIoT environment as, because it is designed to identify normal behavior by training the model using normal behavior in the first training phase, based on the unsupervised deep learning algorithm, it can obtain good representations of normal traffic. Secondly, it conducts an additional training process based on normal and attack behaviors to power the system and ensure that it is capable of detecting sophisticated attacks. Thirdly, it provides an automated process for feature engineering which minimizes the time and effort required, and makes it more effective when deployed in a real environment. Finally, it depends on the parameters which are tuned in only the training phase and has a tolerance for noisy and outlier data.

Although a disadvantage of this model is that choosing its parameters for the training phase is not a trivial process, this is not considered a major problem given the current availability of complex and fast hardware. Also, while it cannot deal with non-numeric and the original range of features values, we overcome this issue by using feature transformation and normalization steps in the pre-processing stage.

## 7. Conclusion

In this paper, an ADS model for detecting intrusive activities in IIoT environments using the data collected from TCP/IP traffic is proposed. It uses deep-learning methods for unsupervised learning with automatic dimensionality reductions and a good representation of normal network patterns. It obtains powerful rather than random parameters for a supervised training for DFFNN. Then, to better tune these parameters, the supervised DFFNN is used. The proposed DAE-DFFNN model can successfully build and extract important features which enhance its performance overall. The final constructed model is tested on different data samples from the NSL-KDD and NSW-NB15 datasets, with the results revealing that it achieves the highest detection rate and fewest false alarms compared with some techniques developed in recent studies. In future, we will extend this work to train this algorithm on real data collected from IIoT systems to demonstrate the efficiency of its implementation.

Also, we present a perception for deploying this proposed ADS model based on the deep-learning algorithm in the real-world IIoT environment by: firstly, using it to collect the TCP/IP traffic using the sniffer implemented in the gateway; and, secondly, preprocessing and analyzing the collected data in order to reveal any intrusive activity. Further modifications and ideas for deploying and validating this model in a real environment, and extending this work to handle different protocols will be considered in future.

## Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.jisa.2018.05.002.

## References

[1] Sherasiya T, Upadhyay H, Patel H. A survey: intrusion detection system for internet of things. J. Comput Sci Eng 2016;5:91–8.
[2] Drath R, Horch A. Industry 4.0: hit or hype? [Industry forum]. IEEE Ind Electron Mag 2014:56–8.
[3] Shahzad A, Kim G, Elgamoudi A. Secure IoT platform for industrial control systems. In: Platform technology and service (PlatCon), international conference on. IEEE; 2017. p. 1–6.
[4] Katsikeas S, Fysarakis K, Miaoudakis A, Van Bemten A, Askoxylakis I, Papaefstathiou I, Plemenos A. Lightweight and secure industrial IoT communications via the MQ telemetry transport protocol. Symposium on computers and communications conference. IEEE; 2017.
[5] Stouffer K, Falco J, Scarfone K. Guide to industrial control systems (ICS) security. NIST special publication; 2011. p. 6–16.
[6] Atlantic Council. http://publications.atlanticcouncil.org/cyberrisks//.
[7] Sitnikova E, Foo E, Vaughn B. The power of hands-on exercises in SCADA cyber security education. In: IFIP world conference on information security education. Springer; 2009. p. 83–94.
[8] Abraham A, Grosan C, Martin-vide C. Evolutionary design of intrusion detection. Int J Netw Secur 2007:328–39.
[9] Modi C, Patel D, Borisaniya B, Patel H, Patel A, Rajarajan M. A survey of intrusion detection techniques in cloud. J Netw Comput Appl 2013:42–57.
[10] Tzokatziou G, Maglaras A, Janicke H, He Y. Exploiting SCADA vulnerabilities using a human interface device. Int J Adv Comput Sci Appl 2015:234–41.
[11] Kushner D. The real story of stuxnet. In: IEEE spectrum 50; 2013. p. 48–53.
[12] Tsai F, Lin Y. A triangle area based nearest neighbors approach to intrusion detection. Pattern Recognit 2010;43:222–9.
[13] Ambusaidi A, He X, Nanda P, Tan Z. Building an intrusion detection system using a filter-based feature selection algorithm. IEEE Trans Comput 2016;65:2986–98.
[14] N. Moustafa, and J. Slay. "A hybrid feature selection for network intrusion detection systems: central points." arXiv preprint arXiv:1707.05505 (2017).
[15] Kim J, Bentley P, Aickelin U, Greensmith J, Tedesco G, Twycross J. Immune system approaches to intrusion detection – a review. Nat Comput 2007:413–66.
[16] Hardy W, Chen L, Hou S, Ye Y, Li X. DL4MD: a deep learning framework for intelligent malware detection. In: Proceedings of the international conference on data mining (DMIN). The steering committee of the world congress in computer science, computer engineering and applied computing (WorldComp); 2016. p. 61–7.
[17] Huang W, Song G, Hong H, Xie K. Deep architecture for traffic flow prediction: deep belief networks with multitask learning. In: IEEE transactions on intelligent transportation systems; 2014. p. 2191–201.
[18] Nadiammai G, Hemalatha M. Effective approach toward intrusion detection system using data mining techniques. Egypt Inf J 2013:37–50.
[19] Mustafa N, Slay J. The evaluation of network anomaly detection systems: statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 dataset. Inf Secur J 2016:18–31.
[20] Shang W, Zeng P, Wan M, Li L, An P. Intrusion detection algorithm based on OCSVM in industrial control system. Secur Commun Netw 2016:1040–9.
[21] Maglaras A, Jiang J. Intrusion detection in SCADA systems using machine learning techniques. In: Science and information conference (SAI). IEEE; 2014. p. 626–31.
[22] Silva P, Schukat M. On the use of K-NN in intrusion detection for industrial control systems. In: 13th international conference on information technology and telecommunication; 2014. p. 103–6.
[23] Stewart B, Rosa L, Maglaras A, Cruz T, Ferrag M, Simoes P, Janicke H. A novel intrusion detection mechanism for SCADA systems that automatically adapts to changes in network topology. Ind Netw Intell Syst 2017:1–12.
[24] Shang W, Cui J, Wan M, An P, Zeng P. Modbus communication behavior modeling and SVM intrusion detection method. In: Proceedings of the 6th international conference on communication and network security. ACM; 2016. p. 80–5.
[25] Maglaras A, Jiang J. Ocsvm model combined with k-means recursive clustering for intrusion detection in scada systems. In: Heterogeneous networking for quality, reliability, security and robustness (QShine), 10th international conference on. IEEE; 2014. p. 133–4.
[26] Linda O, Vollmer T, Manic M. Neural network based intrusion detection system for critical infrastructures. In: Neural networks, international joint conference on. IEEE; 2009. p. 1827–34.
[27] Hodo E, Bellekens X, Hamilton A, Dubouilh L, Iorkyase E, Tachtatzis C, Atkinson R. Threat analysis of iot networks using artificial neural network intrusion detection system. In: Networks, computers and communications (ISNCC), international symposium on. IEEE; 2016. p. 1–6.
[28] Chen R, Liu M, Chen C. An artificial immune-based distributed intrusion detection model for the Internet of Things. In: Advanced materials research; 2012. p. 165–8.
[29] Van Dijk C, Williams P. The history of artificial intelligence. Expert systems in auditing Palgrave Macmillan UK; 1990. 21-16.

[30] Tang A, Mhamdi L, McLernon D, Zaidi R, Ghogho M. Deep learning approach for network intrusion detection in software defined networking. In: Wireless networks and mobile communications (WINCOM), international conference on. IEEE; 2016. p. 258–63.
[31] Svozil D, KvasniEka V, Pospichal J. Introduction to multi-layer feed-forward neural networks. Elsevier; 1997. p. 43–62.
[32] Recht B, Re C, Wright S, Niu F. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In: Advances in neural information processing systems; 2011. p. 693–701.
[33] Erhan D, Bengio Y, Courville A, Manzagol A, Vincent P, Bengio S. Why does unsupervised pre-training help deep learning? J Mach Learn Res 2010:625–60.
[34] Tao X, Kong D, Wei Y, Wang Y. A big network traffic data fusion approach based on fisher and deep auto-encoder. Information 2016:20–30.
[35] Lv Y, Duan Y, Kang W, Li Z, Wang Y. Traffic flow prediction with big data: a deep learning approach. IEEE Trans Intell Transp Syst 2015:865–73.
[36] Yousefi-Azar M, Varadharajan V, Hamey L, Tupakula U. Autoencoder-based feature learning for cyber security applications. In: Neural networks (IJCNN), 2017 international joint conference on. IEEE; 2017. p. 3854–61.
[37] Rathore S, Saxena A, Manoria M. Intrusion detection system on KDDCup99 dataset: a survey. Int J Comput Sci Inf Tech 2015.
[38] Tavallaee M, Bagheri E, Lu W, Ghorbani A. A detailed analysis of the KDD CUP 99 data set. In: Computational intelligence for security and defense applications. CISDA 2009. IEEE symposium on. IEEE; 2009. p. 1–6.
[39] Moustafa N, Slay J. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: Military communications and information systems conference (MilCIS). IEEE; 2015. p. 1–6.
[40] Moustafa N, Creech G, Slay J. Big data analytics for intrusion detection system: statistical decision-making using finite Dirichlet mixture models. In: Data Analytics and Decision Support for Cybersecurity. Springer; 2017. p. 127–56.
[41] Tan Z, Jamdagni A, He X, Nanda P, Liu P, Hu J. Detection of denial-of-service attacks based on computer vision techniques. IEEE Trans Comput 2015:2519–33.
[42] Li M, Zhang T, Chen Y, Smola J. Efficient mini-batch training for stochastic optimization. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM; 2014. p. 661–70.
[43] Niyaz Q, Sun W, Javaid A, Alam M. A deep learning approach for network intrusion detection system. In: Proceedings of the 9th EAI international conference on bio-inspired information and communications technologies (formerly BIONETICS). ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering); 2016. p. 21–6.
[44] Moustafa N, Slay J. The evaluation of network anomaly detection systems: statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. Inf Secur J 2016;25:18–31.
[45] Moustafa N, Slay J. The significant features of the UNSW-NB15 and the KDD99 data sets for network intrusion detection systems. Building analysis datasets and gathering experience returns for security (BADGERS), 2015 4th international workshop on. IEEE; 2015.
[46] Alom MdZ, Bontupalli V, Taha. Intrusion detection using deep belief networks. In: Aerospace and electronics conference (NAECON), 2015 national. IEEE; 2015. p. 339–44.
[47] Gao N, Gao L, Gao Q, Wang H. An intrusion detection model based on deep belief networks. In: Advanced cloud and big data (CBD), 2014 second international conference on. IEEE; 2014. p. 247–52.
[48] Abolhasanzadeh B. Nonlinear dimensionality reduction for intrusion detection using auto-encoder bottleneck features. In: Information and knowledge technology (IKT), 2015 7th conference on. IEEE; 2015. p. 1–5.
[49] Li Y, Rong M, Runhai J. A hybrid malicious code detection method based on deep learning. Int J Secur Appl Methods 2015;9(5).
[50] Chuan-long Y, Yue-fei Z, Jin-long F, Xin-zheng H. A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access 2017:1–7.
[51] Tang T, Mhamdi L, McLernon D, Zaidi S, Ghogho M. Deep learning approach for network intrusion detection in software defined networking. In: Wireless networks and mobile communications (WINCOM), 2016 international conference on. IEEE; 2016. p. 258–63.
[52] Seok S, Howon K. Visualized malware classification based-on convolutional neural network. J Korea Inst Inf Secur Cryptol 2016;26(1):197–208.
[53] Ludwig S. Intrusion detection of multiple attack classes using a deep neural net ensemble. 2017 IEEE symposium series on computational intelligence; 2017.
[54] E. Hodo, B. Xavier, H. Andrew, T. Christos, and A. Robert "Shallow and deep networks intrusion detection system: a taxonomy and survey." arXiv preprint arXiv:1701.02145 (2017).
[55] Z, Lipton, J. Berkowitz, and C. Elkan. "A critical review of recurrent neural networks for sequence learning." arXiv preprint arXiv:1506.00019 (2015).
[56] T. Marsden, N. Moustafa, E. Sitnikova, and G. Creech, G. (2017). Probability risk identification based intrusion detection system for SCADA systems. arXiv preprint arXiv:1711.02826.