Journal of Network and Systems Management (2023) 31:33
https://doi.org/10.1007/s10922-023-09722-7

Deep-Learning Based Detection for Cyber-Attacks in IoT Networks: A Distributed Attack Detection Framework

Olivia Jullian[1] · Beatriz Otero[1] · Eva Rodriguez[1] · Norma Gutierrez[1] · Héctor Antona[1] · Ramon Canal[1]

Received: 23 April 2022 / Revised: 17 November 2022 / Accepted: 20 January 2023

Abstract

The widespread use of smart devices and the numerous security weaknesses of networks has dramatically increased the number of cyber-attacks in the internet of things (IoT). Detecting and classifying malicious traffic is key to ensure the security of those systems. This paper implements a distributed framework based on deep learning (DL) to prevent many different sources of vulnerability at once, all under the same protection system. Two different DL models are evaluated: feed forward neural network and long short-term memory. The models are evaluated with two different datasets (i.e.NSL-KDD and BoT-IoT) in terms of performance and identification of different kinds of attacks. The results demonstrate that the proposed distributed framework is effective in the detection of several types of cyber-attacks, achieving an accuracy up to 99.95% across the different setups.

Keywords Attack detection · Cyber-security · Deep learning · Distributed framework · Feed forward neural network · Long short-term memory

✉ Beatriz Otero
beatriz.otero@upc.edu

Olivia Jullian
oliviajullianparra@gmail.com

Eva Rodriguez
eva.rodriguez@upc.edu

Norma Gutierrez
normaescg@gmail.com

Héctor Antona
hector.antona@gmail.com

Ramon Canal
ramon.canal@upc.edu

[1] Department of Computer Architecture, Universitat Politecnica de Catalunya, Jordi Girona, 31, Barcelona 08034, Spain

1 Introduction

Nowadays, IoT networks comprise billions of smart devices that communicate with each other requiring minimal human intervention. It is estimated that the number of IoT devices will grow in 2022, expecting 46 billion devices by the end of this year. IoT technologies are crucial in real-life smart applications within the smart industry and smart cities. They have become the perfect solution through the creation of smart applications into all the domains including medical, industrial, education, habitat etc. IoT extends the power of the Internet beyond computers to a whole range of processes and environments. End-users put everything that can be monitored into the network to gather and send information back or both. With this new concept of the system, the Internet now embeds over 99% of the objects and environments that traditionally remained beyond the reach of the network. The increased use of IoT devices makes it a fertile ground for malicious actors to conduct cyber-attacks. In 2019, a network of honeypots put in place by Kaspersky found that despite the low sophistication of the attacks, they remained unnoticed by the user until the victim was activated as part of a botnet. In 2020, the leading cyber-attacks in IoT were worms, bots, and Distribute Denial of Service (DDoS), with as many as 16 different types [1]. While in 2021, IoT cyber-attacks more than doubled according to Kaspersky. The current limitations of IoT devices attract malicious actors towards the IoT ecosystem. IoT devices have limited resource capacity (i.e., computation and storage resources), thus, complex security mechanisms that require large memory or data are not applicable. This lack of security software (or minimal), makes IoT devices more vulnerable. In addition to the limited resources, there is a massive increase of IoT devices and, considering their constant connection to the IoT network, these devices have favorable conditions to spread rapidly an attack. In order to detect a network intrusion in an IoT device, the device needs to fulfill the security requirements that secure an IoT system from any kind of thread. With this new IoT paradigm the conventional security goals: confidentiality, integrity and availability (CIA) are not sufficient and fail in addressing novel threats [2]. The new security requirements to consider are: accountability, auditability, privacy and trustworthiness. Traditional cyber-security systems protect users and devices through Intrusion Detection Systems (IDS) user authentication, data encryption, firewalls, and anti-virus software. As shown in Kwon et al. [3], the use of Machine Learning (ML) techniques [4, 5] to detect malicious network traffic, abnormal behaviors and attempts in computer systems in an IDS is not enough. Yet, classic ML lacks automatic feature engineering, they have low detection rate [6], and they are not efficient in detecting small variants of existing attacks. This has led to consider DL techniques to improve cyber-security systems.

DL is an ML sub-field that has gained great recognition in many areas due to its improvement in accuracy in complex tasks and recent developments in hardware and software. DL techniques improve cyber-security systems preventing attacks by identifying patterns that are different from normal behavior [7]. Cyber-attacks share a common feature with image recognition, since more than 99% of the new attacks are small mutants of existing ones; in the same way that changes in images can be identified by small changes in their pixels. In IoT-Fog networks, they are used for detecting network threads and attacks [8, 9]. Even though IoT network features (i.e., its distributed nature and the limited computing capabilities of the end-devices) requires novel solutions for IDS [10].

This work proposes a novel reliable system which applies DL approaches to target several kinds of attacks in IoT networks. The contributions of this work are as follows:

- Design and development of a novel distributed DL-based attack detection framework in IoT networks.
- Preprocessing of the BoT-IoT and NSL-KDD datasets to achieve a higher accuracy of the framework.
- Comparison of FFNN and LSTM models to select the best model for a wide range of cyber-attacks.
- DL model tuning using Hyperband to improve detection rates.

he remaining of this paper is structured as follows. Section 2 reviews the DL based attack detection state-of-the-art in IoT networks. Section 3 presents the distributed attack detection framework developed, the DL models and the datasets used. Section 4 describes the design of the distributed framework, as well as of the neural networks parameters. Section 5 evaluates the distributed attack detection framework. Finally, Sect. 6 concludes the paper and it presents the future work.

2 Related Work

This section discusses the related works that use DL for intrusion detection in IoT. The emergence of Fog computing ecosystems requires new DL models to be resilient, adaptive, and closer to the edge. Edge nodes already provide computing, storage, communication and control services. Similarly, they can host security services. Attacks in IoT systems range from gaining access to the local system to DoS. Initial works in IoT environments consider centralized solutions for DL based intrusion detection systems. Alom et al. [11] develop the first IDS based on DBN. They test the solution using the NSL-KDD dataset normalized through a numerical encoding procedure. For the classification phase, the system achieves a 97.5% ACC versus the 40% of the dataset training data. Kim et al. [12] focus on the detection of DDoS attacks. They chose LSTM to detect network intrusions. They succeed in detecting and predicting complex patterns in a time series by detecting anomalies from a temporal pattern of less reliable symptoms. This work uses the KDD Cup'99 dataset and it achieves an accuracy of 98.8%. Shone et al. [13] propose a solution based on stacked NDAEs and Random Forest as the classification algorithm. They use the KDD Cup'99 and NSL-KDD datasets and they obtain similar results as previous approaches (98% ACC). Kwon et al. [3] chose the MLP model for evaluation. They use the NSL-KDD dataset preprocessed by normalizing numerical values and encoding categorical values as numerical ones. They train the network with different hyper-parameter configurations (i.e. units, hidden layers, epochs and learning rate), and the *Softmax* layer produces the outputs. Authors obtain promising results with an ACC of 90%. Finally, Ferrag et al. [14] conduct a comparative study for different DL models in intrusion detection systems. They consider Recurrent Neural Network (RNN), Deep Neural Network (DNN), Restricted Boltzmann Machine (RBM), Convolutional Neural Network (CNN), Deep Boltzmann Machine (DBM) and Deep Autoencoder (DAE), using two datasets CSE-CIC-IDS2018 [15] and the Koroniotis version [16] of the BoT-IoT, which considers a 5% of the entire dataset for training and testing. The study concludes that CNN and DAE methods achieve the best accuracies, 97% and 98% respectively, obtaining best performance for both datasets.

Subsequent works conceive distributed solutions to prevent network attacks. The first one is conducted by Yadav and Subramanian [17], which propose a solution for detecting DDoS attacks in the application layer through traffic classification using SAEs. They construct their dataset from features extracted from their web server log (request flooding, session flooding, asymmetric attack). The logs are pre-processed and the features are transformed to a numeric form. The system first learns the features through the SAE, then the DDoS dataset is built based on these features. At the end, they are classified with a logistic regression classifier. The experimental results demonstrate that the proposed method learns features from the SAE, which are beneficial for classifications. It improves the DR to 98.99% with an average FPR of 1.27%. Lopez-Martin et al. [18] propose a system based on CVAEs, which uses a detection method that also performs feature reconstruction, recovering missing features from incomplete datasets. They use a deviation-based approach, but with a discriminative framework for traffic samples and classification. Traffic samples are labeled with the intrusion that achieves less reconstruction error instead of using a threshold for intrusion definition. Then, the intrusion labels are included inside the CVAE decoded layers. The method is tested using the NSL-KDD dataset, and it can recover missing categorical features with 3, 11 and 70 values, achieving an ACC of 99%, 92% and 71% respectively. Luo and Nagarajan [19] base their solution on AEs for wireless sensor networks (WSN), overcoming the high computation resource consumption of DL in WSN. The authors design a two-part algorithm which resides on sensors detecting anomalies in a fully-distributed manner. The training model is offloaded to the cloud. They create their dataset by collecting data over 4 consecutive months in a real WSN indoor test-bed. The results show high accuracy and a low false positive rate (AUC > 0.8 in most cases). In this work, the sensors perform distributed anomaly detection without communicating to each other. They evaluate the TPR and FPR. TPR is better (18%) for a random scheme because the majority of the training data is historical, while the prioritized scheme has a much lower FPR, up to 60%, because it updates the weights and biases more responsively. Diro and Chilamkurti [8] propose a distributed architecture for IoT networks where the fog nodes are responsible for training models and hosting attack detection systems, while the master nodes conduct collaborative parameter sharing and optimization. In this architecture the DL system uses three layers of neurons (150, 120 and 50), and the last *Softmax* layer with neurons equal to the number of classes. The model is tested using the NSL-KDD dataset and the results demonstrate that the distributed model performs better than the centralized one improving overall accuracy of detection from 96 to 99%. Finally, Roopak et al. [9] analyze four different classification DL models MLP, CNN, LSTM, and CNN+LSTM, obtaining the best accuracy results for the last one (97.16%). The system uses the CICIDS [20, 21] dataset but balances the DDoS attack dataset by duplicating the data, which improves the training of the DL methods.

Table 1 summarises the surveyed DL models used for intrusion detection in IoT networks, considering both centralized and distributed architectures. Our work differentiates from them in (1) the distributed nature of the attack detection scheme, as well as, (2) the pre-processing of the datasets. This improves accuracy in the detection of attacks. Our proposal an accuracy up to 99.95% using the FFNN model. Moreover, our work is not restricted to a specific environment (e.g. wireless sensors networks), it covers a wide range of cyber-attacks, and it offers flexibility in changing the IoT fog node and environment without risking the device security.

3 Distributed DL-Based Attack Detection Framework

Nowadays, IoT networks need a distributed solution at the edge for cyber-attack detection. In fact, the distributed nature of IoT environments require re-engineering of intrusion detection services. Traditional centralized IDSs have shown to be ineffective in the prevention of novel (zero-day) attacks. This section presents a novel distributed DL-based attack detection framework. It consists of four main stages: Data treatment and pre-processing, DL model training and testing, distributed framework deployment and attack detection and classification (see Fig. 1).

This section first presents the datasets considered, as well as the pre-processing conducted. In order to achieve a broad solution, two different datasets are considered: the BoT-IoT dataset that addresses attacks specific of IoT environments, and the NSL-KDD dataset to broaden the types of cyber-attacks. Both datasets are preprocessed to be later used in the NNs. The second part of this section addresses a

Table 1 Key contributions using DL models for intrusion attack detection in IoT networks

Related work (year)	DL models	Datasets	Metrics
Alom (2015) [11]	DBN	NSL-KDD	ACC (97.5%)
Kim (2016) [12]	LSTM to RNN	KDD Cup'99	ACC (98.8%)
Shone (2018) [13]	Stacked NDAE	KDD Cup'99, NSL-KDD	ACC (98%)
Kwon (2019) [3]	MLP	NSL-KDD	ACC (90%)
Ferrag (2020) [14]	CNN, DAE	Bot-IoT, CSE-CIC-IDS2018	ACC (97–98%)
Yadav (2016) [17]	SAE	Custom	ACC (99.5%)
Lopez-M (2017) [18]	CVAE	NSL-KDD	ACC (99%)
Luo (2018) [19]	AE	Custom	TPR (> 80%)FPR (< 38%)
Diro (2018) [8]	Stacked AE	NSL-KDD	ACC (96–99%)
Roopak (2019) [9]	CNN and LSTM	CICIDS	ACC (97.2%)

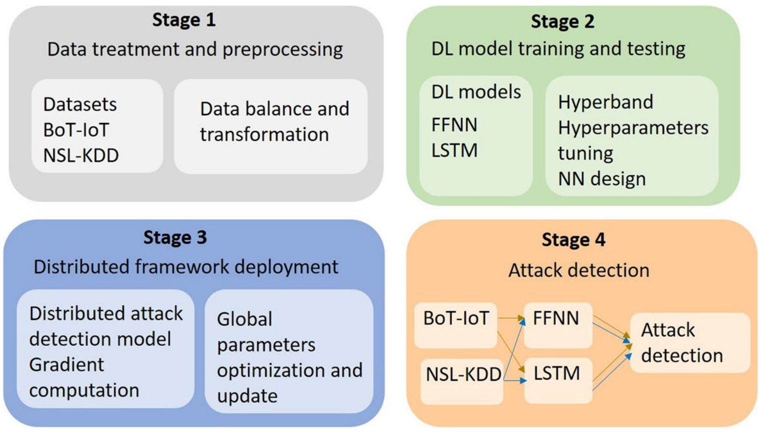
Stage 1
Data treatment and preprocessing
Datasets
BoT-IoT
NSL-KDD
Data balance and transformation
Stage 2
DL model training and testing
DL models
FFNN
LSTM
Hyperband
Hyperparameters tuning
NN design
Stage 3
Distributed framework deployment
Distributed attack detection model
Gradient computation
Global parameters optimization and update
Stage 4
Attack detection
BoT-IoT
NSL-KDD
FFNN
LSTM
Attack detection

Fig. 1 Attack detection framework stages

key issue for the proposed framework, that is the selection of the appropriate DL model. The two models considered are the FFNN and LSTM because they are used for supervised learning and they have good deployment with highly correlated features. Both networks are evaluated in a centralized model. Then, the best performing DL model is used in the distributed framework. Finally, the proposed distributed framework is presented, and its performance assessed in a realistic disperse environment where data is limited.

3.1 Datasets

This paper aims to provide a robust framework effective in the detection of IoT cyber-attacks. Thus, we use both the BoT-IoT (a specific IoT cyber-attacks dataset) and NSL-KDD (general cyber-attack dataset).

3.1.1 BoT-IoT Dataset

The BoT-IoT dataset [16] consists of over 73 million records of network activity in a simulated IoT environment, including both normal and several cyber-attack traffic flows. The training and test dataset have 5 output classes reflecting either normal traffic or 4 different types of attacks: DDoS, Denial of Service (DoS), Keylogging and Data Theft. DoS and DDoS attacks address malicious attempts to disrupt normal traffic of a server, service or network overloading them with a flood of Internet traffic. Keylogging represent attacks that gather information. Finally, data theft represents private user information leak. The BoT-IoT dataset is based on the activity of a network composed by 62 hosts (based in the network mask 192.168.100.0/26). Beyond the scenario presented in the BoT-IoT dataset, in this paper the network traffic is considered for any network size since the increment of IoT devices and the appearance of new networks types (i.e. 5G) generates many possible network sizes and therefore scenarios. All the different attacks of the dataset are based on flooding.

Therefore, the more relevant features are those which provide information about the rates, the number of packets and types of protocol used by the IoT device.

3.1.2 NSL-KDD Dataset

The NSL-KDD dataset [22] is an improvement of the previous KDDCUP'99 dataset, widely used for anomaly detection. The NSL-KDD dataset [23] consists of over 1.152.281 records, each one composed by 41 features. These features are divided in: basic, content, time and host traffic characteristics. It comprises normal traffic and four different types of attacks: Probing, DoS, User-to-root (U2R) and Remote-to-local (R2L). Probing attacks gather information about computer networks discovering vulnerabilities to finally circumvent security controls. In U2R attacks, the attacker sends packets to a machine over a network, locating a vulnerability in the machine and gaining access. Finally, in R2L attacks, the attacker begins accessing a normal user account on the system, to latter gain root access. Table 2 provides a global overview of the BoT-IoT and NSL-KDD datasets. Both datasets are unbalanced in terms of normal traffic and attacks. Consequently, a balancing pre-processing is performed, as described in the next section, to achieve the final dataset structure depicted in this table.

3.2 Data Pre-processing

This section presents the first stage of the framework. Data pre-processing is performed before the DL models are trained. The appropriate pre-processing of network traffic allows DL models efficiently predict non biased results, while a correct pre-processing avoids overfitting problems. The pre-processing is divided in two steps: features selection and data processing.

3.2.1 Features Selection

The features selected for the BoT-IoT are those which provide information about the rates, the number of packets and the types of protocol used by the IoT device. The reason of this selection resides in the characteristics of the attacks present in these devices (explained in the previous section).

In the case of NSL-KDD dataset, features are divided into different categories: basic, content, time and host traffic. In this way, all the characteristics considered for the BoT-IoT dataset from content, time and host traffic characteristics (as the number of packets) are also shared by the NSL-KDD dataset, the common nature of most of

Table 2 Datasets structure

Dataset	Total num. samples	Records after balancing	Attacks/normal	Training/testing
BoT-IoT	73 million	19,348	50%/50%	80%/20%
NSL-KDD	1.152.281	29,701	50%/50%	70%/30%

the attacks present in both datasets. An example of shared features are those referred to duration, protocol-type, rates and number of bytes received and send.

Because of the similarity in the attacks (for instance DoS), most of the characteristics are shared between both datasets. On the other hand, NSL-KDD dataset exploits network cyber-attacks with different characteristics. This is the case of probing attacks. These attacks are based on network vulnerabilities, not only the device itself. Thus, its targets are broader than the attacks considered for the BoT-IoT dataset (mainly focused on the device). Therefore, the framework uses additional features related to the basics of the network such as urgent packets, service, logged-in.

3.2.2 Data Processing

Regarding the content of the dataset records, both datasets are unbalanced since the number of records is far higher for normal traffic than for attacks. Different techniques can be used to balance the data based on oversampling and undersampling techniques. The large number of records lead us to consider undersampling techniques. The datasets are balanced reducing the number of normal traffic records (randomly picked), while maintaining the number of attacks. Hence, we synthesized a subset of the dataset holding a parity of 50% attacks and 50% not attacks.

Different techniques are used before training the models to prevent overfitting problems. After balancing both datasets, all the features are transformed into floating point values. This means that, in the BoT-IoT dataset, all the IP addresses are treated as different numbers, thus, effectively considering all network sizes. This transformation is also shown when applying one hot encoding to the categorical values. Feature selection is not required due to the reduced number of features of the dataset. Finally, a standard normalization of the dataset is done in both scenarios to prevent DL model overfitting and possible biased results.

3.3 Neural Network Models

Once the data is pre-processed, DL model training and testing, the second stage of the framework, starts. DL enables computational models to learn in different stages of representation and abstraction to effectively model complex relationships among input data. DL algorithms work by transforming one module to a more abstract one, always starting from the pre-processed raw data. The combination of multiple levels allows learning high complex models, which are often used for classification tasks, considering the input features relevant for discrimination and suppressing minor variations. Hence, two DL models, FFNN and LSTM, are considered in the proposed framework. They have been chosen because of two main reasons: i) their extensive use in supervised learning; ii) their good performance with highly correlated features. Furthermore, FFNN has been demonstrated a simple and efficient solution for intrusion detection with excellent results in previous studies, achieving 98% of accuracy.

On the other hand, LSTMs have evolved from Recurrent Neural Networks (RNNs) to efficiently learn patterns in long sequences [8] (e.g. network data classification as normal traffic or an attack). They are effective in training on unstructured datasets such as the IoT ones. LSTMs are also a standard solution for malicious activity detection since they consider the time constrains of IoT devices. Moreover, the time flag and sequential features present in both datasets allow LSTM models to take into account past values and search for attack patterns, which is helpful for intrusion detection. Finally, as shown in Fig. 1, stage 4 is the attack detection. Both DL models, FFNN and LSTM, are used in the proposed framework for the attack detection in IoT environment (BoT-IoT dataset) and more global networks (NSL-KDD dataset).

3.3.1 FFNN

FFNN [24] is a widely used Artificial Neural Network (ANN) model, where information moves from the input to the output layer through the hidden layer, only in the forward direction. The number of neurons in the input and output layers are determined by the number of input and output variables, respectively, while in the hidden layer they are determined by a trial and error method. The three layers of the FFNN are fully connected because each neuron of one layer is connected to all the other neurons of the next layer. In real world problems, the network is established by a model that consists of many neurons.

In cyber-attack detection, the output of the FFNN model determines whether there is an attack or not. Figure 2 shows the structure for the FFNN model applied to the attack detection in IoT environments using the BoT-IoT dataset.

3.3.2 LSTM

LSTMs [25] are a particular type of Recurrent Neural Networks (RNNs) used to gather information from multiple layers, contrary to FFNNs. RNN is the preferred model for training sequential data. This type of NN is an extension of a conventional

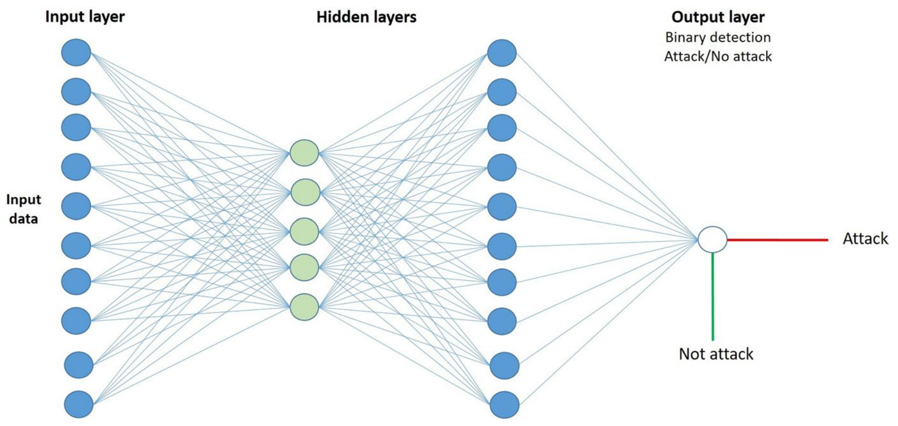

Fig. 2 FFNN model for attack detection (BoT-IoT dataset)

FFNN with cyclic connections. RNNs are more powerful in modeling sequences, but they have gradient vanishing and exploding problems, which make them particularly hard to train. LSTMs overcomes these problems introducing the LSTM cell formed by a set of gates to control the information flow. LSTMs use the connections between hidden units to generate their output and model their weights. Three gates control the information flow: the input, the forget, and the output gates. The input gate determines the ratio of the input, which affects the calculation of the cell state. The output gate determines if the information from a previous train is gathered or discarded and passes the decision to the forget gate. Figure 3 depicts the structure for the LSTM model applied to attack detection in IoT environments using the BoT-IoT dataset. The structure is the same as in Fig. 2, but with an LSTM distribution.

3.3.3 Overfitting Prevention

The use of DL techniques, specifically LSTM and FFNN models, has proven extremely accurate in the past years (i.e. 97%–98% of accuracy for the Bot-IoT dataset [14]). DAEs and AEs (a specific case of FFNN) have also performed well, with more than 95% accuracy for datasets with similar characteristics [5].

Due to the tendency of DL in overfitting, we include specific measures in the development of our framework to ensure realistic results as follows.

Section 3.2 uses undersampling techniques for the correct balance the data. These techniques allow the models to achieve realistic environments and equiprobability for each of the attacks present in IoT devices. In addition to the balance of the data, we consider the IP addresses as floats, to represent all types of networks.

The use of dropouts in some of the DL models to prevent unrealistic high performances. This step is described in Sect. 4.1.

However, the most relevant prevention against overfitting presented in this paper is the use of Hyperband [26], a TensorFlow tool that allows the correct fine tuning

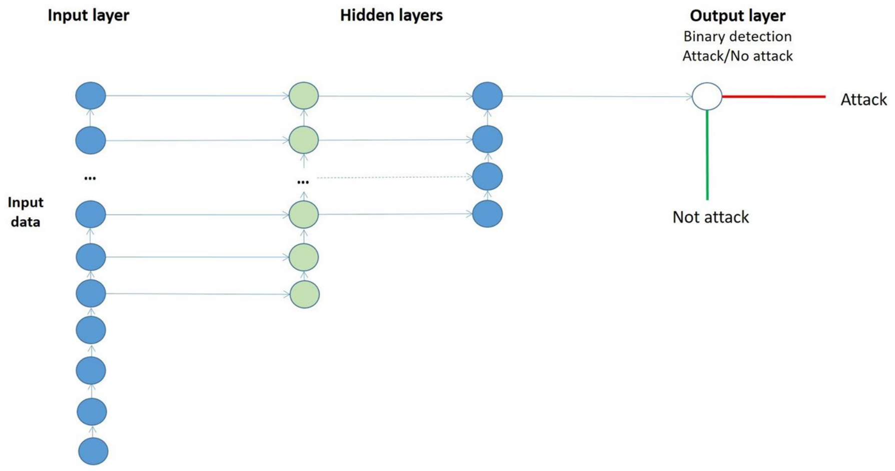

Fig. 3 LSTM model for attack detection (BoT-IoT dataset)

of the model. This mechanism allows the research of the best hyperparameters combination to achieve the highest performance of the model. During the research, the training and validation losses are the main metrics considered in the process of fine tuning. Thus overfitting can be avoided. A more detailed explanation can be found in section 4.

Finally, removing IP addresses during the pre-processing of the datasets is another possibility that is considered to avoid overfitting. However, not the same IP attackers are used in the training and validation set. This means, the model cannot learn by heart the float numbers corresponding to attackers' IP address, since there is no repeated addresses. In addition, we find that the understanding of the IP attackers' nature is crucial, since preventive methods can be deployed in the future based on tracking. As a good possible future line of research, using explainable AI could let us know if there is effectively any relationship between the IP addresses and the way an attack is done, especially in those related to a DoS.

3.4 Distributed Attack Detection Framework

The complexity and variety of Fog computing environments motivates to bring cyber-attack detection closer to the edge. Centralized attack detection has failed in IoT environments because of its low efficiency in the detection of attacks focused in the fog nodes. This motivates the adoption of distributed security architectures in cooperation with secure parameter exchange techniques. Besides, providing security to individual nodes is fundamental to achieve trustworthiness in the complete IoT network. Once trusted nodes are secured, IoT networks can be fully protected by taking advantage of the distributed attack detection system. This section describes a novel DL based distributed attack detection architecture where attack detection is implemented near the edge using distributed fog nodes to analyze network traffic and detect cyber-attacks. The proposed solution improves existing ones getting attack detection closer to the edge by means of local training acceleration and parameter optimization. Moreover, computational overhead and storage is offloaded, since global parameters are updated in the Cloud and the resulting update is propagated to the fog nodes. This design also reduces detection time.

3.4.1 Framework Description

The DL-based distributed attack detection architecture is designed to bring intrusion detection capabilities to Fog computing architectures in IoT environments. An IoT architecture consists of three layers, namely Edge, Fog and Cloud, as depicted in Fig. 4. The edge layer comprises millions of IoT devices with limited resources, like smartphones, sensors, security cameras, smart cars, etc. These devices generate vast volumes of unstructured data that is routed to the fog layer. The fog layer is the intermediate computing layer. It consists of devices that have significant computing power, large memories, and storage, as servers, routers or controllers. This middle layer improves the network performance, reduces latency and provides rapid

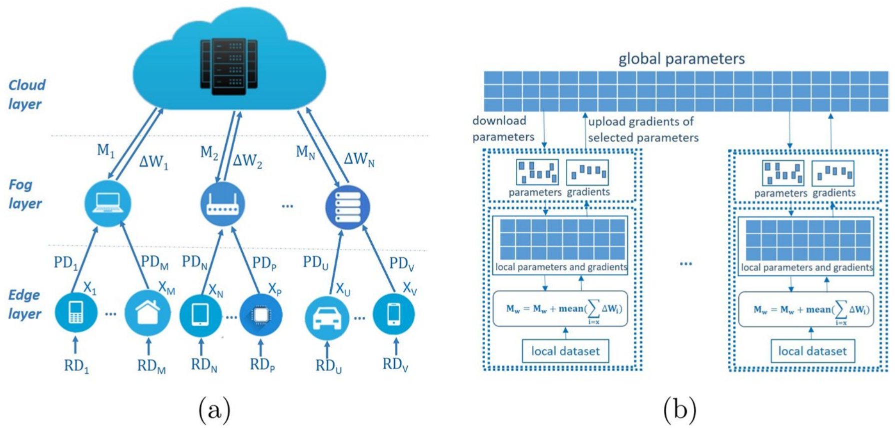

Fig. 4 DL based distributed attack detection architecture for Fog/IoT networks: **a** Distributed architecture; **b** Model update per layers

response to critical actions. Finally, the Cloud layer is the core layer which comprises high-performance servers and storage devices.

Likewise, the DL-based distributed attack detection architecture has three layers matching the IoT architecture. The Cloud layer contains the global model constituted by the latest values of the gradient weights, which are updated by the fog-nodes.The fog nodes host the attack detection system being responsible of training in a distributed manner the DL models. They feed the global parameters managed in the Cloud by the coordinating master. The master manages the parameter validation, optimization and exchange. The optimized parameters are then propagated to the fog nodes to train the distributed DL model. While, IoT devices provide their pre-processed raw data to fog nodes. Figure 4 shows the distributed attack detection architecture and Table 3 summarizes the notations and terminology used in the distributed attack detection algorithm.

Initially, the model weights are initialized to random values, as in most Deep Learning approaches. The fog nodes download the latest version of the global model from the Cloud server (M_w) and train it with the pre-processed raw data from the Iot devices. Each IoT device in the Edge layer pre-processes its raw data (RD_x) and

Table 3 Summary of notations and terminology–distributed detection algorithm

Reference	Description
RD_x	User raw data
PD_x	Preprocessed user raw data
M_w	Global model
x	User or IoT device
W_x	Local weights from an IoT device
ΔW_f	Gradient weight from a fog node

sends the pre-processed data (PD_x) to the corresponding node in the Fog layer. Then, as stated before, the latest version of the global model is downloaded from the Cloud to the fog node. Consequently, the fog node trains the downloaded model locally with the received dataset and sends the selected gradient weights to the server (ΔW_x). Finally, the global model is validated and updated in the Cloud layer (M_w). Overall, this process updates the DL model at each iteration by combining the data from all the fog nodes, which -in turn- gather the information from all the IoT devices. Algorithm 1 describes the steps in pre-processing, updates the model and gradient weights, and validates the model for the proposed distributed attack detection mechanism.

Algorithm 1 Distributed learning algorithm

```
Require: Global parameters $M_w \rightarrow$ Random values
 1: for all epochs do
 2:     for all nodes do
 3:         for all devices do
 4:             Pre-process user raw data $RD_x \rightarrow$PD$_x$
 5:             Send pre-processed user data to the corresponding fog-node
 6:             $M_w$: Download the updated model from the Cloud Server
 7:             Train the model ($M_w$) with the local dataset ($PD_x$)
 8:             and get the new model weights ($W_x$)
 9:          end
10:          Get the mean weights from all devices in the fog node:
11:          $\Delta W_f = \frac{\Sigma_{i=1}^{x}(W_i - W_{global})}{x}$
12:          Send the gradient weights ($\Delta W_f$) to the Cloud server
13:     end
14:     Update the global model with the following criteria:
15:      $M_w = M_w + mean(\Sigma_{i=x}^{x}(\Delta W_i))$
16: end =0
Validated model
```

The best two performing DL models, FFNN and LSTM, for the datasets, BoT-IoT and NSL-KDD, have been integrated into the distributed framework. Their performance is presented and analysed in next section.

3.4.2 Use Case Scenarios

When a device is connected to the cloud, it is exposed to many different threats. The proposed framework allows new devices to be protected against general attacks when connecting to the cloud and facing a new environment. Thus, every fog node in the cloud system is trained against attacks in their traffic networks in the same way as on a centralized environment.

On the other hand, the nature of a cloud system offers the possibility of changes in the environment while remaining connected to the cloud. A change in the cloud environment is produced by the change of the fog node responsible for the device. The proposed framework adapts centralized environments to protect distributed systems without decreasing the flexibility and freedom of its devices. Thus, every training in a fog node is shared among the others and a common model is available on the cloud. Thanks to the fog nodes contribution to the final model's weights, all the possible changes in the environment are covered for a device connected to the cloud.

As a first use case, we consider a general system in the cloud, using the proposed common attacks in clouds found in the NSL-KDD dataset. To also ensure a more present distributed system, we present a second use case, more specific and based on IoT environments. The attacks are considered for IoT devices, provided by the BoT-IoT dataset. In both cases, the dataset is considered as a centralized environment first, training FFNN and LSTM models and comparing them. Then, the partition of the dataset is done for as many fog nodes as the cloud is composed.

Considering the nature of the presented framework (explained in Sect. 3.4.1), the idea of using a centralized model for distributed environments cannot be contemplated. To bring intrusion detection capabilities to Fog computing architectures in IoT environments, the model must be trained in the fog nodes and be updated in the cloud layer. This makes inevitable the propagation of errors during a fog node training to its neighbors.

When training the fog nodes in the distributed framework, each of them has a partition of the dataset. Due to the similar network characteristics present in each of the nodes, during their training, the possible different models (in this case a FFNN and a LSTM) are facing the same challenge as in a centralized environment: the training of a model for attacks detection. If a model has less accuracy for the detection of attacks in a centralized environment, these differences will also propagate to its updated weights in the cloud layer and affect the posterior training of the fog nodes. Thus, the worse the performance of a fog node is, the worse its updated model in the cloud will be and therefore, the more impact this error will have for the rest of the training nodes, sharing the same distributed model in inference.

Overall, because the cloud layer updates the distributed model with similar trained environments coming from the fog nodes, the possibility of a better performance in this distributed framework deploying a less accurate model is very improbable and thus, discarded.

In addition, the computational cost for the deployment of the presented models also must be considered. Similar results are obtained with both models. It is then expected to have similar results in the distributed environment. Thus, the marginal gain that an LSTM model could offer us (in case the fog nodes where trained in different environments), is not comparable to the computational cost that its deployment represents. This is due to its complexity (with higher number of neurons and layers) as well as its training time (which will be always a FFNN even with the same complexity, due to the memory component of the LSTM models).

Therefore, a common model, extracted by the best performance in the centralized environment, is deployed in the cloud layer. Each fog node is then trained with a partition of the dataset and shares the characteristics of its network environment and attacks through the update of the cloud model weights. Finally, each device connected to the cloud is able to change as many times as desired the responsible fog node with total protection against either general attacks for the cloud environment as well as IoT attacks for the IoT distributed environment.

4 Experiments

This section presents the stage 4 of the distributed DL-based attack detection framework, which includes (i) the design of the neural networks parameters and (ii) the design of the distributed framework in terms of node scaling, data partitioning, and distributed algorithms. The design of the neural network parameters has been done using Hyperband and Learning Curve [27] procedures. Hyperband is essential for building the model. FFNN and LSTM models trained on datasets with a small number of features may end up with overfitting problems. Hyperband is used to avoid these problems and to obtain a solid model (by tuning the different parameters of the network). Hyperband must decide between different hyper-parameters to achieve a solid model. In this case, due to the limited number of features, Hyperband not only will determine the number of units per hidden layer, Hyperband also will consider the activation function and the dropout to achieve best accuracy. To this end, the model is tested with the most common and efficient activation layers: Sigmoid, ReLU and Tanh. Moreover, a maximum dropout threshold is set to 75% to also guarantee robustness in the final model.

4.1 Centralized Neural Networks

The design of the neural network depends on the DL model (FFNN or LSTM) considered, its purpose (detection) and the dataset (Bot-IoT or NSL-KDD) used. The use of a different model for each dataset is due to the differences in the attacks presented in both datasets.

As explained in Sect. 3.1, due to the similarity in the attacks (for instance DoS), most of the characteristics are shared between both datasets. The main difference is that NSL-KDD dataset exploits other network cyber-attacks that have different characteristics. Thus, its targets are broader than the attacks considered for the BoT-IoT dataset and therefore, the inputs to the detection model must be different.

Therefore, the FFNN designed for attack detection in IoT environments (Bot-IoT dataset) consists of 3 hidden layers with 100, 50 and 100 neurons respectively, all with ReLU activation functions and an output layer with the Sigmoid activation function. ReLU is used to avoid saturation between weights, hence having continuous weights value modelling during the training phase. On the other hand, binary cross-entropy is used for binary classification problems such as the target system. Thus, Sigmoid is the sole activation function compatible with binary cross-entropy since the loss requires logarithmic values between 0 and 1, which is not accomplished with any other activation function. Furthermore, the model is compiled with an Adam optimizer, a binary cross-entropy loss and 15 epochs. All of these metrics are designed empirically along with the tuning of the hyper-parameters with Hyperband.

For the NSL-KDD dataset, the model consists of four hidden layers, an input layer and an output layer. The hidden layers have 48, 128, 88 and 188 neurons, respectively. Furthermore, the activation function for all hidden layers is ReLU and Sigmoid for the output. After each hidden layer, a dropout is set by Hyperband (i.e. 25% for the first and third layer, 45% for the second, and 30% for the last one). The network is trained with an Adam optimizer, binary cross-entropy loss and 10 epochs.

The LSTM is the second model considered for the intrusion detection framework. As for the FFNN model, the LSTM is designed for detection using the BoT-IoT and NSL-KDD datasets. For attack detection in IoT networks (Bot-IoT dataset), the model consists of 3 hidden layers with 100, 64 and 32 neurons, respectively. All the hidden layers with ReLU activation functions and the output layer with Sigmoid activation function. The model is compiled with an Adam optimizer, binary cross-entropy loss and 30 epochs. As for the FFNN model, all the metrics are designed empirically jointly with Hyperband for tuning the hyper-parameters. Finally, the model is also defined to generalize the intrusion detection framework using the NSL-KDD dataset. It consists of five hidden layers, an input layer and an output layer. The hidden layers have 50, 100, 200, 100 and 50 neurons, respectively. The activation function for all hidden layers is ReLU and Sigmoid for the output layer. For compiling parameters, the network is trained with an Adam optimizer, binary cross-entropy loss and 20 epochs.

Table 4 summarizes the designed of the FFNN and LSTM DL models to detect and classify cyber-attacks in IoT networks.

4.2 Distributed Neural Networks Experiments Description

Once the network is tested in a centralized and theoretical environment, the best performing detection networks are tested in the distributed framework to simulate real environments. In our system, user information can be gathered and processed in a local environment, and then crucial training information is passed to the Cloud environment.

The distributed environment emanates from a subset of pre-processed data. The subsets are generated by dividing the selected dataset into equal parts for each end device the system has. Several experiments are performed. First, the distributed environment performs a non distributed test where the training resides in the Cloud. Then, the system is tested with 3, 5, 7 and 9 fog nodes. Each fog node trains with the information residing in 5 different devices, meaning that the number of devices used is 15, 25, 35

Table 4 FFNN and LSTM DL models to detect cyber-attacks in IoT networks

Purpose	DL model	Dataset	Hidden layers	Units	Dropouts (%)	Epochs
Detection	FFNN	BoT-IoT	3	100/50/100	–	15
		NSL-KDD	4	48/128/88/188	25/45/25/35	10
	LSTM	BoT-IoT	3	100/64/32	–	30
		NSL-KDD	5	50/100/200/100/50	–	20

and 45 for 3, 5, 7, and 9 fog nodes. Hence, the total training samples are equally divided into 15, 25, 35 and 45 parts, with non-repeated information.

The different number of fog nodes used allows us to ensure the proper operation of the system for different groups of devices since we target a system that maintains similar performance level in all different setups.

Overall, the centralized training will contain all the dataset information from all the users. In contrast, the distributed experiments will train individually in each fog node and only pass the essential information to the Cloud system. This essential information is the gradient of the weights of the model. With this information, the global model, residing in the Cloud environment, can be updated by doing the mean of all received results.

5 Results and Evaluation

All the DL models used in our experimental framework are implemented using Keras on TensorFlow. They are evaluated using different metrics that ensure and test their efficiency. These metrics are the Accuracy (ACC), Loss, Precision and Recall. They can be computed from a confusion matrix, i.e., a matrix representation of the classification results (see Table 5), where True Positive (TP) and True Negative (TN) denote the number of attack and normal records correctly classified. While False Positive (FP) and False Negative (FN) denote the number of normal and attack records incorrectly classified.

ACC (see Eq. 1) is the ratio of correctly classified predictions over the total number of instances evaluated:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Precision (see Eq. 2) is the ratio of items correctly classified from the total of items predicted:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Recall (see Eq. 3) represents the ratio of items correctly classified (attack or normal) as class C to all the items that were class C:

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

Table 5 Confusion matrix

		Predicted class Normal	Predicted class Attack
Actual	Normal	True negative (TN)	False positive (FP)
Class	Attack	False negative (FN)	True positive (TP)

The most common loss is the binary cross-entropy loss (see Eq. 4). Note that the y value represents the real output, whereas the $\hat{y}$ represents the output estimation:

$$Loss = -[y \cdot log(\hat{y}) + (1 - y) \cdot log(1 - \hat{y})] \tag{4}$$

All NNs inferences are repeated 20 times in the distributed environment to ensure their stability and consistency without overcharging the cloud server with redundant calculations.

5.1 Centralized DL Model Results

In order to provide the best DL model to the framework, first a centralized environment is created to train and test the best models for ID in IoT networks. Hence, a comparison between FFNN and LSTM models is done based on their accuracy, precision and recall. Furthermore, a generalisation of the framework is also achieved in this section, providing DL models the ability to detect intrusions in more general networks.

Table 6 summarizes the results of the FFNN and LSTM DL models to detect cyber-attacks in IoT networks.

The results show a different performance for the DL models depending on the dataset considered, achieving almost the same results for the BoT-IoT dataset, but showing significant variations for the NSL-KDD dataset. The difference between FFNNs and LSTMs in general cyber-attack detection are significant in terms of accuracy, where LSTM models drop more than 2% compared to the FFNNs. Precision and recall have a similar behaviour for these two different models. On the other side, there is no substantial difference in the performance of FFNN and LSTM models for IoT attacks (less than 1% of ACC is dropped when using LSTM models). However, these differences are notorious when the models require abstraction (dropping more than 1% in LSTM models). Thus, the FFNN model is the best choice in terms of abstraction but if only BoT-IoT is considered, both models are similar.

In conclusion, considering past inferences (i.e. LSTM models) has a negative impact on the final model performance when abstraction is required. In this case, the use of a forget gate forces the model to use redundant information. Thus, the inclusion of this redundant information reduces the abstraction of the model, dropping its inference capabilities and hence, having a worse performance than FFNNs. Still, LSTM models perform well in IoT intrusion detection with an accuracy above 99% in IoT attacks and 96% in general cyber-attacks.

Table 6 FFNN and LSTM results to detect cyber-attacks in IoT networks

Purpose	DL model	Dataset	ACC (%)	Precision (%)	Recall (%)
Detection	FFNN	BoT-IoT	99.97	99.95	99.99
		NSL-KDD	98.67	98.30	99.22
	LSTM	BoT-IoT	99.95	99.90	99.95
		NSL-KDD	96.44	95.74	97.66

Finally, taking into account the simplicity, the training method of the fog nodes (described in Sects. 3.4.1 and 3.4.2), and the performance shown by FFNNs, this DL model is the final choice for the intrusion detection framework and it is implemented in a distributed environment hereafter.

5.2 Distributed Framework Results

FFNNs are demonstrated, in Sect. 5.1, the best DL models for cyber-attack detection in IoT centralized environments.

Due to the nature of the distributed framework, only FFNN is integrated in the distributed framework and tested using the BoT-IoT and NSL-KDD datasets. Two different environments are set up with the corresponding DL model and data. Figures 5 and 6 show the accuracy and precision for the BoT-IoT dataset. Likewise, Figs. 7 and 8 show the results for the NSL-KDD dataset.

As the datasets are balanced, meaning that there is a very similar percentage of benign and malign samples, the accuracy, precision and recall are representative performance metrics. While, the loss function detects the possible over-fitting in the model.

The results show the performance of the network in the validation phase, which takes place in the Cloud environment after being individually trained in each fog node, as depicted in Sect. 3.4.1.

In the precision graph of the Fig. 5, the accuracy reaches almost 100% for both the centralized model and the distributed model with 3 nodes.The reason resides in the minimal data distribution, since the distributed framework can train with 1/3 of the samples. When using 9 nodes, the accuracy diminishes due to the smaller number of samples used for training. If the 9 node system could be trained with a similar

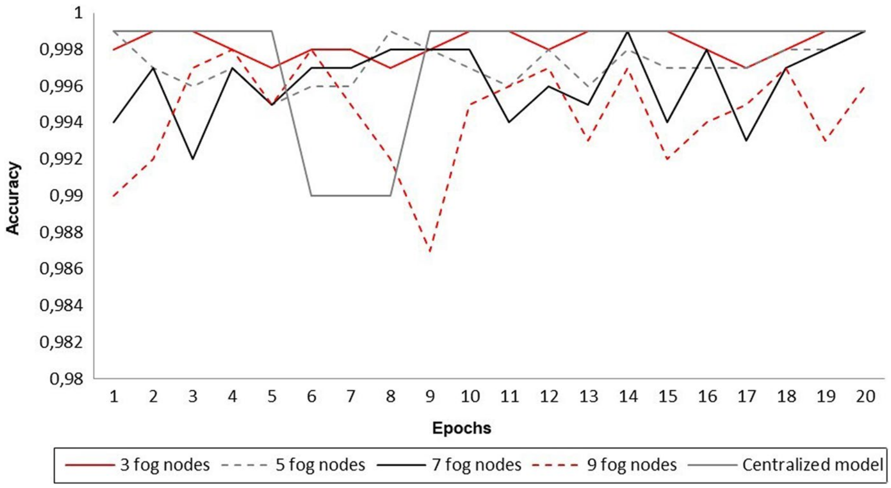

Fig. 5 BoT-IoT dataset FFNN ACC results in the distributed framework for 1, 3, 5, 7 and 9 fog nodes

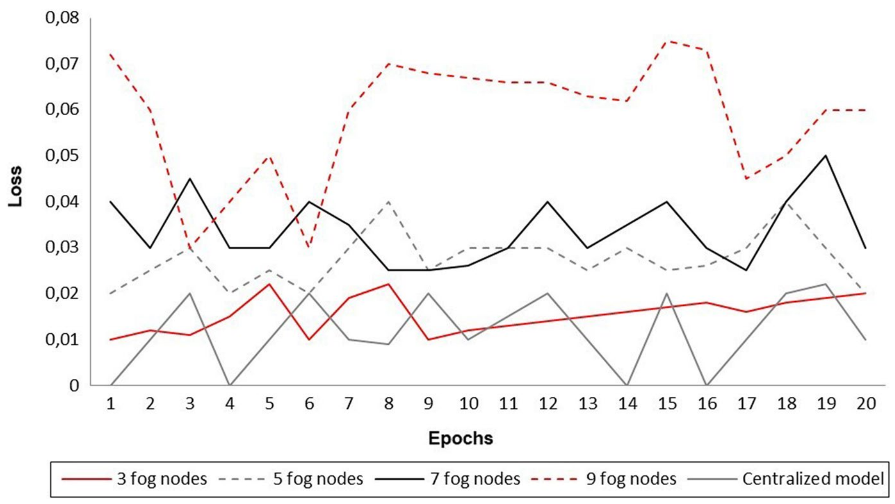

Fig. 6 BoT-IoT dataset FFNN Loss results in the distributed framework for 1, 3, 5, 7 and 9 fog nodes

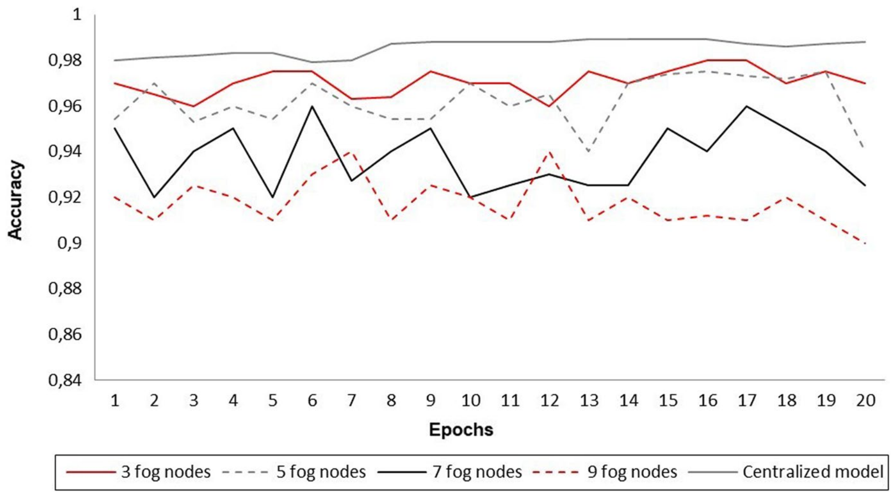

Fig. 7 NSL-KDD dataset FFNN ACC results in the distributed framework for 1, 3, 5, 7 and 9 fog nodes

number of samples to that of the 3 nodes system, the performance would be almost as in the centralized architecture.

Nevertheless, the network overall performance is minimally affected by the distributed framework. Performance loss is below 1% for all metrics and it maintains a loss significantly below 1. Hence, the final performance of the distributed system maintains its excellence by acquiring high results.

Figures 7 and 8 show the results obtained for the NSL-KDD dataset for the centralized and distributed (3, 5, 7 and 9 nodes) frameworks.

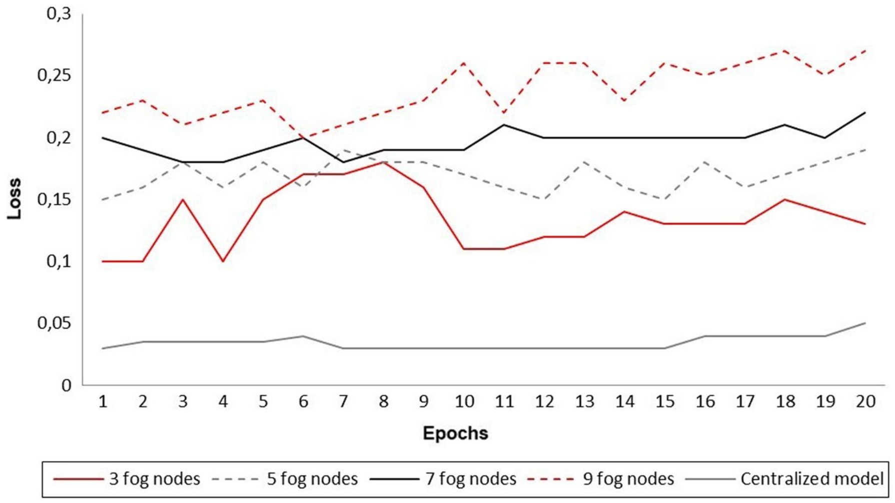

Fig. 8 NSL-KDD dataset FFNN Loss results in the distributed framework for 1, 3, 5, 7 and 9 fog nodes

The overall performance of the NSL-KDD dataset in the distributed framework decreases by approximately 2% compared with the BoT-IoT dataset. First, the system accuracy decreases between 1% and 6% depending on the number of fog nodes. The accuracy varies depending on the number of nodes due to the lack of variance in the samples contained in the smaller nodes. Furthermore, when the system has enough variance in the samples to train the network (less fog nodes are used in the environment), the accuracy only decreases 1%-2%. The reason of this drop resides in the nature of the dataset. Since there are not sufficient different samples, records are replicated. As seen in previous section, the huge amount of replicated records guides the model to overfiting problems, leaving no option but the reduction of the dataset to avoid them. Thus, there is a trade-off between the amount of samples and the ACC achived by the model, guiding to overfiting problems when too much replicated samples are coexisting in the dataset and dropping the accuracy in smaller nodes when reducing too much the number of records. This is why techniques like Learning curve are applied in the paper, offering the best number of samples to either minimise the drop of accuracy and the overfiting of the model.

As for loss, all results are close to zero, meaning that the system does not overfit. The performance remains almost the same for the recall metrics, but the precision has a similar curve as accuracy, decreasing almost 6%.

In conclusion, we achieve a smaller performance in the NSL-KDD network than in the BoT-IoT network. Our analysis shows that it is caused by the construction (i.e. number of samples) of each dataset. Nevertheless, both systems can be applied in a distributed framework proving that the networks work in a distributed environment maintaining good performance.

6 Conclusions and Future Work

New variants of cyber-attacks are continuously threatening nowadays IoT networks. The lack of robustness, as well as the limited computing capabilities of IoT devices require sophisticated defense mechanisms against zero-day attacks. Furthermore, the exponential increase of cyber-attack variants hinders its detection, forcing the defense mechanisms to a constant dynamism. In this line of research, this paper proposes a dynamic DL-based distributed framework for cyber-attack detection. The dynamism of the framework resides in its nature: a self-taught model applied in both centralized and distributed environments. These characteristics allow the protection of IoT devices themselves and the network where they belong to.

The proposed framework is trained in a distributed form, and the fog layer nodes act as a cyber-attack detection engine. Fog nodes analyze the data close to the edge improving detection and classification accuracy while minimizing latency. The framework departs from the comparison of two models, FFNN and LSTM, for two datasets BoT-IoT and NSL-KDD. FFNN achieves better detection rates for both datasets, specially for the NSL-KDD dataset. FFNN is then evaluated in a distributed architecture. The detection system is implemented in the fog nodes, close to the edge to achieve better detection rates and minimize latency. The experiments have demonstrated the success of distributed DL models to be adopted by IoT networks to detect different attacks with high detection and accuracy rates. The proposed framework improves cyber-attack detection in IoT neworks with limited capacity end-devices, detect several attacks with high accuracy, specially for the BoT-IoT dataset. In the future, we will improve the performance of the distributed framework for the NSL-KDD dataset considering data augmentation techniques to have enough data when the number of fog nodes of the framework increases. In this way, we expect to obtain accuracy results similar to the achieved for the BoT-IoT dataset, where accuracy, precision and recall remains similar to the centralized architecture.

Other techniques can be considered for the research of a causal relationship between the IP addresses and the way an attack is done, especially in those related to DoS attacks.

As presented in Sect. 3.4.2, similar trained environments are used for the fog nodes training. However, there is the possibility to train each fog node on different environments without sharing similar characteristics. In this scenario, a lower performance of a centralized model does not imply a worse performance in a distributed environment. A possible training with different datasets for each of the fog nodes, can lead to a more general model, avoiding the creation of a model per dataset.

Finally, in the same line of research, both datasets, NSL-KDD and Bot-IoT, share similar characteristics that can lead us to the creation of a common general model for attacks detection in distributed environments. Thus, the distributed framework presented in this paper can be contemplated in future projects, to be expanded in different fields by using different transfer learning methods.

Author Contributions OJ contributed on manuscript writing, creation of DL models, software development and performing the experiment work.BO contributed on research goals conceptualization, methodology and project design, experiments validation, investigation process, management and coordination of the research activities. ER contributed on manuscript writing, investigation and project administration. NG contributed on algorithm design and code implementations. HA contributed on software development. RC contributed on writing-original draft and acquisition of the project financial support. All authors read and approved the final manuscript.

Funding declaration Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. This work is partially supported by the Spanish Ministry of Science and Innovation under contract PID2021-124463OB-IOO, by the Generalitat de Catalunya under grants 2017SGR962, 2021SGR00326, and by the DRAC (IU16-011591), the HORIZON Vitamin-V (101093062) and the HORIZON-AG PHOENI2X (101070586) projects.

Declarations

Conflicts of Interest The authors declare no conflict of interest.

References

1. Lab, W.T.: Internet security report: WatchGuard's threat lab analyzes the latest malware and internet attacks. Technical report (Q3 2020). https://www.watchguard.com/wgrd-resource-center/security-report-q3-2020. Accessed 17 Mar 2021
2. Cherdantseva, Y., Hilton, J.: A reference model of information assurance security. In: 2013 International Conference on Availability, Reliability and Security, pp. 546–555 (2013). https://doi.org/10.1109/ARES.2013.72
3. Kwon, D., Kim, H., Kim, J., Suh, S., Kim, I., Kim, J.: A survey of deep learning-based network anomaly detection. Clust. Comput. **22**(5), 949–961 (2019)
4. Anderson, J., Carbonell, J., Mitchell, T., Michalski, R., Amarel, S., Tecuci, T., Kodratoff, Y.: Machine Learning: An Artificial Intelligence Approach. M. Kaufmann, Los Altos, CA (1983)
5. Kilincer, I., Ertam, F., Sengur, A.: Machine learning methods for cyber security intrusion detection: datasets and comparative study. Comput. Netw. **188**, 107840 (2021). https://doi.org/10.1016/j.comnet.2021.107840
6. Fadlullah, Z.M., Tang, F., Mao, B., Kato, N., Akashi, O., Inoue, T., Mizutani, K.: State-of-the-art deep learning: evolving machine intelligence toward tomorrow's intelligent network traffic control systems. IEEE Commun. Surv. Tutor. **19**(4), 2432–2455 (2017). https://doi.org/10.1109/COMST.2017.2707140
7. Tsimenidis, S., Lagkas, T., Rantos, K.: Deep learning in IoT intrusion detection. J. Netw. Syst. Manag. (2022). https://doi.org/10.1007/s10922-021-09621-9
8. Diro, A.A., Chilamkurti, N.: Distributed attack detection scheme using deep learning approach for Internet of Things. Futur. Gener. Comput. Syst. **82**, 761–768 (2018). https://doi.org/10.1016/j.future.2017.08.043
9. Roopak, M., Yun Tian, G., Chambers, J.: Deep learning models for cyber security in IoT networks. In: 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), pp. 452–457 (2019). https://doi.org/10.1109/CCWC.2019.8666588

10. Shalaka, M., Pawar, P.M., Muthalagu, R.: Efficient Intelligent Intrusion Detection System for Heterogeneous Internet of Things (HetIoT). J. Netw. Syst. Manag. (2023). https://doi.org/10.1007/s10922-022-09697-x
11. Alom, M.Z., Bontupalli, V., Taha, T.M.: Intrusion detection using deep belief networks. In: 2015 National Aerospace and Electronics Conference (NAECON), pp. 339–344 (2015). https://doi.org/10.1109/NAECON.2015.7443094
12. Kim, J., Kim, J., Thi Thu, H.L., Kim, H.: Long Short Term Memory Recurrent Neural Network classifier for intrusion detection. In: 2016 International Conference on Platform Technology and Service (PlatCon), pp. 1–5 (2016). https://doi.org/10.1109/PlatCon.2016.7456805
13. Shone, N., Ngoc, T.N., Phai, V.D., Shi, Q.: A deep learning approach to network intrusion detection. IEEE Trans. Emerg. Topics Comput. Intell. **2**(1), 41–50 (2018). https://doi.org/10.1109/TETCI.2017.2772792
14. Ferrag, M.A., Maglaras, L., Moschoyiannis, S., Janicke, H.: Deep learning for cyber security intrusion detection: approaches, datasets, and comparative study. J. Inform. Secur. Appl. **50**, 102419 (2020). https://doi.org/10.1016/j.jisa.2019.102419
15. Cibersecurity, C.I.: Dataset IDS 2018: CSE-CIC-IDS2018 on AWS. https://www.unb.ca/cic/datasets/ids-2018.html Accessed 17 Mar 2021
16. Koroniotis, N., Moustafa, N., Sitnikova, E., Turnbull, B.: Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset. Futur. Gener. Comput. Syst. **100**, 779–796 (2019). https://doi.org/10.1016/j.future.2019.05.041
17. Yadav, S., Subramanian, S.: Detection of application layer DDoS attack by feature learning using Stacked AutoEncoder. In: 2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT), pp. 361–366 (2016). https://doi.org/10.1109/ICCTICT.2016.7514608
18. Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A., Lloret, J.: Conditional variational Autoencoder for prediction and feature recovery applied to intrusion detection in IoT. Sensors (Basel) **17**(1967), 1–17 (2017). https://doi.org/10.3390/s17091967
19. Luo, T., Nagarajan, S.G.: Distributed anomaly detection using autoencoder neural networks in WSN for IoT. In: 2018 IEEE International Conference on Communications (ICC), pp. 1–6 (2018). https://doi.org/10.1109/ICC.2018.8422402
20. Sharafaldin., I., Habibi Lashkari., A., Ghorbani., A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP), vol. 1, pp. 108–116 (2018)
21. Vijayanand, R., Devaraj, D., Kannapiran, B.: Intrusion detection system for wireless mesh network using multiple support vector machine classifiers with genetic-algorithm-based feature selection. Comput. Secur. **77**, 304–314 (2018). https://doi.org/10.1016/j.cose.2018.04.010
22. University of California, I.: KDD Cup'99. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. Accessed 17 Mar 2021
23. Cibersecurity, C.I.: NSL-KDD. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html Accessed 17 Mar 2021
24. Ramchoun, H., Idrissi, M.J., Ghanou, Y., Ettaouil, M.: Multilayer perceptron: architecture optimization and training. Int. J. Interact. Multimed. Artif. Intell. **4**(1), 26–30 (2016). https://doi.org/10.1145/3090354.3090427
25. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735
26. Li, L., Jamieson, K.G., DeSalvo, G., Rostamizadeh, A., Talwalkar, A.: Efficient hyperparameter optimization and infinitely many armed bandits. CoRR (2016) . https://doi.org/10.48550/arXiv.1603.06560
27. Borgianini, F.: Using the learning curve to design effective training. PM Netw. **12**(7), 50–52 (1998)