Internet of Things Intrusion Detection: A Deep Learning Approach

Ahmed Dawoud
School of Computer, Data, and Mathematical Sciences
Western Sydney University, Sydney, Australia

Omid Ameri Sianaki
Victoria University Business School, Victoria University, Melbourne, Australia

Seyed Shahristani
School of Computer, Data, and Mathematical Sciences
Western Sydney University, Sydney, Australia.

Chun Raun
School of Computer, Data, and Mathematical Sciences
Western Sydney University, Sydney, Australia.

***Abstract*— The Internet of Things (IoT) is a shifting paradigm that allows the integration of billions of devices with the Internet. With its wide range of application domains, including smart cities, smart homes, and e-health, the IoT has created new challenges, particularly security threats. Traditional security solutions, such as firewalls and intrusion detection systems, need amending to fit the new networking paradigm. Given the recent advances in machine learning, we investigated the use of deep learning algorithms for anomaly detection. The IoT collects a massive amount of data from the environment, and deep learning is based on a set of algorithms striving for the data. Intrusion detection systems are used to expose network threats and are an effective means of protecting network assets. Anomaly detection is a conventional intrusion detection approach that separates normal and abnormal network traffic using statistical, rule-based, or machine learning models. Of the machine learning models, deep learning is a neural network algorithm that has provided breakthroughs in domains such as object and voice recognition. However, there are limitations in applying deep learning to network anomaly detection. This paper proposes a novel anomaly detection framework based on unsupervised deep learning algorithms for revealing network threats. Our research explores the applicability of deep learning to detect anomalies by evaluating the use of Restricted Boltzmann machines as generative energy-based models against Autoencoders as non-probabilistic algorithms. The study provides an in-depth analysis of unsupervised deep learning algorithms. The simulations studies show ≈ 99% detection accuracy, which is significantly improved compared to the closely related work.**

Keywords*— *Network intrusion detection, Unsupervised deep learning, Autoencoders, Restricted Boltzmann machines.

I. INTRODUCTION

The Internet of Things (IoT) expanded the Internet's capability through its ability to connect smart objects. Advances in wireless communication, embedded systems, and sensor technologies have accelerated IoT technologies in various applications fields. However, dense connectivity increases the risk of privacy and security threats. Significant challenges of the IoT include the heterogeneous composition of the network, the adoption of widely distributed architecture, particularly in widely distributed IoT applications, for instance, IoT environmental applications, smart city, and smart power grids. Additionally, the new protocols vector to address specific concerns related to power limitation and computation processing of network devices [1-4]. The IoT threat list has been expanded with the onset of new attacks, such as object cloning, firmware replacement, and extraction of security parameters.

For decades, conventional networks have employed firewalls and intrusion detection systems as standard security solutions to network threats. However, innovative solutions are required for the unprecedented security threats emerging from recent advances in internetworking such as the IoT, Software Defined Network (SDN), and grid computing.

Intrusion detection systems include both signature-based and anomaly-based approaches to detecting threats [5]. Signature-based methods suffer limitations because they are unable to recognize attacks that do not exist in their threat profiles. However, anomaly detection (AD) can overcome the precision deficiencies of signature-based systems [5-7].

AD techniques are classified as either statistical or machine learning [8,9]. The latter includes algorithms such as Support Vector Machines (SVM), K-means, Neural Network, and Principal Components Analysis (PCA), all of which fail to provide high detection accuracy [5,4,10,11]. However, the current breakthroughs in training neural network with multiple hidden layers are promising [13, 14].

DL architecture consists of a multilayered neural network with additional layers connecting the first or input layer with the last or output layer; these layers are known as hidden layers. The input layer represents features included in the dataset, while the last layer represents the neural network output for an example set of classes or clusters. Although DL neural networks have existed for an extended period, they have been unable to implement a trained neural network for various reasons, including the vanishing gradient descent, where the gradient with minimal value preventing updating the weights effectively. Other limitations with a classical neural network is a poor generalization and the need for intensive computation power [7].

DL algorithm with a restricted Boltzmann machine (RBM) was devised, leading to innovative algorithms' development to solve generalization problems. These algorithms are categorized as supervised or unsupervised [13]. However, supervised algorithms are limited by their need for labeled data and their inability to detect new threats [6].

978-1-7281-2547-3/20/$31.00©2020 IEEE

In this study, we introduce a DL based anomalies detection system to identify the threats in network packet traffic. We investigated the accuracy and feasibility of DL in network traffic anomalies detection and conducted a comparative study between two DL unsupervised algorithms i.e., AEs and RBMs. This research employed unsupervised DL (USDL) algorithms as the most capable method of detecting novel attacks [5, 6]. However, USDL algorithms cannot be used as efficient standalone systems because they assume that anomalies occur less frequently than normal samples. If this assumption is not valid, as seen in the frequency of Distributed Denial of Service (DDoS) attacks, the algorithm may suffer a high false alarms rate [14]. Therefore, we adapted the algorithms for AD purposes.

This paper makes three contributions:

1. Proposes and implements a framework integrating two algorithms that apply to different networking models, including conventional networks, SDNs, and IoT;

2. Compares the accuracy of two major USDL algorithms—RBMs and AEs (to the best of our knowledge, experimental comparative studies evaluating AEs and RBMs in network traffic AD have not been conducted);

3. Improves detection accuracy: The framework achieved over 99% detection accuracy, which is significantly improved compared to previously achieved in several studies discussed in section 6.

The rest of this study is structured as follows. Section 2 presents the background on DL and AD and related research, focusing on USDL, specifically AEs and RBMs. Section 2 also introduces a literature review on the currently ongoing research on DL and network traffic AD. Section 3 discusses the detection framework in which the first phase has two unsupervised modes (AE and RBM). Section 4 outlines the experimental study in which the framework was implemented. Section 5 analyses the results using confusion matrix statistics and compares AE and RBM for precision and accuracy. Section 6 concludes the paper.

II. Background and Related work

Various studies have introduced SDN-based solutions to improve IoT privacy and security aspects. Some studies have considered a domain-based architecture for networks with multiple domains [15, 16]. The separation of domains enhances the availability of resources, but robust performance analyses have not yet been conducted. Bhunia and Gurusamy [17] proposed an SDN-based detection system for DoS attacks on the IoT, achieving a precision rate of 98%. In another study, Chakrabarty et al. introduced an SDN-based IoT solution known as Black SDN, where the payloads and Metadata of the packets are encrypted [18]. One major limitation with this approach is the routing complications, as the headers need to be decrypted to identify the source and destination addresses. Jararweh et al. introduced an architecture which entirely depends on SDN known as SDIoT to improve IoT management by enhancing the routing, recording, and securing data produced from IoT devices [19].

DL is a set of non-linear algorithms used for multilayered models. DL algorithms are categorized as supervised and unsupervised. In supervised DL learning, the training dataset contains labeled data. The supervised learning algorithm trained to predict p(y|x), where x and y are the inputs and outputs, respectively. Supervised learning can be used to solve the regression and classifications problems.

On the other hand, unsupervised algorithms have only unlabelled datasets. USDL algorithms target to find the probability distribution of the given dataset. Unsupervised algorithms are suitable to solve highly dimensioned data, clustering, and noise removal problems.

An AE is a neural network consisting of two components:

The first component is An encoder, which is a deterministic mapping function (f_θ) in which an input vector (x) is transformed into a hidden representation (y), where θ=[2] and f_θ (x)≈x';

A decoder: reconstructs the hidden representation (y) to the reconstructed input (x') via g_θ.

AEs measure the reconstruction errors between the inputs (x) and the reconstructed x' to minimize error (information loss) using:

$$J(W) = \sum ||x_n - x'_n||$$

where J(W) is the loss function used to reduce the loss, and:

$$\text{Arg min } (J(W))_{\{w,w',b,b,\}}$$

where w and b are encoder weights and biases, respectively, and w' and b' are decoder weights and biases.

Squared error is used to compute the loss where there are multiple options for the optimization functions, for example, gradient descent and its variations like stochastic and Momentum, Adam Optimizer, and Adagrad

RBM is an energy based, stochastic network models in which each neuron is assigned stochastic scalar energy. Unlike Boltzmann Machines, there is a restriction on connections between layers. During the learning phase, the energy function is updated to ensure the model has the required properties. The probability distribution of the energy function is estimated using

$$p(x) = \frac{e^{-E(x)}}{Z}$$

where Z is the partitioning function defined as

$$Z = \sum_x e^{-E(x)} .$$

The Boltzmann machine's energy function is defined as $E\ (x) = \ -x^T\ W\ x - b^T x$

where W is the weight matrix, and b is the bias parameter.

RBM is a variation of Boltzmann machines, with hidden units used to improve RBM performance.The energy function of RBMs is represented by

$$E\ (v,h) = \ -b'v - c'h - h'Wv,$$

Where b' and c' are the biases for visible and hidden units, respectively, and W is the weight of connections between the hidden and visible units.

A comprehensive study evaluated seven unsupervised learning algorithms [20] and concluded that all algorithms performed poorly in detecting remote-to-local attacks, while SVMs and Y-means performed better than other methods in detecting user-to-root traffic. C-means and fuzzy logic algorithms delivered the most unsatisfactory performance.

Projecting highly dimensional data on a lower subspace that still preserves the important information, that is referred to as dimensionality reduction. For anomalies detection, projecting data on lower subspace will increase the probability of finding the abnormalities [21]. A major example of dimensionality reduction algorithms is PCA, which can learn linear relationships:

f f(x) = W(x)^T+b,

where x is input and x ∈ R^(d_x).

Another non-linear variation of PCA is called Kernel PCA. It is used to represent non-linear relationships; unlike PCA in KPCA will project the data to higher dimensions before reducing it. AE algorithms imply dimensionality reduction because they convert data into new representations preserving the most critical features during the encoding, then reconstruct the original input during decoding. Several studies have studied AEs against PCA and its non-linear variations, kernel PCA, as a dimensionality reduction algorithm. One study found that AEs and demonizing AEs performed significantly better than PCA and kernel PCA. However, when the authors tested a discriminative RBM in a new network, the results were not encouraging.

TABLE 1. RELATED RESEARCH SUMMARY

Research	DL Algorithm		Classic Machine Learning Algorithms			Dataset
	AE	RBM	SVM	PCA	KPCA	
[20]	✓			✓	✓	Images
[21]		✓	✓	✓	✓	KDD99, USENET, Thyroid
[22]		✓				KDD99 and bot data
[23]	✓					Generated traffic
[24]		✓				KDD99
[25]	✓			✓	✓	Lorenz, sat-A
[26]	✓					KDD99
[27]	✓					KDD99

Note: DL = deep learning; AE = autoencoder; RBM = restricted Boltzmann machine; SVM = support vector machine; PCA = principal component analysis; KPCA = kernel PCA.

Table 1 summarizes the related research papers with approaches deployed and datasets used for the simulations studies. Various researchers proposed the utilization of DL to improve the detection accuracy of classical ML algorithms. Mostafa et al. deployed detection systems that include the deep belief network as a dimensionality reduction tool stage before using the SVM classifier. While the SVM and deep belief network scored 88% and 90%, the hybrid approach achieved 93% accuracy [25]. The authors applied the four algorithms to a generated dataset and two non-artificial datasets. Fiore et al. [27] used a semi-supervised DL tool for network AD. Dong et al. presented a study to compare Bayes network, C4, and SVM with a system based on RBM and SVM components. Their simulations showed that the hybrid approach was superior for detecting various attacks, including DoS and user-to-root attacks [20]. In another broader paper on outliers detection, Zhai et al. introduced a deep structured energy-based network to compare algorithms in two different decision boundaries to compare the performance of five algorithms used for anomalies detection; these algorithms included SVM and PCA. The authors used static, spatial, and sequential datasets. KDD99 was one of these datasets they used in their experimental studies. Their results showed that PCA and kernel PCA performed similarly to or better than other methods [22].

This paper presents a DL-based framework that demonstrates the potential of DL in network AD. Our approach produced results that excelled those of previously mentioned works. Additionally, we compared two USDL algorithms.

III. PROPOSED FRAMEWORK

Figure 1 shows the suggested threat detection system as an integrated component in an IoT architecture. The suggested IDS includes two stages. The primary stage represents the unsupervised DL network, which is consuming the inputs features through multiple chunks to find patterns in the data then establish the neural network model. The output from the first stage is further processed by clustering algorithms to create clusters of normal/abnormal network traffic. To validate our results generated by the proposed system, we applied two clustering algorithms, K-means, and mean-shift. The final output includes various clusters; each cluster contains normal or abnormal traffic. The main components of the DL algorithm are the model, the cost function, and the optimizer. The inputs (features) are fitted to the model before the cost (information loss) between the input data and the model's output. Through multiple iterations of forwarding and backward propagation, the loss is reduced to an acceptable form for generalization.

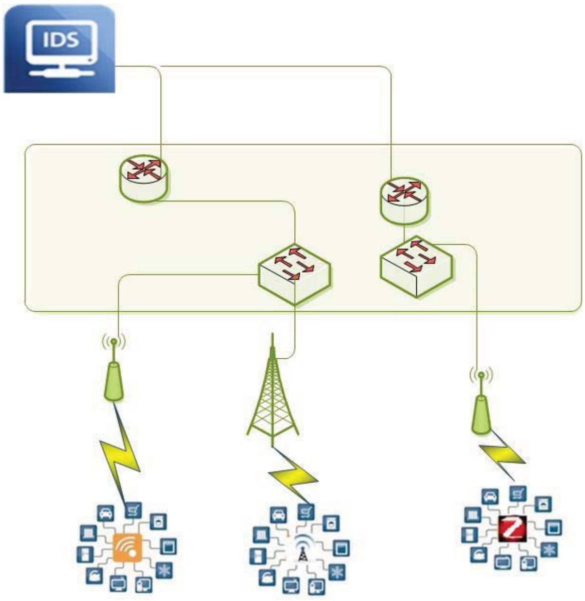

Fig. 1. Detection framework design integrated in IoT environment

In USDL, a decision boundary must be produced to identify normal/abnormal records. Decision boundaries are values utilized by the systems to detect anomalies. For DL representation learning algorithms, both AEs and RBMs may be used to compute the reconstruction errors. During the second stage, the decision boundaries generated will be clustered into normal and abnormal packet traffic. Hence, we provided four different configurations consisted of two DL algorithms and two clustering algorithms to offer comprehensive results for the analysis.

The suggested system uses DL and a simpler algorithm during the second stage of processing. Several current proposed methods have used DL in two phases: an unsupervised and a supervised phases. Implementing two heavyweight computation components significantly increases the processing overheads. Additionally, the features used in network traffic are limited; for example, the number of elements in an IP packet; DL algorithms show better performance with highly dimensioned data like images or audio.

IV. Simulations Studies

The implemented system for simulations studies consisted of an input layer representing the elements in an IP packet in the captured network traffic, then a number of hidden layers, in the end, an output layer that reconstructs the input with minimum loss. Reconstruction errors were then computed for classification. USDL can not be used directly for classification or clustering. Therefore, instead of using the neural network output, we considered the difference between the input and the reconstructed output.

A. Normalizing the Dataset

A KDD99 dataset has been vastly used in the simulations for intrusion detection and artificial intelligence. Also, it is a realistic network traffic capture, which includes ≈ five million traffic records for training and ≈ 300k records for testing. It included four types of attacks: DoS attacks (which represented the largest traffic record), probe (active information gathering) attacks, remote-to-local attacks, and user-to-root attacks (privilege escalation).

Each record contained 41 features, which are a mix of continuous and discrete values. A sample record is as follows:

l.0,tcp,http,SF,227,633,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,12 ,0.00,0.00,0.00,0.00,1.00,0.00,0.25,255,255,1.00,0.00,0.01,0. 00,0.00,0.00,0.00,0.00,normal

For the network to process the input data, the data must be converted to the numerical format. The dictionary method was used for the input normalization. The entire dataset was scanned to identify the alphabetical values, as shown in the previous sample, for example, protocols names like TCP, HTTP, and SF. Then these values were substituted with numerical values, as shown in the sample below. Also, the labels were replaced with the values 10, and 01 replaced normal and abnormal, respectively. These labels were used for results verification, as we are using unsupervised learning.

Labels 0 1 |features 0.0 2.0 22.0 60.0 105.0 146.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 255.0 254.0 1.0 0.01 0.0 0.0 0.0 0.0 0.0

As an additional step for the data normalization we scaled the numerical values to be in closed interval of [0- 1]. This step was included as some experiments shows the AE and RBM perform well with binaries and the [0-1] interval

|labels 0 1 |features 0.0 0.0 0.67 0.31 0.85 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 1.0 1.0 0.01 0.01 0.0 0.0 0.0 0.0

B. Experiments setups

We used Tensorflow for the DL development library. As the name suggests, TensorFlow shows matrix flows using a graph model, which includes nodes representing mathematical operations and edges representing multidimensional data arrays (tensors).

The first stage of the experiment involved building the AE network. The AE consisted of two passes—the encoder and the decoder—composed of multiple layers. The dataset records (training samples = 41 features) were loaded into the Tensorflow input tensor dimension. As part of our model, the weights and biases tensors were created for the encoder and decoder components. The dimensions of weights and biases depended on the number of neurons (units in the hidden layer). For instance, if we decoded the input into five units, this created (41, 5) tensors, where 41 was some input units (features of one network traffic record) plus 41 biases. The same dimensions were used for the decoder. Subsequently, we trained the network using logits as the forward pass's activation function before applying the activation function to reconstruct the record from the decoded units, weights, and biases for the output.

The second stage involved comparing the original data with the reconstructed output. We used a cost function (e.g., the squared error function) to compute data loss. The third stage was to minimize the cost (data loss). Several optimization algorithms (e.g., Adam optimizer) were used to mitigate the loss or reconstruction rate.

Once the network was established after iterations of sweeping through the data batch, the second testing stage began. The network model is fed with data samples to generate reconstructed outputs. To clarify, assuming the model could manipulate images, then the expected outcome is an analogous image (with minimal data loss).

For AD, we measured the difference between the input and the reconstructed output. If the difference was high (as per our defined thresholds), this meant that the input data were not sufficiently precise to be reconstructed. Additionally, the records with high reconstruction error can be classified as outliers. This concept was also valid for RBMs because both algorithms were used to reconstruct the input.

The algorithm performance depends on several factors:

- Features data type and domain (binary or decimal);
- Activation function (e.g., sigmoid is more effective for binary data, while Rectified Linear Unit ReLu is more effective for decimals);
- Loss function (e.g., Mean Square Error MSE, Mean Absolute Error, MAE and cross-entropy);

- Optimizer (i.e., stochastic gradient descent, Adam optimizer): the AE aims to reduce information loss by several iterations of forwarding and backward propagations.

C. *Simulation Results*

Figure 2 shows the distribution of reconstruction errors from the testing records. The vertical axis represents the error, while the horizontal axis represents the number of testing samples (400 data records). The records in the lower part of the graph represent traffic with a low number of errors, while those in the upper part of the graph represent traffic with a higher number of errors. We can use either a threshold of data loss or different groups of thresholds to define various attacks.

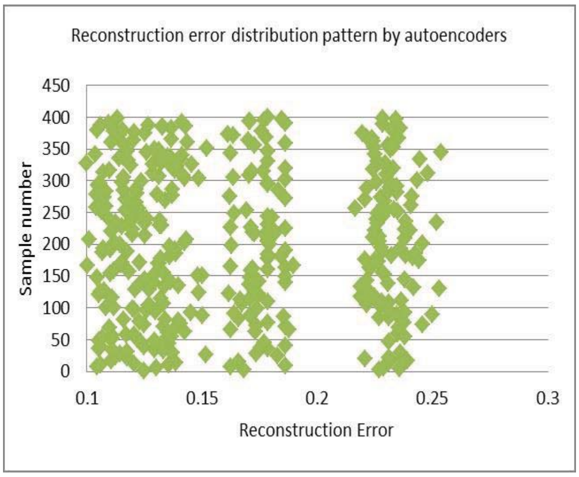

Fig. 2. Reconstruction error distribution pattern by autoencoders

Tables 2 and 3 summarize the confusion matrix statistics from the experiments. Table 2 displays the results for AE and RBM in conjunction with K-means. Accuracy represents how often the framework was correct. The AE with the lowest number of samples achieved the highest accuracy. Remarkably, accuracy declined as the number of samples increased.

Additionally, AE achieved the best accuracy in conjunction with the mean shift algorithm. The F1 score indicates precision (true positive results/total true positive by the framework) and recall (number of true positive results in the total sample). AE had the highest F1 score for both K-means and mean shift. AE with K-means had the lowest false discovery rate, while RBM with mean shift had a notable high discovery rate for 400 samples. Nevertheless, the results for RBM with mean shift for other sample numbers showed the same trend.

TABLE 2. AUTOENCODER (AE) AND RESTRICTED BOLTZMANN MACHINE (RBM) WITH K-MEANS (KM)

	AE 1300 KM	**RBM 1300 KM**	**AE 800 KM**	**RBM 800 KM**	**AE 400 KM**	**RBM 400 KM**
Sensitivity	0.9917	1	0.9974	1	1	1
Specificity	0.9943	0.9661	1	0.972	1	0.9747
Precision	0.9933	0.9443	1	0.9554	1	0.9645
Negative Predictive Value	0.9929	1	0.9976	1	1	1
False Positive Rate	0.0057	0.0339	0	0.028	0	0.0253
False Discovery Rate	0.0067	0.0557	0	0.0446	0	0.0355
False Negative Rate	0.0083	0	0.0026	0	0	0
Accuracy	0.9931	0.9785	0.9988	0.9825	1	0.985
F1 Score	0.9925	0.9714	0.9987	0.9772	1	0.9819

TABLE 3. AUTOENCODER (AE) AND RESTRICTED BOLTZMANN MACHINE (RBM) WITH MEAN SHIFT (MS)

	AE 1300 MS	**RBM 1300 MS**	**AE 800 MS**	**RBM 800 MS**	**AE 400 MS**	**RBM 400 MS**
Sensitivity	0.9932	1	0.9974	1	1	1
Specificity	0.9789	0.9684	0.9952	0.972	0.9808	0.7241
Precision	0.975	0.9483	0.9947	0.9466	0.9796	0.4793
Negative Predictive Value	0.9943	1	0.9976	1	1	1
False Positive Rate	0.0211	0.0316	0.0048	0.028	0.0192	0.2759
False Discovery Rate	0.025	0.0517	0.0053	0.0534	0.0204	0.5207
False Negative Rate	0.0068	0	0.0026	0	0	0
Accuracy	0.9854	0.98	0.9963	0.9813	0.99	0.78
F1 Score	0.984	0.9735	0.996	0.9725	0.9897	0.648

V. Comparison with Related Work

The authors of [23] used a discriminative RBM in the unsupervised pre-step. In their study, RBMs were deployed as either discriminative classifiers or as standalone supervised classifiers. Others [28] have proposed an AD framework based on a variation of the AE known as a non-symmetric deep AE. The authors benchmarked a revised version of the KDD99 dataset—NSL-KDD—by removing redundant records. A confusion matrix was used for the analysis data, showing an accuracy of ≈ 98%. Similar to other approaches discussed, this framework used all extracted features from the first phase as the input for the second phase [24].

AEs have been used for AD based on dimensionality reduction. However, accuracy has been relatively low compared with traditional PCA. Its variation achieved a significant improvement in some datasets [26]. However, as proposed in this paper, adding a simple algorithm to cluster the preprocessed data significantly improved accuracy. Another approach used K-means clustering in the preprocessing step, with results being fed into an SVM classifier. However, the performance was low, with only 90% accuracy [29]. Another work focused on improving K-means for intrusion detection in KDD99, showing inconsistent results ranging between 85% and 95% [30].

In contrast, the proposed framework is more stable, and its accuracy was noticeably better. Compared with all previously mentioned related works, the proposed framework (AE + K-means) presented in this paper outperformed with respect to accuracy. The frameworks presented in [24] and [29] used a similar approach; however, technically, the proposed framework in this research improves dimensionality reduction, while others have forwarded the learned features from the DL approach. Additionally, this comparative study shows the superiority of the adopted approach.

Table 4. Related Work Accuracy Results

Study	Algorithm	Dataset	Performance
[23]	DRBM	KDD99	Accuracy ≈ 84%

[29]	AE + classifier	KDD99	Accuracy = 97.85%
			Precision = 99.99%
			Recall = 97.85%
			F-score = 98.15%
			False alarm = 2.15%

[24]	Sparse AE	NSL-KDD	Accuracy ≈ 98% F-score ≈ 98.84%
[30]	SVM + K-means	KDD99	Accuracy = up to 90%
[26]	AE	Different dataset	Accuracy = 70–93%
[31]	K-means	KDD99	Accuracy = 85–95% for different samples

VI. Conclusion

Our research explored the applicability of deep learning to detect anomalies by evaluating the use of Restricted Boltzmann machines as generative energy-based models against Autoencoders as non-probabilistic algorithms. The study provided an in-depth analysis of unsupervised deep learning algorithms. The simulations studies showed ≈ 99% detection accuracy, which is significant improvement compared to the closely related work. The result showed the promising potential of DL in network AD. We recommend further investigations for DL in intrusion detection systems. Where there is a small number of network traffic features, network traffic data dimensionality is low and it requires further normalization to fit the model. For example, data should be normalized to binary values.

References

[1] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu and D. Qiu, ‘Security of the Internet of Things: Perspectives and challenges’, Wireless Netw., vol. 20, no. 8, pp. 2481–2501, Nov. 2014.

[2] F. A. Alaba, M. Othman, I. A. T. Hashem and F. Alotaibi, ‘Internet of Things security: A survey’, J. Netw. Comput. Appl., vol. 88, pp. 10–28, Jun. 2017.

[3] D. E. Kouicem, A. Bouabdallah and H. Lakhlef, ‘Internet of things security: A top-down survey’, Comput. Netw., vol. 141, pp. 199–221, Aug. 2018.

[4] O. Flauzac, C. González, A. Hachani and F. Nolot, ‘SDN based architecture for IoT and improvement of the security’, in 2015 IEEE 29th Int. Conf. Adv. Inf. Netw. Appl. Workshops, Gwangiu, South Korea, 2015, pp. 688–693.

[5] J. Li, Z. Zhao, R. Li, H. Zhang, AI-based two-stage intrusion detection for software defined IOT networks, IEEE IoT J. 6 (2) (2019) 2093–2102. https://doi.org/10.1109/JIOT.2018.2883344.

[6] M.H. Bhuyan, D.K. Bhattacharyya, J.K. Kalita, Network anomaly detection: Methods, systems and tools, IEEE Commun. Surv. Tut. 16 (1) (2014). 303–336. https://doi.org/10.1109/SURV.2013.052213.00046.

[7] A.A. Ghorbani, W. Lu, M. Tavallaee, Network Intrusion Detection and Prevention: Concepts and Techniques, Springer, New York, 2010.

[8] I. Mukhopadhyay, M. Chakraborty, S. Chakrabarti, A comparative study of related technologies of intrusion detection and prevention systems, J. Inf. Secur. 2 (1) (2011) 28–38. https://doi.org/10.4236/jis.2011.21003.

[9] E. Hodo, X. Bellekens, A. Hamilton, C. Tachtatzis, R.C. Atkinson, Shallow and deep networks intrusion detection system: A taxonomy and survey, CoRR, abs/1701.02145 (2017) 1–43. http://arxiv.org/abs/1701.02145.

[10] A. Patcha, J.-M. Park, An overview of anomaly detection techniques: Existing solutions and latest technological trends, Comput. Netw. 51 (12) (2007) 3448–3470. https://doi.org/10.1016/j.comnet.2007.02.001.

[11] D. Mudzingwa, R. Agrawal, A study of methodologies used in intrusion detection and prevention systems (IDPS), in: IEEE Southeastcon, Orlando, Florida, USA, IEEE, 2012, pp. 1–6.

[12] R. Sadoddin, A.A. Ghorbani, A comparative study of unsupervised machine learning and data mining techniques for intrusion detection, in: P. Perner (ed.), Machine Learning and Data Mining in Pattern Recognition. Lecture Notes in Computer Science, 4571 (2007), 404–418. https://doi.org/10.1007/978-3-540-73499-4_31.

[13] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, Cambridge, MA: MIT Press, 2017.

[14] G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (7) (2006) 1527–1554. https://doi.org/10.1162/neco.2006.18.7.1527.

[15] C. González, O. Flauzac, F. Nolot and A. Jara, ‘A novel distributed SDN-secured architecture for the IoT’, in 2016 Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS), Washington, DC, USA, 2016, pp. 244–49.

[16] C. González, S. M. Charfadine, O. Flauzac and F. Nolot, ‘SDN-based security framework for the IoT in distributed grid’, in 2016 Int. Multidisciplinary Conf. Comput. Energy Sci. (SpliTech), Split, Croatia, 2016, pp. 1–5.

[17] S. S. Bhunia and M. Gurusamy, ‘Dynamic attack detection and mitigation in IoT using SDN’, in 2017 27th Int. Telecommun. Netw. Appl. Conf. (ITNAC), Melbourne, Australia, 2017, pp. 1–6.

[18] S. Chakrabarty, D. W. Engels and S. Member, ‘A secure IoT architecture for smart cities’, in 2016 13th Annu. Consum. Commun. Netw. Conf. (CCNC), Las Vegas, NV, USA, 2016, pp. 812–813.

[19] Y. Jararweh, M. Al-Ayyoub, A. Darabseh, E. Benkhelifa, M. Vouk and A. Rindos, ‘SDIoT: A software defined based internet of things framework’, J. Ambient Intell. Humanized Comput., vol. 6, no. 4, pp. 453–461, Aug. 2015.

[20] B. Dong, X. Wang, Comparison deep learning method to traditional methods using for network intrusion detection, in: 2016 8th IEEE Int. Conf. Commun. Softw. Netw. (ICCSN), Beijing, China, IEEE, 2016, pp. 581–585.

[21] L. Zamparo, Z. Zhang, Deep autoencoders for dimensionality reduction of high-content screening data, CoRR, abs/1501.01348 (2015) 1–5. https://arxiv.org/abs/1501.01348

[22] S. Zhai, Y. Cheng, W. Lu, Z. Zhang, Deep structured energy based models for anomaly detection, in: Proc. 33rd Int. Conf. Mach. Learn., New York, NY, USA, IMCL, 2016, pp. 1100–1109.

[23] U. Fiore, F. Palmieri, A. Castiglione, A. De Santis, Network anomaly detection with the restricted Boltzmann machine, Neurocomput. 122 (2013) 13–23. https://doi.org/10.1016/j.neucom.2012.11.050.

[24] A. Javaid, Q. Niyaz, W. Sun, M. Alam, A deep learning approach for network intrusion detection system, in: 9th EAI Int. Conf. Bio-inspired Inf. Commun. Tech., New York, NY, USA, EAI, 2015, pp. 21–26.

[25] M.A. Salama, H.F. Eid, R.A. Ramadan, A. Darwish, A.E. Hassanien, Hybrid intelligent intrusion detection scheme, in: A. Gaspar-Cunha, R. Takahashi, G. Schaefer, L. Costa (Eds.), Soft Computing in Industrial Applications. Advances in Intelligent and Soft Computing, vol. 96, Springer, Berlin, 2001, pp. 293–303.

[26] M. Sakurada, T. Yairi. Anomaly detection using autoencoders with non-linear dimensionality reduction, in: 2014 2nd Workshop Mach. Learn. Sensory Data Anal., Gold Coast, Australia, MLSDA, 2014. https://doi.org/10.1145/2689746.2689747.

[27] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat and S. Venkatraman, "Deep Learning Approach for Intelligent Intrusion Detection System," in IEEE Access, vol. 7, pp. 41525-41550, 2019. doi: 10.1109/ACCESS.2019.2895334

[28] N. Shone, T. N. Ngoc, V. D. Phai and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection," in IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 41-50, Feb. 2018.

[29] A. Shrivastava and R. R. Ahirwal, “A SVM and K-means Clustering based Fast and Efficient Intrusion Detection System,” International Journal of Computer Applications, vol. 72, no. 6, pp. 25–29, 2013.

[30] L. Tian and W. Jianwen, "Research on Network Intrusion Detection System Based on Improved K-means Clustering Algorithm," 2009 International Forum on Computer Science-Technology and Applications, Chongqing, 2009, pp. 76-79.

.