Modern Applied Science; Vol. 16, No. 2; 2022
ISSN 1913-1844 E-ISSN 1913-1852
Published by Canadian Center of Science and Education

DDoS Attacks Detection in the IoT Using Deep Gaussian-Bernoulli Restricted Boltzmann Machine

Gafarou O. Coli[1], Segun Aina[1], Samuel D. Okegbile[1], Aderonke R. Lawal[1] & Adeniran I. Oluwaranti[1]

[1] Department of Computer Science and Engineering, Obafemi Awolowo University, Ile-Ife, Nigeria

Correspondence: Segun Aina, Department of Computer Science and Engineering, Obafemi Awolowo University, Ile-Ife, Nigeria.

Received: February 5, 2022 Accepted: March 16, 2022 Online Published: April 2, 2022

doi:10.5539/mas.v16n2p12 URL: https://doi.org/10.5539/mas.v16n2p12

Abstract

Distributed denial of service (DDoS) attack is generally known as one of the most significant threats to the internet of things (IoT). Current detection technologies of DDoS attacks are not adequate for IoT systems because of the peculiar features of IoT such as resource constraint nodes, specific network architecture, and specific network protocols. Providing adequate DDoS attacks detection systems to IoT, however, becomes a necessity since IoT is ubiquitous. This study hence developed a deep learning-based model for detecting DDoS in IoT, while considering its peculiarities. The proposed deep learning-based model was formulated using a deep Gaussian-Bernoulli restricted Boltzmann machine (DBM) because of its capability to learn high-level features from input following the unsupervised approach and its ability to manage real-time data that is common in the IoT network. Furthermore, the SoftMax regression was used for classification. The accuracy of the proposed model on the network socket layer-knowledge discovery in databases was obtained as 93.52%. The outcome of the study shows that the proposed DBM can efficiently detect DDoS attacks in IoT.

Keywords: Boltzmann machine, deep learning, DDoS, IoT, nsl-kdd

1. Introduction

The Internet of Things (IoT), is generally a self-configuring network of small sensor nodes in which nodes communicate with one another to sense, supervise, comprehend the physical world and provide services (Aris et al., 2015). Because of its ubiquitous and pervasive characteristics, its applications have appeared in a variety of domains, including health care, fitness, home energy management, classroom automation, smart cities, and many more (Alqahtani et al., 2020). IoT is now receiving wide adoption in many sectors that require things to interchange via the internet to process chores smartly with little or no human involvement. Unfortunately, security in IoT has gained considerable concern in recent times. The distributed denial of service (DDoS) attack represents the most important threat to IoT (Arnaboldi & Morisset, 2017; Robert & Wang, 2020). This denial of service (DoS) is a class of cyberattacks whose aim is to destruct or deny the use of services, such as environmental monitoring or services accessible remotely. An attack might come from one source - DoS or multiple sources - referred to as DDoS (Robert and Wang, 2020). In a study conducted by Kaspersky Labs in 2017, the consequences of DDoS on organizations have increased considerably and is, on average between $120 thousand and $2 million, which implies the incapability to run businesses, insurance premium increases, loss of contracts, and opportunities. In addition, the appraisal reveals that 70% of IoT devices are very easy to attack (Denise, 2020).

Despite the existence of various DDoS detection techniques based on traditional methods, current solutions are limited for IoT systems, because of IoT's peculiar features that affect detection system development. First, finding nodes in IoT networks that can support detection system agents is difficult. In fact, nodes are generally resourced constraints in IoT networks. The second peculiar feature is the architecture. Communications in IoT networks often follow the multi-hop method and nodes can simultaneously forward packets while working as end devices. The last feature is related to IoT's network protocols. Network protocols used in IoT are not common (for instance, IEEE 802.15.4, 6LoWPAN, RPL, CoAP, etc.). New protocols bring new vulnerabilities and require novel detection systems. Encryption and authentication are insufficient to protect IoT systems (Kasinathan et al., 2013). By inheriting Internet protocol (IP) technologies and low-power wireless links, IoT systems inherit the security vulnerabilities and issues of these technologies (Sarigiannidis et al., 2015). The heterogeneity and the dispersed nature of the nodes render traditional intrusion detection systems (IDS) difficult to implement (Fu et al., 2011;

Chen et al., 2009). Existing detection systems are not conceived for these problems and there are no universal IDS for DDoS in IoT (Kasinathan et al., 2013; Raza et al., 2013).

Meanwhile, the deep learning (DL) method has been earlier used to provide solutions to many interesting problems in the big data field and offers a unique opportunity for DDoS security mechanisms in IoT. Although the use of the DL method is primarily restrained to big data, the latest findings on network traffic classification and intrusion detection systems in (Diro & Chilamkurti, 2018; Elsaeidy et al., 2019; Javaid et al., 2016; Kang & Kang, 2016; Li et al., 2015) show that DL can have new applications in identifying attacks in IoT. The main benefit of DL is its ability to extract high-level features, the absence of manual feature engineering, unsupervised pre-training, and compression capabilities. These benefits are now encouraging the use of DL in resource constraint networks such as IoT (Vincent et al., 2010). In this paper, taking the advantage of the DL, we develop a DDoS attack detection model in IoT using a deep Gaussian-Bernoulli restricted Boltzmann machine (DBM). To do so, in section 2, we present relevant literature reviews regarding intrusion detection and IoT. The contribution to the knowledge of this study - an established blueprint and a deep learning based-model for detecting distributed denial of service attacks in IoT considering the peculiarities of IoT systems - is exposed in section 3. Section 4 discusses the system implementation, data description, and the experimental environment. Section 5 detailed the results of this work, while the last section presents some concluding remarks.

2. Related Works

The DL approach was employed in (Li et al., 2015) to detect malicious code in information technology systems, where an auto-encoder was used for features extraction and data dimensionality reduction by converting complicated high-dimensional data into low dimensional codes with nonlinear mapping. A deep belief network (DBN) was used for classification while considering a centralized anomaly-based detection approach. The model was simulated using the knowledge discovery in databases cup 99 (KDD-Cup99) dataset and an accuracy of 92.10% was obtained. Illustrative results showed that the proposed model performed better than a single DBN. The work supports that deep networks are better at identifying cyber-attack than simple machine learning algorithms.

Another study based on the DL approach was conducted in (Javaid et al., 2016). The authors used sparse auto-encoders along with a backpropagation (BP) algorithm for feature extraction following an unsupervised approach. They simulated the proposed model using the network socket layer-knowledge discovery in databases (NSL-KDD) dataset. The learned features were classified using a softmax regression function. The study employed a cross-validation technique based on n-fold for performance evaluation, and the accuracy of the developed model was 88.39%. A deep neural network (DNN) based IDS was also adopted in (Kang & Kang, 2016) to improve the security of an in-vehicular network. During the simulation phase, sensors placed in the controlled area network (CAN) bus were used to analyze the traffic entering and leaving the vehicle. They used a DBNs with the conventional stochastic gradient descent algorithm for feature extraction on the in-vehicular network packets. Then, DNN was used as a classifier to provide the accuracy of each class to distinguish normal packets from hacked ones. Experimental results showed 98% in terms of accuracy and showed that the developed method can give a quick response to any attack on the vehicle.

In Ma et al. (2016), spectral clustering deep neural network (SCDNN) - a combination of spectral clustering (SC) and DNN algorithms was proposed. The dataset was first split using cluster centers, as in SC; and similarity amongst features was used to measure the distance between data points both in testing and training set. The KDD-Cup99, NSL-KDD, and sensor network dataset (SND) were used to test the model. The outcomes showed that the SCDNN classifier outperformed back propagation neural network (BPNN), support vector machine (SVM), random forest (RF), and Bayes tree models in terms of detection accuracy. Haddadpajouh et al. (2020) proposed a multi-kernel support vector machine (SVM) for IoT cloud-edge gateway malware hunting, using the grey wolf optimization (GWO) algorithm. In Yadav & Subramanian (2016), a stack of auto-encoders was used to study features in the application-layer DDoS (AL-DDoS) attack dataset - an artificial DDoS dataset. To classify the input into normal and intrusion traffic, logistic regression was used. The average detection rate and false-positive rates were 98.99% and 1.27% respectively. These results were used to benchmark the proposed method with the existing ones, such as hidden Markov models (HMM), hierarchical clustering, and random walk graph. They concluded that the proposed technique outperformed the existing ones.

Pang et al., 2021 reviewed the various models that have been employed for anomaly detection while (Li et al., 2020) surveyed different random forest models used in IDS with a variety of datasets using different features and classes. In Diro & Chilamkurti (2018), a distributed anomaly-based IDS was developed to detect DDoS attacks in IoT systems using a DL model for feature extraction and softmax regression for classification. The developed model was also simulated using the NSL-KDD dataset. The model achieved 99% in terms of accuracy, while the

distributed detection approach was shown to outperform the centralized one. In addition, the results showed that deep network-based models were more effective against DDoS attacks than shallow neural networks. However, the deep learning model used for feature extraction in the work was not specified. Latif et al. (2020) also proposed an approach using convolutional neural networks (CNN) to classify botnet attacks. They further split the dataset into four (4) separate parts, the same model was trained and tested on the data separately for binary classification. The research in Imamverdiyev & Abdullayeva (2018) proposed a Gaussian-Bernoulli type restricted Boltzmann machine (RBM) for detecting DDoS attacks. They used seven additional layers between the visible and the hidden layers of the RBM for feature extraction and features classification. They simulated the model using the NSL-KDD dataset with Matlab and the detection accuracy was 78%. The results analysis of the developed model showed it performed better compared to the other type of RBM.

Latif et al., 2020 in their work developed a deep random neural network (DRaNN) and they trained the model using the UNSW-NB15 dataset. Their model could achieve an accuracy of 99.54%. In Elsaeidy et al. (2019), DDoS attack detection for the smart city using an RBMs based smart city intrusion detection system for feature extraction was proposed. Four classifier including feed-forward neural network (FFNN), automated FFNN, RF, and SVM, were used. Classifiers were selected and trained. The accuracy of the developed method was tested and benchmarked using a dataset from a smart water distribution plant. The best accuracy (98%) was achieved with SVM as a classifier. The outcomes showed the performance of the developed method in detecting DDoS attacks.

3. System Model

The developed model for detecting DDoS attacks in IoT systems is made of the softmax regression for attack detection and deep Gaussian-Bernoulli restricted Boltzmann machine (DBM) for feature learning. Fog nodes were used to host the developed attack detection system to unload storage and computation from IoT nodes. Parallelly, each fog node hosts data training using the DBM. Holdout set validation method was used to build the final model with the training dataset for training (70%), validation (10%), and testing (20%).

3.1 Deep Gaussian-Bernoulli Restricted Boltzmann Machine

DBM is a stack of RBMs. The first RBM in the stack is a Gaussian-Bernoulli RBM, a type of RBM that handles real data and is suitable to the data generated by IoT systems (Elsaeidy et al., 2019). DBM is a generative model with symmetrically coupled stochastic units and no-intralayer connections between the units. Connections in the deep RBM exist only between units of neighboring layers. The deep RBM learns complex internal representations and is effective in resolving IoT network traffic analysis problems. Complex representations can be constructed from a large amount of unlabelled IoT network traffic and some labeled sensory data can then be employed to fine-tune the model. Typically, a DBM is made of a set of visible units $\boldsymbol{v} \in \{0,1\}^D$ and a set of hidden units $h \in \{0,1\}^P$. Consider a DBM with two hidden layers $\{h^1, h^2\}$ and one visible layer v, the conditional distributions over the visible and the two sets of hidden units are given by (Salakhutdinov & Hinton, 2019)

$$p\left(h_j^1 = 1 | \boldsymbol{v}, h^2, \theta\right) = sigm\left(\sum_i \frac{1}{\sigma_i^2} v_i w_{ij} + \sum_k h_k^2 w_{jk}^1 + b_j^1\right), \quad (1)$$

$$p(h_k^2 = 1 | h^1, \theta) = sigm\left(\sum_j h_j^1 w_{jk}^1 + b_k^2\right), \quad (2)$$

$$p(v_i = 1 | h^1, \theta) = N\left(\sum_j h_j^1 w_{ij} + b_i, \sigma_i^2\right) \quad (3)$$

where $\theta = \{w_{ij}, w_{jk}^1, b_i, b_j^1, b_k^2\}$ are the model parameters; *sigma (x)* is the sigmoid function ($\frac{1}{1+e^{-x}}$); w_{ij} are the weights between v and h^1; w_{jk}^1 the weights between h^1 and h^2; b is the biases of v, b^1 is the biases of h^1, b^2 is the biases of h^2 and σ_i is the standard–deviations of v. $N(\mu, \sigma_i^2)$ is a probability density of Normal distribution with a mean μ a standard deviation σ_i.

The learning of the DBM model was done using (4). The first term on the right side is called the data-dependent expectation and the second one is called the expectation of the model.

$$\frac{\partial \log p(v, \theta)}{\partial \theta} = Ep(h^1, h^2 | v)\left[\frac{\partial E(v, h^1, h^2 | \theta)}{\partial \theta}\right] - Ep(v, h^1, h^2)\left[\frac{\partial E(v, h^1, h^2 | \theta)}{\partial \theta}\right] \quad (4)$$

The data-dependent expectation and the model's expectation can only be approximated. Because of that, persistent contrastive divergence (PCD) through mean-field approximation (Salakhutdinov & Hinton, 2019), (Cho et al., 2013) was used to estimate the data-dependent expectation and Gibbs sampling through Markov chain Monte Carlo method (Salakhutdinov & Hinton, 2019; Cho et al., 2013) was used to approximate the model's expectation.

i. *Gibbs sampling*: States of the neurons in the hidden layers are calculated using probability functions $p(h_j^1 = 1|\, v, h^2, \theta)$ in (1) and $p(h_k^2 = 1|\, h^1, \theta)$ in (2). Once these values are available, the other function $p(v_i = 1|\, h^1, \theta)$ presented as (3) is used to predict new input values for the visible layer. This process can be repeated *m–times*. In the end, vectors ($\tilde{v}^{m}$, $\tilde{h}^{1,m}$, $\tilde{h}^{2,m}$) which were recreated from original input values ($v_0, h^{2,0}$) were obtained.

ii. *PCD*: The state of each hidden neuron h_j^1 and h_k^2 is calculated using its probability of being active μ_j^1 and μ_k^2 respectively. PCD shown in (7) - (9) was then used along with the vectors ($\tilde{v}^{m}$, $\tilde{h}^{1,m}$, $\tilde{h}^{2,m}$) obtained at the Gibbs sampling stage to update parameters.

$$\mu_j^1 = sigm(\sum_i \frac{1}{\sigma_i^2} v_i w_{ij} + \sum_k h_k^2 w_{jk}^1 + b_j^1 \tag{5}$$

$$\mu_k^2 = sigm(\sum_j h_j^1 w_{jk}^1 + b_k^2) \tag{6}$$

$$\Delta w_{ij} = \alpha(\frac{1}{N}\sum_{n=1}^{N} v^n (\mu^{1,n})^T - \frac{1}{M}\sum_{m=1}^{M} \tilde{v}^{1,m} (\tilde{h}^{1,m})^T) \quad (9) \tag{7}$$

$$\Delta w_{jk}^1 = \alpha(\frac{1}{N}\sum_{n=1}^{N} \mu^{1,n} (\mu^{2,n})^T - \frac{1}{M}\sum_{m=1}^{M} \tilde{h}^{1,m} (\tilde{h}^{2,m})^T) \tag{8}$$

$$\theta_{new} = \theta_{old} + \Delta\theta \tag{9}$$

To initialize the parameters of the DBM, a stack of two RBMs were trained on the training data and its parameters were used to initialize the parameters of the three-layer DBM during the training process (Salakhutdinov & Hinton, 2019; Cho et al., 2013). The overall learning procedure of the DBM is shown in Algorithm 1.

Algorithm 1. The training procedure of the DBM

1) **Input:** Training set $\{v\}_{n=1}^N$, *n*umber of Markov particles *M,* Iterations *m*
2) **Output**: A trained DBM model with parameters $\{w_{ij}, w_{jk}^1, b_i, b_j^1, b_k^2\}$.
3) Use $\{v\}_{n=1}^N$ to pre-train the DBM, and get the initial parameters of DBM $\{w_{ij}, w_{jk}^1, b_i, b_j^1, b_k^2\}$.
4) **for** *ite*=0 to *m* **do**
5) For each training sample v^n, use mean field approach (Equations 5 and 6) to get the variational parameters μ^n.
6) For each Gibbs sampling step, use equations 1, 2, and 3 repeatedly to obtain the state ($\tilde{v}^{m}$, $\tilde{h}^{1,m}$, $\tilde{h}^{2,m}$)
7) Update the parameters of DBM with PCD using equations 7, 8 and 9
8) **end**

3.2 Softmax Regression

The softmax regression is mathematically defined in (10) and was used as a classifier.

$$\phi_{\text{softmax}}(z^{(i)}) = p(y = j|z^{(i)}) = \frac{e^{z^{(i)}}}{\sum_{j=0}^{k} e^{z_k^{(i)}}}, \tag{10}$$

given that $z = w_1x_1 + ... + w_mx_m + b = \sum_{l=1}^{m} w_l x_l + b = w^T x + b$, where *w* the weight vector. The feature vector of one training sample is represented as *x* while *b* is known as the bias unit. The SoftMax function computes the probability that the training sample $x^{(i)}$ belongs to class *j* given the weight and net input $z^{(i)}$. The probability $p(y = j|x^{(i)};\, w_j)$ for each class label in *j=1,..., k* was then computed.

4. Model Implementation

Next, we provide the details of the system implementation, data description, and presented the experimental environment.

4.1 System Implementation

We deployed sniffers in the fog nodes and gateways' network. These monitor and gather the information exchanged amongst nodes. Since an important quantity of data can be extracted from packets, directly processing

it might decrease the performance of the DDoS attack detection system and can be resource-consuming. Therefore, the collected network traffic was used to feed a data pre-processing module, which included feature transformation and normalization. These operations entailed, amongst others, the inspection and selection of important features from the large-scale data, and the conversion of data into numeric features. The output of the pre-processing was used to train and develop the DBM for DDoS attack detection in the IoT environment. Finally, the SoftMax regression was used to classify traffics as "normal" or "attack".

4.2 Data Description

The NSL-KDD (Tavallaee et al., 2009) dataset which is a fundamental dataset for the evaluation of DDoS IDS was used to benchmark the developed model. The dataset involved normal traffics and attack traffic, including DoS, probing, user-to-root (U2R), and root-to-local (R2L) as described in Table 1.

Table 1. NSL-KDD distribution

Traffic		**Training**	**Test**
Normal		67343	9711
	DoS	45927	7458
	U2R	52	67
Attack	R2L	995	2887
	Probe	11656	2421
Total		**125973**	**22544**

The NSL-KDD dataset has 41 features (03 are nominal, 04 are binary, and the remaining 34 are continuous). We labeled the features as normal or a specific type of attack. The training data contained 23 traffic classes (22 attacks and 01 normal) and the test data contained 38 traffic classes (21 attacks from the training data, 16 novel attacks, and 01 normal). We also divided the attacks into four categories (DoS, Probing, U2R, and R2L) according to their purpose.

4.3 Data Pre-Processing

Before training the model, features that an algorithm cannot immediately process were encoded into discrete features using a one-hot-encoding (1-of-N) technique. The one-hot-encoding is known as the method of splitting the column containing numerical categorical data into many columns, depending on the number of categories present in that column. Each column contains "0" or "1" which is corresponding to the column it has been placed. This is accomplished using the OneHotEncoder Class of the Python library for data pre-processing Scikit-learn. Columns that were categorical and converted to binary are protocol_type (3 categories), service (70 categories), flag (11 categories). Because of encoding, 122 input features were obtained. The obtained dataset was scaled and centered using the StandardScaler Class of Scikit-learn library to avoid features with large values that may weigh too much in the results.

4.4 Experimental Environment

We trained the multilayer RBM on a normalized dataset. During the training, labels of the dataset were not used. For the simulation, 2-classes (normal vs attack) and 5-classes (normal, DoS (Denial of Service attacks), Probe, R2L (Root to Local attacks), and U2R (User to Root attack) were considered. After hyper-parameter optimizations, the model used 122 input features, 200 first layer neurons, 100-second layer neurons, and the last SoftMax layer with neurons equal to the number of classes. To train the network, 3 epochs were used.

We carried the experimental aspect of this study using Ubuntu x86 18.04.3 LTS installed on an HP-Pavillon-Dv6 with 6 GB memory, processor Intel® Core™ i5-2450M CPU @ 2.50GHz x 4. The PC had a graphics card Intel® Sandybridge Mobile with 4GB available. The model was simulated using the Integrated Development Environment (IDE) Spyder 3.3.3 under Anaconda 4.7.10 version 2019.03 installed with Python 3.7.3 version, SciPy library 1.2.1, Keras 2.2.4, and TensorFlow 1.14.0 backend. Scikit-learn 0.20.3, Pandas 0.24.2, NumPy 1.16.2, and Matplotlib 3.0.3 libraries were all installed to provide a robust simulation environment.

4.5 Evaluation Method

Evaluation of the performance of the developed model was carried out using accuracy, precision (P), recall (R), F-measure, and Throughput. The mentioned performance criteria were computed as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

$$Precision = \frac{TP}{TP + FP} \tag{12}$$

$$Recall = \frac{TP}{TP + FN} \tag{13}$$

$$F - measure = \frac{2.P.R}{P+R} \tag{14}$$

Where TP represents true positive and is defined as the case where the network attack types are correctly distinguished, TN represents the true negative, defined as cases where normal network data were correctly classified as normal. Similarly, FN represents a false negative defined as the case where an attack was classified as normal dataflow, while FP represents a false positive defined as cases where normal cases were classified as attacks.

Accuracy hence shows the overall correct detection accuracy of the dataset, while precision denotes the ratio of correctly detected attacks to total detected attacks. Recall signifies the degree of attacks that were correctly detected among all cases classified as attacks. Finally, the weighted average of Precision and Recall was captured by the F-measure. A good performance is indicated by higher accuracy and recall.

5. Results

In this subsection, we present the result of the proposed model. We first present the performance of the system using the selected evaluation methods which were followed by the comparative analysis.

5.1 Performance Evaluation

Table 3 shows the effectiveness of the DBM based on various metrics. Thus, the accuracy of the DBM in classifying traffic records as an attack or normal was 93.52%. Precision, Recall, and F-measure in the same context were 93%. For the classification of network traffics as "Normal" and four different attack categories (probe, U2R, R2L, DoS), the accuracy of the DBM was 91.69%.

Table 2. Evaluating the effectiveness of the methods

Class	Accuracy	Precision	Recall	F-measure
2-classes (Normal and attack)	93.52%	93%	93%	93%
5-classes (Normal and 4 attacks)	91.69%	92%	92%	91%

In the same scenario, the precision and the recall were 92% and the F-measure was 91%. It was concluded that the DBM efficiently detected DDoS attacks in the IoT environment. Figure 1 and Figure 2 visually show the confusion matrices of the two different classification with details over the reparation of the TP, TN, FP, and FN in each situation. The occurrence of DoS, U2R, R2L, probe, normal classes, and the ability of the developed DBM to detect these points respectively for each type of classification is also pictured in Figure 1 and Figure 2.

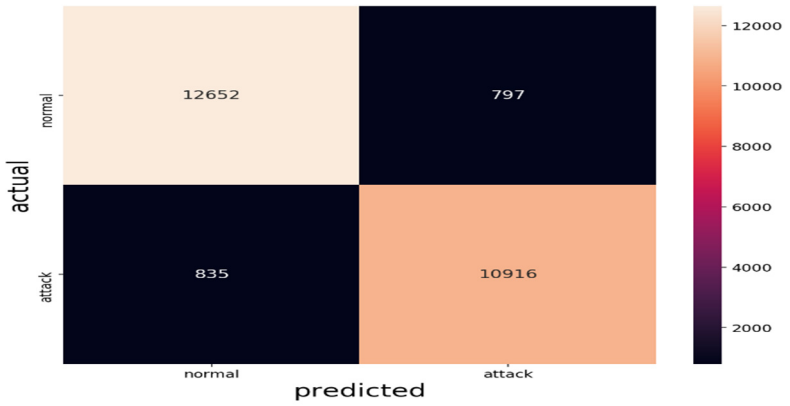

Figure 1. 2-classes (Normal and Attack)

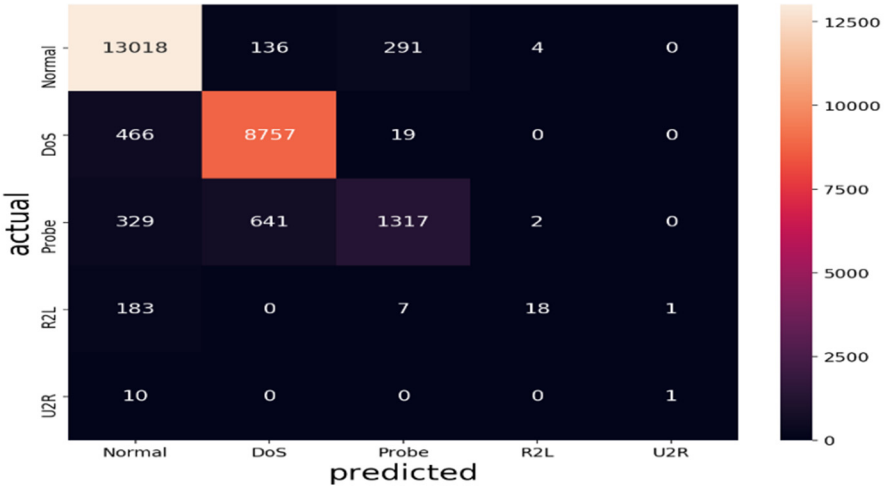

Figure 2. Confusion matrix: 5-classes (1 Normal and 4 Attacks: DoS, Probe, U2R, R2L)

Experiments related to the accuracy dynamics with respect to the number of epochs and the training time were also conducted (as shown in Figure 3, Figure 4, and Figure 5). Based on the number of epochs, the detection accuracy dynamics of the DBM are depicted in Figure 3. As illustrated, the DBM's DDoS attack detection accuracy gradually decreases as the number of epochs increases from 30 to 100. But the DBM's DoS attack detection accuracy gradually increases as the number of epochs increases from 3 to 30. As seen from Figure 3, when the number of iterations over the data decreases, the model detects DDoS attacks with high precision, but as the number of iterations increases, the accuracy of the model for DDoS attacks detection gradually falls. Figure 4 depicts the time consumption of the training with respect to the number of epochs. As given, the time consumption increases as the number of epochs increase from 3 to 30. Starting from 30 epochs, the computation time did not evolve anymore as the epoch increased. This, as well as the observations mentioned above, are due to the fact that the best model's parameters were obtained around this point and the system stopped learning.

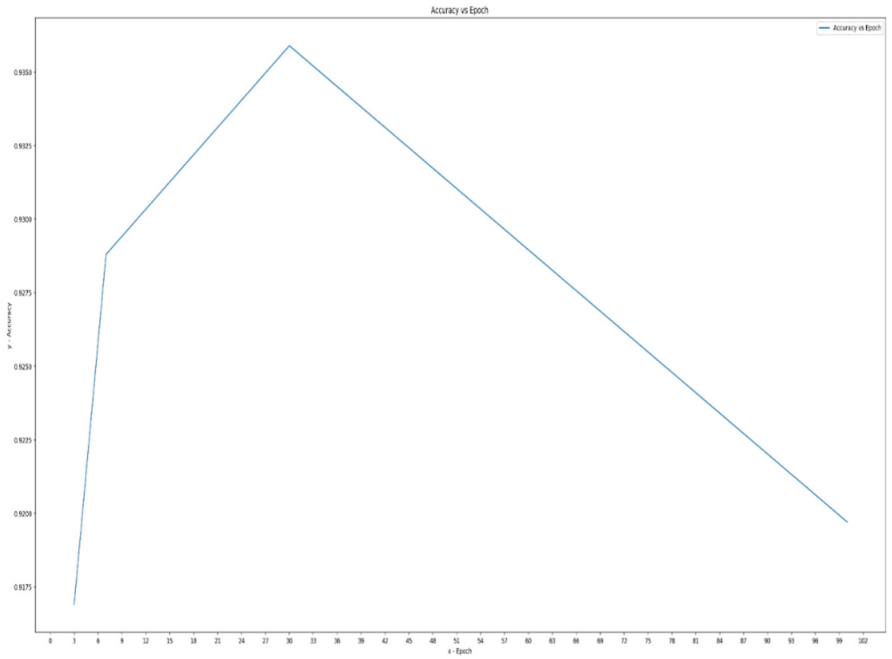

Figure 3. Detection rate dynamics of the DBM by number of epochs

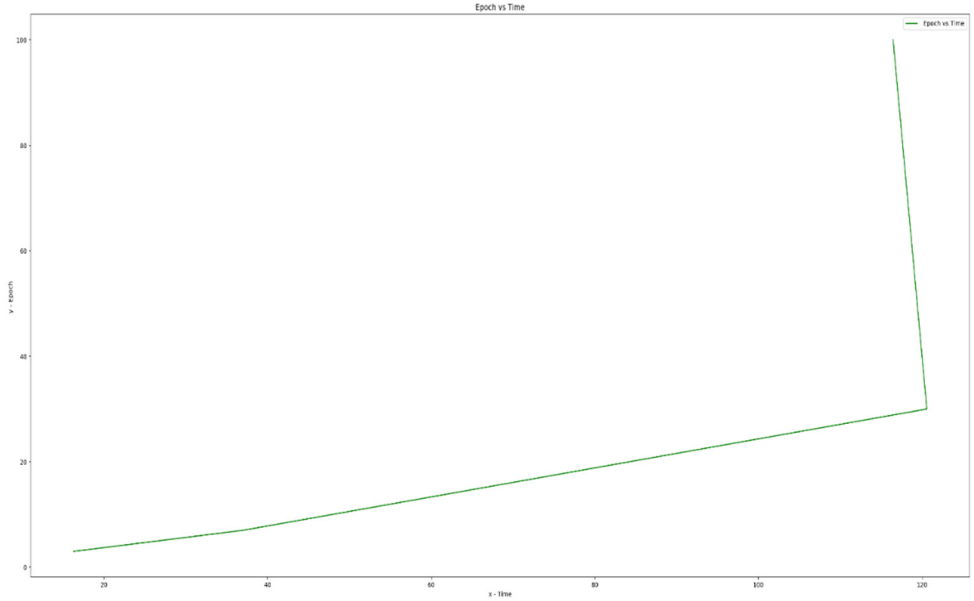

Figure 4. Epoch rate dynamics of the DBM over time consumption

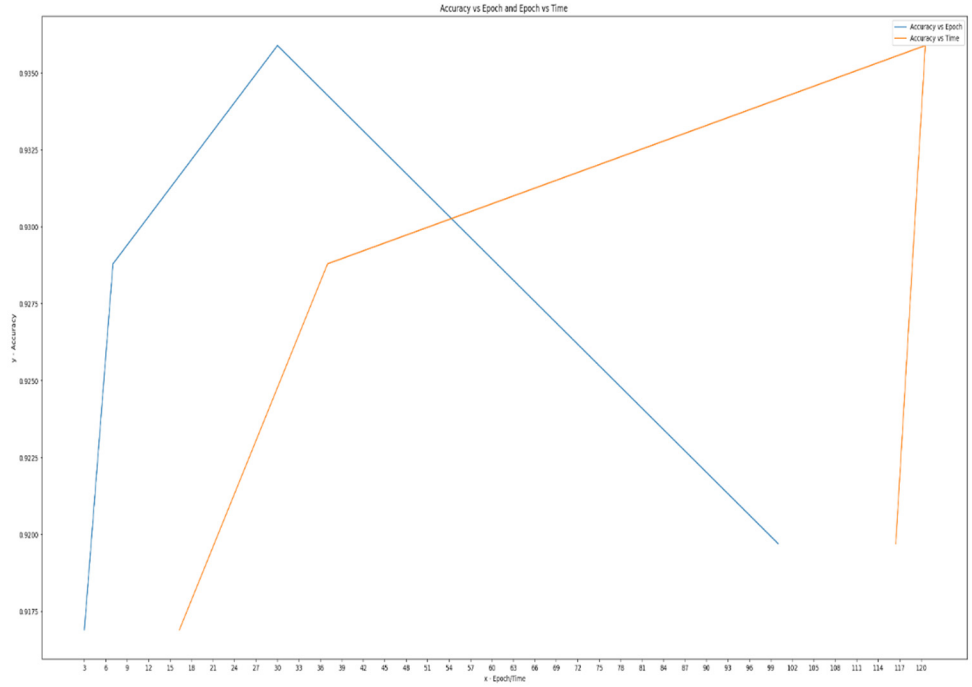

Figure 5. Accuracy dynamics as a function of Epochs and Time consumption

5.2 Results of Comparative Analysis

The performance of the proposed DBM for detecting DDoS attacks is evaluated using knowledge discovery in databases cup 99 (KDDCUP99) (Tavallaee et al., 2009) dataset, which is the detection of another DDoS attack benchmark dataset different from the NSL-KDD dataset used during the training of the proposed model. We passed kDDCUP99 to the developed model to observe and comparing performances. In Table 4 and Table 5, comparative analysis of the proposed DBM results on the NSL-KDD dataset with existing methods using NSL-KDD as a benchmark dataset on one hand and the proposed DBM results on the KDDCUP99 dataset with existing methods using KDDCUP99 as benchmark dataset, on the other hand, is described. The proposed DBM results on NSL-KDD outperform the results of the methods in (Javaid et al., 2016) and (Imamverdiyev & Abdullayeva, 2018). The accuracy of the proposed DBM was 93.52% while the model in (Imamverdiyev & Abdullayeva, 2018) had an accuracy of 78%, and (Javaid et al., 2016) had an accuracy of 88.39%. The proposed DBM results on KDDCUP99 performed better than the results of the method in (Li et al., 2015). The model in

(Sarigiannidis et al., 2015) achieved better performance on NSL-KDD than the developed model though the details of the DL model adopted in (Sarigiannidis et al., 2015) were not mentioned.

Table 3. Evaluation of the effectiveness of the proposed model on the NSL-KDD dataset with existing

Author	Method	Accuracy on NSL-KDD
[12]	Sparse autoencoder + Softmax	88.39%
[18]	Deep Restricted Boltzmann Machine	78%
[10]	Deep networks + Softmax	99%
The model developed in this study	**DBM + Softmax**	**93.52%**

Table 4. Evaluation of the effectiveness of the proposed model on KDDCUP99 dataset with existing

Author	Method	Accuracy on KDDCUP99
[14]	AutoEncoder + DBN	92.10%
Model developed in this study	**DBM + Softmax**	**97.95%**

6. Conclusion and Future Work

Internet of Things (IoT) is ubiquitous and providing adequate security to IoT is inevitable. This study explored the ability of Deep Learning approaches to detect DDoS attacks in IoT networks with regard to IoT peculiarities. In particular, we investigated it whether applying fog computing principles could make it possible to employ Deep Learning algorithms in IoT. Thus, a deep learning based-model for detecting distributed denial of service attacks in the Internet of Things was developed and hosted on fog nodes. The results of the simulation experiments showed that the proposed DBM efficiently detects DDoS attacks and outperforms previously reported works. This study contributes to knowledge by establishing a blueprint and a deep learning based-model for detecting distributed denial of service attacks in IoT.

The developed IDS with DBM for DDoS in IoT evaluation was limited to two different datasets. More datasets could be used to evaluate the proposed model in future works. Moreover, more improvement in the accuracy of the DBM by fine-tuning its parameters can be considered. Experimenting with the behavior of the developed model in a network with more nodes can be attempted. The developed model was also limited to DDoS attacks. It may also be expanded to consider more cybersecurity attacks such as Wormhole attacks, Sinkhole attacks, Sybil attacks, Selective Forwarding Attacks, etc all of which jeopardize the existence of IoT.

References

Alqahtani, B., & AlNajrani, B. (2020, October). A Study of Internet of Things Protocols and Communication. In *2020 2nd International Conference on Computer and Information Sciences (ICCIS)* (pp. 1-6). IEEE. https://doi.org/10.1109/ICCIS49240.2020.9257652

Arış, A., Oktuğ, S. F., & Yalçın, S. B. Ö. (2015, May). Internet-of-Things security: Denial of Service Attacks. In *2015 23rd Signal Processing and Communications Applications Conference (SIU)* (pp. 903-906). IEEE. https://doi.org/10.1109/SIU.2015.7129976

Arnaboldi, L., & Morisset, C. (2017, September). Quantitative Analysis of DoS attacks and Client Puzzles in IoT Systems. In Livraga G. & Mitchell C. (Eds.), *Security and Trust Management.* STM 2017. Lecture Notes in Computer Science, vol 10547. Springer, Cham. https://doi.org/10.1007/978-3-319-68063-7_16

Chen, X., Makki, K., Yen, K., & Pissinou, N. (2009). Sensor Network Security: A Survey. *IEEE Communications surveys & tutorials, 11*(2), 52-73. https://doi.org/10.1109/SURV.2009.090205

Cho, K. H., Raiko, T., & Ilin, A. (2013, August). Gaussian-Bernoulli Deep Boltzmann Machine. In *The 2013 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-7). IEEE. https://doi.org/10.1109/IJCNN.2013.6706831

Denise, B. (2020). DDoS Breach Costs Rise to over $2M for Enterprises finds Kaspersky Lab Report. Retrieved from https://usa.kaspersky.com/about/press-releases/

Diro, A. A., & Chilamkurti, N. (2018). Distributed Attack Detection Scheme Using Deep Learning Approach Internet of Things. *Future Generation Computer Systems, 82*, 761-768. https://doi.org/10.1016/j.future.2017.08.043

Elsaeidy, A., Munasinghe, K. S., Sharma, D., & Jamalipour. A. (2019). Intrusion Detection in Smart Cities Using Restricted Boltzmann Machines. *Journal of Network and Computer Applications, 135*, 76-83. https://doi.org/10.1016/j.jnca.2019.02.026

Fu, R., Zheng, K., Zhang, D., & Yang, Y. (2011). An Intrusion Detection Scheme Based on Anomaly Mining in Internet of Things. In *IET International Conference on Wireless, Mobile and Multimedia Networks*, Beijing, Nov. 2011, pp. 315-320.

Haddadpajouh, H., Mohtadi, A., Dehghantanaha, A., Karimipour, H., Lin, X., & Choo, K. K. R. (2020). A Multikernel and Metaheuristic Feature Selection Approach for IoT Malware Threat Hunting in the Edge Layer. *IEEE Internet of Things Journal, 8*(6), 4540-4547. https://doi.org/10.1109/JIOT.2020.3026660

Imamverdiyev, Y., & Abdullayeva, F. (2018). Deep Learning Method for Denial of Service Attack Detection Based on Restricted Boltzmann Machine. *Big data, 6*(2), 159-169. https://doi.org/10.1089/big.2018.0023

Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2016). A Deep Learning Approach for Network Intrusion Detection System. *EAI Endorsed Transactions on Energy Web, 16*(9). https://doi.org/10.4108/eai.3-12-2015.2262516

Kang, M. J., & Kang, J. W. (2016). A Novel Intrusion Detection System Using Deep Neural Network for In-vehicle Network Security. In *2016 IEEE 83rd Vehicular Technology Conference (VTC Spring)*. https://doi.org/10.1109/VTCSpring.2016.7504089

Kasinathan, P., Pastrone, C., Spirito, M. A., & Vinkovits, M. (2013, October). Denial-of-Service Detection in 6LoWPAN Based Internet of Things. In *2013 IEEE 9th international conference on wireless and mobile computing, networking and communications (WiMob)* (pp. 600-607). IEEE. https://doi.org/10.1109/WiMOB.2013.6673419

Latif, S., Idrees, Z., Zou, Z., & Ahmad, J. (2020, August). DRaNN: A Deep Random Neural Network Model for Intrusion Detection in Industrial IoT. In *2020 International Conference on UK-China Emergin Technologies (UCET)* (pp. 1-4). IEEE. https://doi.org/10.1109/UCET51115.2020.9205361

Li, Y., Ma, R., & Jiao, R. (2015). A hybrid Malicious Code Detection Method Based on Deep Learning. *International Journal of Security and Its Applications, 9*(5), 205-216. https://doi.org/10.14257/ijsia.2015.9.5.21

Li, Y., Xu, Y., Liu, Z., Hou, H., Zheng, Y., Xin, Y., ... & Cui, L. (2020). Robust Detection for Network Intrusion of Industrial IoT Based on Multi-CNN Fusion. *Measurement, 154*, 107450. https://doi.org/10.1016/j.measurement.2019.107450

Ma, T., Wang, F., Cheng, J., Yu, Y., & Chen, X. (2016). A Hybrid Spectral Clustering and Deep Neural Network Ensemble Algorithm for Intrusion Detection in Sensor Networks. *Sensors, 16*(10), 1701. https://doi.org/10.3390/s16101701

Pang, G., Shen, C., Cao, L., & Hengel, A. V. D. (2021). Deep Learning for Anomaly Detection: A review. *ACM Computing Surveys (CSUR), 54*(2), 1-38. https://doi.org/10.1145/3439950

Raza, S., Wallgren, L., & Voigt, T. (2013). SVELTE: Real-time Intrusion Detection in the Internet of Things. *Ad hoc networks, 11*(8), 2661-2674. https://doi.org/10.1016/j.adhoc.2013.04.014

Robert J., & Wang, E. (2020). Distributed Denial of Service (DDoS) Attacks and IoT Security. Retrieved from https://www.grin.com/document/371086

Salakhutdinov, R., & Hinton, G. (2019). *Deep Boltzmann Machines, Artificial Intelligence and Statistics* (pp. 448-455).

Sarigiannidis, P., Karapistoli, E., & Economides, A. A. (2015, June). VisIoT: A Threat Visualisation Tool for IoT Systems Security. In *2015 IEEE International Conference on Communication Workshop (ICCW)* (pp. 2633-2638). IEEE. https://doi.org/10.1109/ICCW.2015.7247576

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009, July). A Detailed Analysis of the KDD CUP99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications* (pp. 1-6). IEEE. https://doi.org/10.1109/CISDA.2009.5356528

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P. A., & Bottou, L. (2010). Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a local denoising criterion. *Journal of machine learning research, 11*(12).

Yadav, S., & Subramanian, S. (2016, March). 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). IEEE. https://doi.org/10.1109/ICCTICT.2016.7514608