International Journal of Electrical and Computer Engineering (IJECE)
Vol. 12, No. 3, June 2022, pp. 3265~3275
ISSN: 2088-8708, DOI: 10.11591/ijece.v12i3.pp3265-3275

A feature selection method based on auto-encoder for internet of things intrusion detection

Ahmed Fahad Alshudukhi, Saif Ahmed Jabbar, Basel Alshaikhdeeb
Center for Software Technology and Management (Softam), Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Bangi, Malaysia

Article Info

Article history:

Received May 5, 2021
Revised Dec 28, 2021
Accepted Jan 17, 2022

Keywords:

Auto-encoder
Feature selection
Internet of things
Intrusion detection
Neural network

ABSTRACT

The evolution in gadgets where various devices have become connected to the internet such as sensors, cameras, smartphones, and others, has led to the emergence of internet of things (IoT). As any network, security is the main issue facing IoT. Several studies addressed the intrusion detection task in IoT. The majority of these studies utilized different statistical and bio-inspired feature selection techniques. Deep learning is a family of techniques that demonstrated remarkable performance in the field of classification. The emergence of deep learning techniques has led to configure new neural network architectures that is designed for the feature selection task. This study proposes a deep learning architecture known as auto-encoder (AE) for the task of feature selection in IoT intrusion detection. A benchmark dataset for IoT intrusions has been considered in the experiments. The proposed AE has been carried out for the feature selection task along with a simple neural network (NN) architecture for the classification task. Experimental results showed that the proposed AE showed an accuracy of 99.97% with a false alarm rate (FAR) of 1.0. The comparison against the state of the art proves the efficacy of AE.

Corresponding Author:

Ahmed Fahad Alshudukhi
Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia
Bangi, Malaysia
Email: Alshudukhi@me.com

1. INTRODUCTION

The last decade has witnessed a dramatic evolutions in gadgets where wide range of devices have become connected to the internet such as sensors, cameras, smartphones and others [1]. Such evolution has led to the emergence of internet of things (IoT) as a new research area that explores the utilization of the massive number of connected devices in order to perform specific tasks [2]. One of the IoT applications is the smart house that can take the advantage of cameras, sensors, and smartphone to build an intelligent system for alerting the owners regarding suspicious and emerging events that could happen during his absence. In addition, a framework for a smart hospital is also proposed by utilizing medical devices to determine the priority and emergency list of patients [3].

This massive evolution of technology has brought numerous challenges, one of the concerning challenges is the security. The protection of IoT network from traditional threats or intrusions such as viruses, worms, Trojan horses and others is the main challenge [4]. The security is representing an essential demand especially if the IoT network is related to medical or private agencies which makes the violation of private information is intolerable [5].

In fact, intrusion detection (ID) is a research field that is examining the identification of any abnormal activity conducted in certain networks [6]–[9]. ID has been investigated extensively in the last two decades where wide range of techniques have been proposed for the detection task. However, the intrusions on specified networks such as the IoT would have different characteristics which requires new techniques that can address these differences.

One of the significant techniques that can address the characteristics of IoT intrusion detection is the feature selection where the aim is to analyze the features of IoT intrusions in order to identify the most important subset of features. Recently, many researchers have examined the feature selection in IoT detection for example, Gharaee and Hosseinvand [10] have examined the problem of dimensionality of feature space within the intrusion detection in IoT. The authors have focused on the challenging task of reducing the false positive rate within the intrusion detection. For this purpose, the authors have proposed a combination of genetic algorithm (GA) as a feature selection/reduction technique along with support vector machine (SVM) classifier. The dataset used in the experiments was UNSW-NB15 in which the average accuracy of detection was 93.25% with a false alarm rate (FAR) of 8.6.

Similarly, Khammassi and Krichen [11] have proposed a feature selection approach based on a wrapper technique. The authors have attempted to identify the most significant features that might impact the accuracy of intrusion detection. Therefore, a wrapper technique has been used where a genetic algorithm is being used as a feature selection approach with decision tree (DT) as a classification method. The dataset used in the experiments was UNSW-NB15 where the best subset of features has acquired an accuracy of 81.42% with a FAR of 6.39.

Apart from the traditional meta-heuristic feature selection approaches, Moustafa and Slay [12], have proposed an association rule mining technique for the feature selection/reduction in IoT intrusion detection. The proposed method has concentrated on central points of significant attributes that impact the detection of intrusion. The dataset used in the experiments was UNSW-NB15 where the average accuracy obtained by the proposed method was 83% with a FAR of 14.2.

Similarly, Mogal *et al.* [13] have utilized the Apriori algorithm in order to determine the most significant features within IoT intrusion detection. The proposed algorithm has conducted to rank the features based on its significance where the irrelevant ones will be dismissed. After that, two classifiers of naïve Bayes and logistic regression have been used to classify the data instances based on the selected features. The dataset of UNSW-NB15 has been used where the average accuracy obtained by the proposed method was 90% with a FAR of 10.5.

Another study that addressed the feature selection in IoT intrusion detection conducted by Papamartzivanos *et al.* [14] where a combination of genetic algorithm and decision tree has been proposed for this purpose. GA has been applied in order to make rule induction for the rules produced by the DT. As all the studies on IoT intrusion detection, the UNSW-NB15 dataset has been used in the experiments. Results of accuracy for the best subset of features showed 84.33% with a FAR of 8.9.

In the same regard, Hajisalem and Babaie [15] have proposed a combination of artificial bee colony (ABC) and artificial fish swarm (AFS) algorithms in order to accommodate a holistic feature selection task on IoT intrusion detection. The authors have taken the advantage of the two algorithms in order to find the best solution of features. Finally, an association rule classifier of classification and regression trees (CART) has been used to classify the intrusion based on the selected features. UNSW-NB15 dataset has been used in the experiments with an average accuracy of 85% with a FAR of 14.9.

On the other hand, some authors have used the feature selection approaches in order to improve the classifiers themselves in IoT intrusion detection. For example, Tama and Rhee [16] have proposed a grid search algorithm in order to search for the best parameters of classifiers. In fact, every classifier its own parameters, and sometimes, it is difficult to examine every parameter individually. Therefore, the proposed grid search has been used to identify the best parameters for three classifiers including neural network (NN), SVM and fuzzy classifier. Results showed that the proposed grid search has improved all the classifiers in which the combination of grid search and neural network has got the highest accuracy on UNSW-NB15 dataset where the average accuracy was 82.6% with a FAR of 16.2.

Ullah and Mahmoud [5] have proposed a linear method for the feature selection which is called recursive feature elimination (RFE) for the task of IoT intrusion detection. The proposed method will iteratively divide the feature space into much smaller subsets and recursively evaluate each feature. UNSW-NB15 dataset has been used in the experiment and the average accuracy obtained was 97% with a FAR of 7.8. Dwivedi *et al.* [17] proposed a combination of grasshopper optimization algorithm (GOA) and simulated annealing (SA) for IoT intrusion detection. An SVM classifier has been used as a fitness function. Result of accuracy using UNSW-NB15 was 98.85% with FAR of 0.084. Kasongo and Sun [18] proposed a feature selection technique based on XGBoost algorithm for IoT intrusion detection. The authors have examined different classifiers such as artificial neural network (ANN), DT, k-nearest neighbors algorithm (k-NN), SVM and DT. Using UNSW-NB15 dataset, DT obtained the highest accuracy of 90.85%.

Moualla *et al.* [19] proposed a feature selection technique based on extra trees classifier for IoT intrusion detection. Consequentially, the authors have utilized the extreme learning machine for the classification task. Using UNSW-NB15 dataset, the proposed method showed an accuracy of 98.43%. As noticed from the state of the art in feature selection for IoT intrusion detection, most of the studies have relying on traditional methods such as the rule-based and meta-heuristic. The drawback behind these methods lies on its inability to find optimal solution where the best subset of the features can be identified.

This study proposed the auto-encoder as a feature selection approach in IoT intrusion detection in order to improve the accuracy of classification by enhancing the feature learning. A benchmark dataset of IoT intrusions has been considered in the experiments. In addition, different normalization tasks have been conducted including irrelevant attribute removal and converting categorical attributes into numeric ones. After that, the proposed AE has been carried out where the connection features have been processed as an input and the same features have been processed as an output. Within the hidden layer, the reduced feature space or the selected features is being acquired. Based on such selected features, a simple neural network will be fed to classify the connection into its class label. Using various hidden size for the proposed AE architecture, results of accuracy and FAR showed superiority for the proposed AE over the traditional feature selection techniques where the best results have been obtained when the hidden size was 4 by achieving an accuracy of 99.97% with a 1.0 of FAR.

2. PROPOSED METHOD

The framework of the proposed method is composed of four stages as shown in Figure 1. The first stage tackles the details of the datasets where a full description is being provided for such dataset. Second stage aims to accommodate normalization tasks that are intended to remove unnecessary data and transform the data into much suitable formant. Third stage aims to apply the proposed AE for feature reduction. The final stage aims to apply the classification using NN classifier.

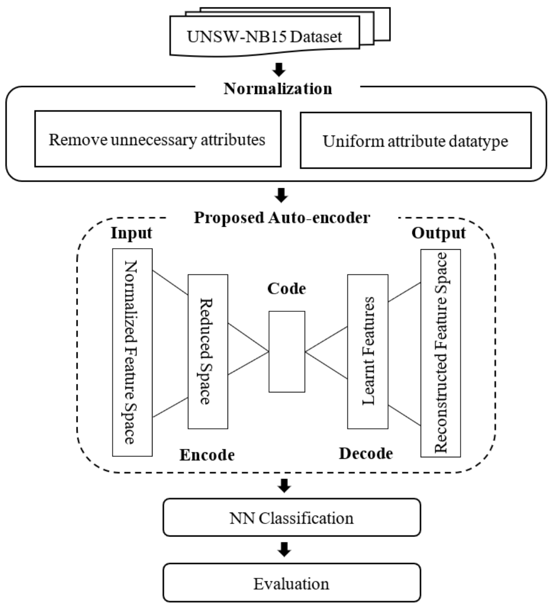

Figure 1. Framework of the proposed method

2.1. UNSW-NB15 dataset

The key challenge in any intrusion detection system lies on the availability of historical data that describes the intrusion features. Several datasets have been proposed for this purpose including KDD-CUP99 [20] and NSL-KDD [21]. Yet, the dramatic evolution on network technologies has facilitated toward the emergence of new attacks and threats. Therefore, both KDD-CUP99 and NSL-KDD would seem to be obsolete. Hence, this study will examine a recent dataset which known as UNSW-NB15 [22]. What can discriminate this dataset from the previous ones, is that UNSW-NB15 contains new threats and attacks like Shellcode which aims to take advantage of specific software in a certain network. Table 1 describes the connection class in UNSW-NB15.

This stage aims to utilize the benchmark dataset of UNSW-NB15. Unlike previous datasets of intrusion detection such as KDD-CUP99 and NSL-KDD where the simulation was conducted using traditional networks, UNSW-NB15 dataset is a simulation for both normal connections and intrusions that might target modern networks such as wireless sensor networks (WSN) and internet of things (IoT) [23]. The key distinguish between this dataset from the previous ones lies on the new threats and attacks that have been introduced such as Shellcode which aims to exploit specific software in a particular network. Table 1 shows the classes (i.e., attacks) that have been depicted in the dataset. While Table 2 depicts the features of UNSW-NB15.

Table 1. Classes and attacks of UNSW-NB15 dataset

No.	Attacks/Classes	Description
1.	Fuzzers	Targeting networks with data generated randomly
2.	Analysis	Associated with scanning and probing attacks
3.	Backdoors	Searching for vulnerabilities in network
4.	DoS	Ultimate exploitation of network resources
5.	Exploits	Searching for vulnerabilities in operating system
6.	Generic	Attacks associated with block ciphers and their keys
7.	Reconnaissance	Attacks that intended to gather information
8.	Shellcode	Taking advantage of specific software in a network
9.	Worms	Replicates itself to spread a network computer
10.	Normal	Legitimate connections

Table 2. Feature description of UNW-NB15

Feature Type	Quantity	Description
Flow Features	5	Features related to IP and Port details
Basic Features	13	Features related to protocol and service used by the connection
Content Features	8	Features related to size of transmitted and received packets
Time Features	9	Features related to connection time intervals
Connection Features	8	Features related to connection sessions
Total	43	

As represented in Table 2, UNSW-NB15 contains five types of features. The first type which is the flow features where the characteristics of IP and port are being examined. In addition, the second type is the basic features which are associated with protocol and service used by the connection. The third type is the content features which are associated with the size of transmitted and received packets. The fourth type is the time features which are associated with connection time intervals. Finally, the fifth type is connection features which are associated with connection session details. Table 3 shows the general statistics of UNSW-NB15 dataset.

Table 3. Statistics of UNSW-NB15

Attributes	Details
Total number of Connections	257,673
Training connections number	175,341
Testing connections numbers	82,332
Number of features	43
Number of attacks	9
Number of classes	10 (9 attacks with 1 Normal class)

2.2. Pre-processing

Unlike old datasets such as KDD-CUP99 or NSL-KDD where numerous noisy data is being located along with redundant records, the UNSW-NB15 dataset has been carefully designed. However, there are still some issues need to be tackled in such dataset. Therefore, this section aims to examine these issues. Table 4 shows a sample of connections from the dataset.

As shown in Table 4, multiple connections brought from the dataset. First observation would reveal that there are various features for each connection for example, the ID of such connection, its protocol, its service, and the duration of the connections. Lastly, there is a column for the class label where the connection is being categorized into ‘normal’ or any intrusion classes such as ‘exploits’ or ‘DoS’. Another attribute related to the class is located also which is the ‘Binary Class’. Such attribute contains only two values either ‘0’ for normal connection, or ‘1’ for the intrusion classes.

Table 4. Sample of connections from UNSW-NB15 dataset

Connection ID	Protocol	Service	Duration	….	Class	Class (Binary)
1	TCP	FTP	0.121478		Normal	0
2	TCP	HTTP	0.649902		Normal	0
3	UDP	HTTP	1.623129		Exploits	1
4	TCP	HTTP	1.681642		Normal	0
5	UDP	FTP	0.449454		DoS	1

Now, some attributes are not needed within the machine learning processing such as the ID in which the ID cannot indicate the status of any connection. In addition, the datatypes within the features are vary, this can hinder the machine learning from gaining a good training of such features. Hence, some normalization tasks are needed.

The first task aims to filter the attributes. There are some attributes that do not have any importance in terms of identifying the status of a connection. The first attribute is the ID in which the identification number of the connection would not have any significance in terms of determining the connection status. In addition, the 'Binary class' attribute is also unnecessary because it has only binary values (0 for normal connection and 1 for intrusion). In order to train the machine learning adequately, all the classes should be fed. Therefore, the aforementioned attributes must be removed. Table 5 depicts removing the unnecessary attributes.

Table 5. Removing unnecessary attributes

Protocol	Service	Duration	….	Class
TCP	FTP	0.121478		Normal
TCP	HTTP	0.649902		Normal
UDP	HTTP	1.623129		Exploits
TCP	HTTP	1.681642		Normal
UDP	FTP	0.449454		DoS

Note that, the number of features after removing the unnecessary attributes is 43 along with one attribute for the class label. The second task is the attribute transformation. In fact, the features contain variant datatypes where some attributes consist of numeric values (e.g., 0.12), while other attributes consist of nominal values (e.g., 'FTP' and 'TCP'). For adequate feature learning in MLT, it is important to transform the attributes. In this regard, the one-hot encoding approach is being used to turn the nominal values into numeric [24]. Table 6 shows an example of applying one-hot encoding on the data in Table 5.

Table 6. Example of applying one-hot encoding

Protocol_TCP	Protocol_UDP	Service_FTP	Service_HTTP	Duration	….	Class
1	0	1	0	0.121478		Normal
1	0	0	1	0.649902		Normal
0	1	0	1	1.623129		Exploits
1	0	0	1	1.681642		Normal
0	1	1	0	0.449454		DoS

As shown in Table 6, the one-hot encoding was intended to examine all the possible values in nominal attribute and then, turn these values into independent/additional attributes. For example, the 'Protocol' attribute was containing two values including 'TCP' and 'UDP' thus, the attribute has been divided into two attributes including 'Protocol_TCP' and 'Protocol_UDP'. Once the nominal attributes are being splitted based on its values, the matching value will be filled with '1' while the mis-match will be represented as '0'. In this way, the datatype of all attributes will be unified into numeric values. Note that, after splitting the nominal attributes, the number of features has been increased into 196 attributes.

2.3. Auto-encoder (AE)

Auto-Encoder is one of the Neural Network architectures that has been recently examined in terms of feature selection. In fact, any neural network would have three main layers including input, hidden and output. The input is the layer that takes the features of a connection, while the output layer would represent the class label of the connection. However, the hidden layer is the part of neural network where the features are being analyzed to find the deep relationship among them.

In contrast, the main aim behind AE is to learn a compressed and distributed representation of a given data [25]. In other words, AE aims to process a data as input and output the same data itself. Let assume a feature row from the sample in Figure 2, AE will process such feature row in which the input will be the features and the output are the same features. As shown in Figure 2, the input of AE are the features of a connection, while the output are the same values of the features. The first layer is also known as encoding within the AE where the data is being encoded until getting the coding and then, decoding the data.

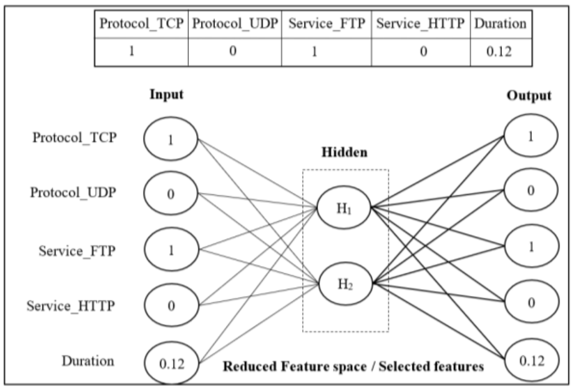

Figure 2. Feature selection via AE architecture

In order to understand such mechanism, the input weights will be initiated with random values, then the hidden will be computed. After that, the hidden weights will be initiated with random values to compute the output. After considering an activation function, the predicted output will be compared against the actual output to calculate the error. If there is error, the Backpropagation will be used to reduce the error rate until predicted output corresponds the actual output. Once the error is being minimized to zero where precited output is identical to the actual output, the hidden neuron values will be considered as the selected and reduced feature space as shown in Figure 2. Note that, the type of AE used in this study is symmetry where the encoder and decoder have the same number of neurons.

2.4. Neural network classification

After acquired the selected features by the proposed AE, a simple neural network will be used to classify the connections into intrusion and normal. As shown in Figure 3, the input of this neural network is the set of selected features produced by the proposed AE. As shown in Figure 3, F_1 and F_2 refers to the selected features produced by the AE architecture. In other words, these features are the value of the hidden nodes in AE after training it extensively.

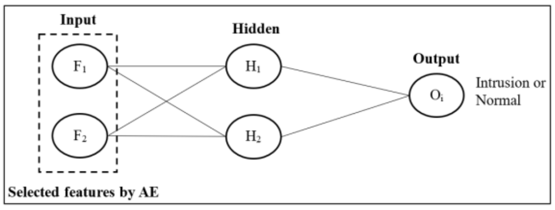

Figure 3. Classification using NN

3. RESULTS AND DISCUSSION

Since this study is utilizing neural network as a classification and auto-encoder as a feature selection. Which also a neural network thus, there are numerous parameters that should be determined. Following subsection will show the experiment setting alongside the results.

3.1. Experiment setting

Since both input and the output of AE are articulating the features of the connection thus, their length would be equivalent to the number of features. As mentioned earlier, the number of features within UNSW-NB15 dataset is 43. However, this study has applied the one-hot encoding to transform the attributes.

Hence, the number of attributes has been increased into 196. Therefore, both input and output layer have 196 neurons. On the other hand, the hidden layer used in the study was a single layer with different size of neurons. In fact, the hidden layer of AE is considered to be the reduced dimension of features therefore, it is necessary to examine different number of hidden neurons. Table 7 depicts the details of the three layers.

Table 7. AE architecture details

AE Layer	Size	Details
Input Layer	196	Length of connection features in UNSW-NB15 dataset
Hidden layer	4, 5, 10, 20, and 30	Experimenting
Output layer	196	Length of connection features in UNSW-NB15 dataset

As shown in Table 7, the hidden layer size has been set to different values for experimenting the most accurate results. The way of choosing these values lies on the fact that feature selection requires selecting lower dimension. According to Luo *et al.* [26], it is recommended for any feature selection task to examine number of features that lower fourth or fifth times the whole feature space. Since the feature space in our study is 196 thus, the experiments have begun with 30 number of features. This has been followed by a descending task in order to examine lower dimension of features including 20 and 10. Finally, for both 4 and 5 number of features, it was recommended by the study of Zhang *et al.* [27] that AE performs better with tiny numbers for the hidden layer size.

Unlike the traditional feature selection techniques where the output is a subset of features, the selected features in AE take different form. Similar to principle component analysis (PCA), AE will summarize the features with lower dimension using mathematical values [25]. These values are depicted in the hidden nodes of the AE and it can summarize or generalized the feature space.

Once the proposed AE produces the best features, another simple neural network will be used to classify the connection into intrusion and non-intrusion. Such neural network would have three layers including input, hidden and output. The input is considered to the selected features produced by AE, while the output is the class label whether the connection is intrusion or non-intrusion. The hidden layer here will represent the feature learning step where the features from the input layer is being learnt. Apparently, the size of input layer is simply equivalent to the size of hidden layer in AE, while the size of output layer is simply the number of class labels which is 10. However, for the hidden layer size, there are different approaches to get the size. This study has utilized the approach of averaging input and output which can be calculated as shown in (1) [28]. Table 8 represents the size of the three layers used by NN.

$$hidden\ size = \frac{input\ size + output\ size}{2} \tag{1}$$

Table 8. NN classification architecture details

NN Layer	Size	Details
Input Layer	4, 5, 10, 20, and 30	Hidden layer size of AE
Hidden layer	Input + output / 2	The mean average between input size and output size
Output layer	10	Number of class labels

After determining the length of each layer within AE and NN, it is important to mention the activation functions used by both architectures. The activation function used by AE was the rectified linear units (ReLU) which can be calculated as (2).

$$Relu(x) = \max(0, x) \begin{cases} if\ x < 0 & 0 \\ if\ x \geq 0 & x \end{cases} \tag{2}$$

According to Krizhevsky *et al.* [29], ReLu has remarkable performance with deep learning architecture. Therefore, it has been used with the proposed AE. On the other hand, the NN classification has utilized the Logistic Sigmoid which can be calculated as (3).

$$Sigmoid(x) = \frac{1}{1+e^{-x}} \tag{3}$$

The last parameter related to the neural network architecture is the number of epochs. In fact, epoch is the iteration required to accommodate error-tuning by changing the weights' values in order to reduce the error rate. Table 9 shows the number of epochs.

Table 9. Number of epochs for each neural network architecture

Architecture	Number of Epochs
AE	Experimenting (100 to 1200)
NN	1

In fact, determining the number of epochs is a challenging issue. It is a common agreement that selecting a greater number of epochs would lead to better result of accuracy. This is because the greater number of epochs would contribute toward minimizing the error rate which directly improve the accuracy. However, in order to get an efficient performance, it is better to get a high accuracy as much as possible using the minimal number of epochs.

In this regard, this study has utilized different values for the epoch number for the proposed AE (i.e., from 100 iterations to 1200). Yet, for the NN classification, only one epoch has been selected. The reason behind that is that the AE would have extensively trained on the data in order to produce the most accurate sub-set of features. Hence, there is no need for exhausted training conducted on the NN classification. On the other hand, the evaluation will be based on accuracy and false alarm rate (FAR) [30]. Accuracy refers to the correctly classified connection in respect to all the connections and it computed as (4):

$$Accuracy = \frac{TP+TN}{Total\ number\ of\ connections} \tag{4}$$

where TP and TN are the correct classified instances of intrusions and normal connections. While FAR refers to the ratio of the incorrectly classified connections in respect to all the connections which can be calculated as (5):

$$False\ Alarm\ Rate\ (FAR) = \frac{FP}{Total\ number\ of\ connections} \tag{5}$$

where FP is the number of incorrectly classified connections.

3.2. General investigation for feature number

Before applying the proposed AE, it is necessary to initiate a strategy for selecting the required number of features (i.e., Hidden size of AE). Since the length of input and output layers is 196 thus, the hidden size, or the reduced feature space, must be less than 196. Hence, there are numerous probabilities to select (i.e., from 1 to 198). In this regard, this section will accommodate a general investigation where three number of features are being used to highlight the performances. After reviewing these features numbers, the results of accuracy and FAR would contribute whether to increase or decrease the number of features. For this purpose, the number of features has been set to 10, 20 and 30.

As shown in Table 10, the accuracies for the three feature numbers have been increased as the number of epochs were increased. However, the highest accuracy depicted by 10 number of features where the accuracy was 81.45%. In terms of FAR as shown in Table 10, all the number of features showed similar rates of FAR in which the values have been decreased as the number of epochs increased where the minimal value of FAR was 1.0. This can demonstrate that the best number of features selected was 10 where it has the highest accuracy. Therefore, the best choice is to examine lower dimension of features. Hence, next section will depict such examination.

Table 10. Results of large number of features

Epoch No.	Features =30		Features = 20		Features = 10	
	Accuracy	FAR	Accuracy	FAR	Accuracy	FAR
100	0.0780	26.5264	0.1318	57.4522	0.1269	57.1282
200	0.3421	6.8484	0.1715	32.6364	0.1857	32.4523
300	0.5836	1.7681	0.2388	18.5394	0.2413	18.4349
400	0.5664	1.0	0.6516	10.5316	0.6820	10.5316
500	0.5471	1.0	**0.8037**	5.9826	0.7498	5.9826
600	**0.6782**	1.0	0.7913	3.3984	0.7493	5.9488
700	0.6697	1.0	0.7605	1.9305	**0.8145**	3.3984
800	0.6682	1.0	0.7520	1.0966	0.7027	1.9196
900	0.6631	1.0	0.7514	1.0	0.7032	1.0966
1000	0.6504	1.0	0.7520	1.0	0.7035	1.0
1100	0.6410	1.0	0.7525	1.0	0.7038	1.0
1200	0.6384	1.0	0.7529	1.0	0.7037	1.0

3.3. Specifying feature number

As shown in the previous section, the lowest number of features showed the best accuracy. Therefore, this section will examine multiple number of features that are less than 10. Table 11 shows the results. As shown in Table 11, when number of features was 4, the accuracy has reached to 99.97% compared to the maximum accuracy obtained when the number of features was 5 which is 90.33%. This can prove that 4 number of features is the most accurate reduction of the features produced by AE. Finally, for FAR, both number of features showed similar performance where the minimal value of FAR was 1.0.

Table 11. Results of lower feature number

Epoch No.	Features = 5		Features = 4	
	Accuracy	FAR	Accuracy	FAR
100	0.1433	57.4522	0.0103	57.4522
200	0.1844	32.6364	0.0122	32.6364
300	0.2286	18.5394	0.0292	18.5394
400	0.7205	10.5316	0.1347	10.5316
500	0.7977	5.9826	0.9834	5.9826
600	0.8014	3.3984	0.9972	3.3984
700	0.8717	1.9305	0.9990	1.9305
800	0.8944	1.0966	0.9993	1.0904
900	0.8954	1.0	0.9995	1.0
1000	0.8977	1.0	0.9996	1.0
1100	0.9002	1.0	0.9996	1.0
1200	**0.9033**	1.0	**0.9997**	1.0

Comparing the best results achieved by the proposed AE which was at 4 number of features against the related work is necessary. For instance, Gharaee and Hosseinvand [10] has obtained an accuracy of 93.25% with a FAR of 8.6 using the original GA with DT classifier. As well as, Papamartzivanos *et al.* [14] obtained an accuracy of 84.33% with a FAR of 8.9 using GA with DT. Additionally, Ullah and Mahmoud [5] have obtained an accuracy of 97% with a FAR of 7.8. Recently, Dwivedi *et al.* [17] obtained an accuracy of 98.85% with a FAR of 0.084, Kasongo and Sun [18] acquired an accuracy of 90.85%, and lastly, Moualla *et al.* [19] got an accuracy of 98.43%. Apparently, the proposed method has outperformed all the related work in terms of both accuracy and FAR. It is worth mentioning that most of the related work were based on either statistical or bio-inspired feature selection techniques. Since the proposed method is based on neural-network-based feature selection, it can be noticed that the deep learning architectures are outperforming the traditional feature selection techniques.

4. CONCLUSION

This study has successfully proposed and implemented AE for the task of feature selection in IoT intrusion detection. The novelty of this study is represented in examining a neural-network-based feature selection rather than the traditional bio-inspired feature selection technique. Results of applying AE showed a remarkable enhancement on the accuracy and FAR where the accuracy has been increased by approximately 3% compared to the state of the art, while FAR has been reduced by approximately 6.0. Experimenting different parameter settings for the AE such as the hidden layer, hidden neurons and activation function would reveal improvements in the future researches.

ACKNOWLEDGEMENTS

This study has been funded by the Universiti Kebangsaan Malaysia.

REFERENCES

[1] L. Da Xu, W. He, and S. Li, “Internet of things in industries: a survey,” *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2233–2243, Nov. 2014, doi: 10.1109/TII.2014.2300753.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, “Internet of things: a survey on enabling technologies, protocols, and applications,” *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015, doi: 10.1109/COMST.2015.2444095.

[3] B. Eskofier *et al.*, “An overview of smart shoes in the internet of health things: gait and mobility assessment in health promotion and disease monitoring,” *Applied Sciences*, vol. 7, no. 10, Sep. 2017, Art. no. 986, doi: 10.3390/app7100986.

[4] M. Sammour, B. Hussin, M. F. I. Othman, M. Doheir, B. AlShaikhdeeb, and M. Saad Talib, “DNS tunneling: a review on features,” *International Journal of Engineering & Technology*, vol. 7, no. 3.20, pp. 1–5, Sep. 2018, doi: 10.14419/ijet.v7i3.20.17266.

[5] I. Ullah and Q. H. Mahmoud, “A two-level hybrid model for anomalous activity detection in IoT networks,” in *2019 16th IEEE*

Annual Consumer Communications & Networking Conference (CCNC), Jan. 2019, pp. 1–6, doi: 10.1109/CCNC.2019.8651782.

[6] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 686–728, 2019, doi: 10.1109/COMST.2018.2847722.

[7] K. Farhana, M. Rahman, and M. T. Ahmed, "An intrusion detection system for packet and flow based networks using deep neural network approach," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 5, pp. 5514–5525, Oct. 2020, doi: 10.11591/ijece.v10i5.pp5514-5525.

[8] S. Laqtib, K. El Yassini, and M. L. Hasnaoui, "A technical review and comparative analysis of machine learning techniques for intrusion detection systems in MANET," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 3, pp. 2701–2709, Jun. 2020, doi: 10.11591/ijece.v10i3.pp2701-2709.

[9] S. Rajagopal, P. P. Kundapur, and H. Katiganere Siddaramappa, "A predictive model for network intrusion detection using stacking approach," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 3, pp. 2734–2741, Jun. 2020, doi: 10.11591/ijece.v10i3.pp2734-2741.

[10] H. Gharaee and H. Hosseinvand, "A new feature selection IDS based on genetic algorithm and SVM," in *2016 8th International Symposium on Telecommunications (IST)*, Sep. 2016, pp. 139–144, doi: 10.1109/ISTEL.2016.7881798.

[11] C. Khammassi and S. Krichen, "A GA-LR wrapper approach for feature selection in network intrusion detection," *Computers & Security*, vol. 70, pp. 255–277, Sep. 2017, doi: 10.1016/j.cose.2017.06.005.

[12] N. Moustafa and J. Slay, "A hybrid feature selection for network intrusion detection systems: Central points," *arXiv*, Jul. 2017, doi: 10.4225/75/57a84d4fbefbb.

[13] D. G. Mogal, S. R. Ghungrad, and B. B. Bhusare, "NIDS using machine learning classifiers on UNSW-NB15 and KDDCUP99 datasets," *IJARCCE*, vol. 6, no. 4, pp. 533–537, Apr. 2017, doi: 10.17148/IJARCCE.2017.64102.

[14] D. Papamartzivanos, F. Gómez Mármol, and G. Kambourakis, "Dendron : genetic trees driven rule induction for network intrusion detection systems," *Future Generation Computer Systems*, vol. 79, pp. 558–574, Feb. 2018, doi: 10.1016/j.future.2017.09.056.

[15] V. Hajisalem and S. Babaie, "A hybrid intrusion detection system based on ABC-AFS algorithm for misuse and anomaly detection," *Computer Networks*, vol. 136, pp. 37–50, May 2018, doi: 10.1016/j.comnet.2018.02.028.

[16] B. A. Tama and K.-H. Rhee, "An in-depth experimental study of anomaly detection using gradient boosted machine," *Neural Computing and Applications*, vol. 31, no. 4, pp. 955–965, Apr. 2019, doi: 10.1007/s00521-017-3128-z.

[17] S. Dwivedi, M. Vardhan, and S. Tripathi, "Incorporating evolutionary computation for securing wireless network against cyberthreats," *The Journal of Supercomputing*, vol. 76, no. 11, pp. 8691–8728, Nov. 2020, doi: 10.1007/s11227-020-03161-w.

[18] S. M. Kasongo and Y. Sun, "Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset," *Journal of Big Data*, vol. 7, no. 1, Dec. 2020, Art. no. 105, doi: 10.1186/s40537-020-00379-6.

[19] S. Moualla, K. Khorzom, and A. Jafar, "Improving the performance of machine learning-based network intrusion detection systems on the UNSW-NB15 dataset," *Computational Intelligence and Neuroscience*, vol. 2021, pp. 1–13, Jun. 2021, doi: 10.1155/2021/5557577.

[20] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, Jul. 2009, pp. 1–6, doi: 10.1109/CISDA.2009.5356528.

[21] S. Revathi and A. Malathi, "A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection," *International Journal of Engineering Research & Technology (IJERT)*, vol. 2, no. 12, pp. 1848–1853, 2013.

[22] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, Nov. 2015, pp. 1–6, doi: 10.1109/MilCIS.2015.7348942.

[23] S. Hanif, T. Ilyas, and M. Zeeshan, "Intrusion detection in IoT using artificial neural networks on UNSW-15 dataset," in *2019 IEEE 16th International Conference on Smart Cities: Improving Quality of Life Using ICT & IoT and AI (HONET-ICT)*, Oct. 2019, pp. 152–156, doi: 10.1109/HONET.2019.8908122.

[24] C. Seger, "An investigation of categorical variable encoding techniques in machine learning: binary versus one-hot and feature hashing," Royal Institute of Technology, Stockholm, 2018.

[25] S. N. Mighan and M. Kahani, "Deep learning based latent feature extraction for intrusion detection," in *Electrical Engineering (ICEE), Iranian Conference on*, May 2018, pp. 1511–1516, doi: 10.1109/ICEE.2018.8472418.

[26] M. Luo, F. Nie, X. Chang, Y. Yang, A. G. Hauptmann, and Q. Zheng, "Adaptive unsupervised feature selection with structure regularization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 4, pp. 944–956, Apr. 2018, doi: 10.1109/TNNLS.2017.2650978.

[27] C. Zhang, Y. Liu, and H. Fu, "AE2-nets: autoencoder in autoencoder networks," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2019, pp. 2572–2580, doi: 10.1109/CVPR.2019.00268.

[28] X. Liu and L. Xu, "The universal consistency of extreme learning machine," *Neurocomputing*, vol. 311, pp. 176–182, Oct. 2018, doi: 10.1016/j.neucom.2018.05.066.

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[30] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 184–208, 2016, doi: 10.1109/COMST.2015.2402161.

BIOGRAPHIES OF AUTHORS

Ahmed Fahad Alshudukhi received the B.Eng. degree in Software Development from University of South Australia, Australia, in 2016 and the M.S. degrees in Network Technology from Universiti Kebangsaan Malaysia, Malaysia, in 2020. His research interests include Computer Networking, Information Security, and Machine/Deep Learning Techniques. He can be contacted at email: alshudukhi@me.com

Saif Ahmed Jabbar receive the B.Sc. in Computer Science from Yarmouk University, and the M.Sc. in Computer Networking from Universiti Kebangsaan Malaysia. His research interests include Computer Networking, Information Security, and Machine/Deep Learning Techniques. He can be contacted at email: P100260@siswa.ukm.edu.my

Basel Alshaikhdeeb holds a PhD in Computer Science from Universiti Kebangsaan Malaysia. He is currently a Postdoctoral Researcher. His research interests include Natural Language Processing, Information Security, Image Processing, and Machine/Deep Learning Techniques. He can be contacted at email: shaikhdeeb@gmail.com