Wireless Local Area Network Intrusion Detection System Using Deep Belief Networks

Temesgen Mihiretu Abebe[1(✉)] and Menore Tekeba Mengistu[2]

[1] Faculty of Electrical and Computer Engineering, Bahir Dar Institute of Technology, Bahir Dar University, Bahir Dar, Ethiopia

[2] School of Electrical and Computer Engineering, Addis Ababa Institute of Technology, Addis Ababa University, Addis Ababa, Ethiopia

menore.tekeba@aait.edu.et

Abstract. In computer security Intrusion Detection System (IDS) is a mechanism of detecting an intruder in the system and notifying malicious activities for system administrators. The IDS researches on wireless Local Area Network (LAN) started recently. Until now there are some researches like publishing Aegean Wi-Fi Intrusion Dataset (AWID) dataset publically for the research community and evaluating the dataset using different machine learning algorithms. In this paper, we propose Deep Belief Network (DBN) to evaluate AWID dataset for intrusion detection analysis. Since AWID dataset contains different data types which are numeric, string, and hexadecimals; before training the model and evaluation of its performance the dataset is preprocessed and finally 102 attributes are used for system training. Also, two-stage feature selection is implemented to reduce the training cost and improve system performance. The first stage is removing duplicated attributes which reduced the dataset size to 68 attributes. The second stage is done by applying Weka implemented Information Gain Ratio (IGR). Using three thresholds three datasets are prepared with 41 attributes, 34 attributes, and 25 attributes. The system was able to achieve 98.55% accuracy with 102 attributes and it was able to improve this result to 98.97% with selected 34 attributes set evaluation.

Keywords: Wireless Intrusion Detection System (WIDS) · AWID dataset · Deep Belief Networks (DBN) · Feature selection

1 Introduction

In the history of computing, there are different security threats. Most of the security measure to protect computers and networks from threats can be classified as protection and detection mechanisms. Protection mechanisms are the way of protecting a system from threats by taking different actions even though it is not enough to secure the system. That is why detection and monitoring mechanisms are as necessary as protection mechanisms to alert the system administrators about security threats by tracking the system activities and network traffic. An IDS is a security system that monitors computer systems

M. A. Delele et al. (Eds.): ICAST 2020, LNICST 384, pp. 319–334, 2021.
https://doi.org/10.1007/978-3-030-80621-7_23

and network traffic and analyzes that traffic for possible hostile attacks originating from outside the organization and furthermore for system misuse or attacks beginning from inside the organization [1]. In general, IDS can be classified into two broad categories that are Host-based IDS (HIDS) and Network based IDS (NIDS). Host-based IDS look after the host system activities and in other way NIDS monitors the network traffics.

Based on the methods of intrusion detection mechanism they used to detect malicious traffic NIDS can be classified as a signature (misuse) based NIDS (SNIDS) and anomaly detection-based NIDS (ADNIDS) [2]. In the case SNIDS rules for the specific type of attack are pre-installed in the network environment, then the IDS based on the specified rule detects the signature of the traffic for the specified attack. In the case of ADNIDS whenever deviation from the normal traffic is observed it will be classified as intruders.

In a computer network, Wireless Fidelity (Wi-Fi) is a wireless connection type other than wired local area network. Wi-Fi is a recently growing technology for local area connections because it is easy and less expensive to implement than wired LAN. But it is vulnerable to various attacks and intruders because of its security vulnerable features [3–6]. So, studying IDS for wireless network independently from the wired network gives a better advantage for wireless network security.

Anomaly detection could be done using different mechanisms such as using different artificial intelligence (AI) algorithms, statistical methods, data-miming based, etc. and there are different researches and implementation using such methodologies. Artificial intelligence is one of the modern computing mechanism with a dream of achieving human level intelligence. Deep learning is one of the most recent studies of AI under the machine learning field to reach the promise of AI. Deep learning is the way of multi-layer network machine learning architecture. It includes different algorithms, among them DBN is recently proved to be very effective for a variety of machine learning problems [7].

There are different tool and researches regarding IDS based on different approach like rule-based, data mining techniques, machine learning algorithms, support vector machines, etc. But most of the researches are done for LAN [1, 8–10]. Since the vulnerability and attack type of wireless network is different from that of wired network [11–14], there should be an independent equivalent research practice for wireless network IDS. On wireless IDS after AWID dataset [3] there are some work using different machine learning techniques [3, 15–17]. The main problem is still the performance of the classification of attacks and normal traffic in not satisfactory and need to be improved as much as possible. So, since DBN is proven to be the best deep learning classifier in some recent works like [1], evaluating AWID dataset using DBN and also studying the effect of feature selection on the performance of the implemented system is a very important and challenging task as the wireless domain IDS.

The rest of the paper is organized as follows. In Sect. 2, related researches on wireless LAN IDS are discussed in detail. The system design and implementation is discussed in Sect. 3 and the dataset and dataset preprocessing techniques will discuss in Sect. 4. Then the result is discussed in Sect. 5. Finally, in the conclusion is goes in Sect. 6.

2 Literature Review on Related Works

Intrusion detection system in computer network was the active research field for a long period of time. Based on the nature of LAN IDS can be classified as wired LAN and wireless LAN. Since this research work focused on wireless IDS in this section literature on wireless IDS will discuss.

An intrusion detection system in a computer network was the active research field for a long period of time. Based on the nature of LAN IDS can be classified as wired LAN and wireless LAN. Since this research work focused on wireless IDS in this section literature on wireless IDS will discuss.

For wireless LAN IDS, the researchers in [3] publish AWID dataset by collecting from 802.11 Wi-Fi networks which contain normal and different attack traffics in 2015. Even though the dataset is collected from WEP protected network, the researcher states that the dataset can be used for WPA/WPA2 protected network IDS researches [3]. Their work can be seen in two perspectives. The first one is the publication of publicly available labeled dataset for the Wi-Fi network IDS research community. The second one is, they were able to evaluate the most known attack from 802.11 WEP protected Wi-Fi network standard using different machine learning techniques, which are Adaboost, J48, ZeroR, OneR, Random Tree, Random Forest, Nave Bayes and Hyperpipes [3]. For both feature sets J48 outperform the other classification algorithms and the reduced 20 attributes dataset performs better than the 156 featured datasets [3]. They also showed that feature selection and avoiding feature redundancy have great significance on the detection performance of IDSs [3].

The research in [16] works on the evaluation of AWID dataset using machine learning algorithms with Information Gain (GI) and Chi-Squared statistics (CH) filter-based feature selection techniques. Their main aim was to show the effect of feature selection on the performance of IDS attack detection [16]. Through their experimental result, they were able to show that feature reduction can improve detection accuracy and classification speed. But when they further reduce the number of features, the performance of the system decreases [16]. The machine learning technique they used for evaluation was OneR, J48, Random Forest, Random Tree, and Ada Boost. From their experimental result, Random Tree performs better for high-level class distribution and Random forest performs better for finer grain class distribution [16].

The research in [17] implements a majority voting technique for wireless IDS with a data mining technique for feature selection which was able to reduce the feature set to 20. The majority voting is a combination of Extra Trees of 20 trees, Random Forests of 20 trees, and the Bagging classifier of 10 Decision Trees as base estimator [17]. They implement Decision Trees, Extra Trees, and Random Forests for all feature sets and reduced feature set and they found that none of them performed better than J48 which is the best performer in [3]. Then they implement a bagging classifier for all feature sets and reduced feature set and it gives slightly better performance with better timing [17]. Finally, they implement their proposed system for all feature sets and reduced feature set and they found that it has better performance and the reduced dataset was able to achieve similar performance with minimal execution time [17].

The work in [15] focused on improving the detection level of impersonation attack which was insignificance in [1]. They use Artificial Neural Network (ANN) for feature

selection and Stacked Auto Encoder (SAE) for classification and they evaluate their proposed system using AWID-CLSR for both training and testing [15]. Also, they prepare the balanced version of the dataset in their preprocessing step for training purpose since the dataset contains a huge number of normal instances compared to attack instances, especially impersonation Attack [15].

3 System Design and Implementation

The proposed system architecture is shown in Fig. 1. It starts by selecting a specific dataset for training and testing from the AWID dataset collection. Then the next step is preprocessing of both the training and testing dataset which results a normalized version of both training and testing datasets. The preprocessed dataset is able to use for training and testing purposes on the system. Then feature selection process was applied to select the most discriminant features of the datasets. Finally, from the above steps, there are two datasets that are full featured and selected feature version dataset version.

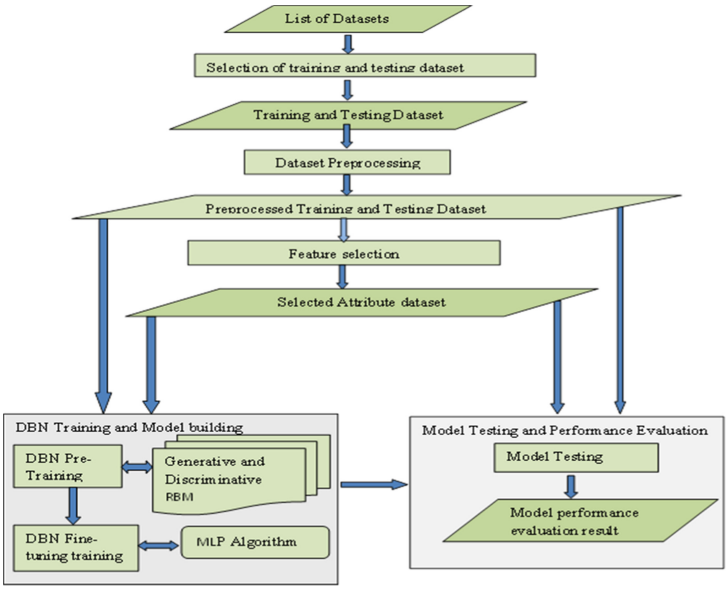

Fig. 1. System workflow

Then the next step is the training of DBN using each version of the datasets. The training process contains two steps. The first one is unsupervised pre-training using greedy layer-wise training through stacked Restricted Boltzmann Machine (RBM) and the second is fine-tuning the weight parameter of pre-trained DBN using back propagation neural network (BPNN) algorithm. Then the results of the above two training process will be trained DBN classifier which is the intended wireless IDS model.

3.1 Deep Belief Network Training

Deep Belief Network (DBN) is a probabilistic generative model made of multiple layers of latent stochastic variables [18]. It is a graphical model as depicted in Fig. 2, which learns to extract a deep hierarchical representation of training data. The top two layers give an undirected bipartite Boltzmann Machines (BMs) while the lower layers form a directed sigmoid belief network.

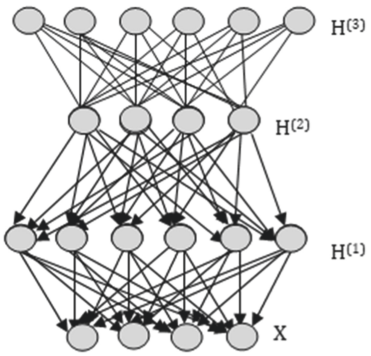

Fig. 2. Deep belief network

The training process consists of two steps [1, 18]. The first one is unsupervised pre training and the second one is supervised fine-tuning. The unsupervised pre-training method is used to initialize the hidden layer weights while building a deeper model. In the pre-training step of DBN each layer of the network will train using RBM in a greedy layer-wise manner. The trained RBM will be used to construct the pre-trained layer of DBN. Starting from the first up to the last layer they will modeled as generative RBM but the last layer of DBN with the classifier layer will be modeled as discriminative RBM [19]. The training process will continue from the input layer up to the last layer in a greedy layer-wise manner for unsupervised pre-training of DBN. After unsupervised pre-training, the next part of the system training is supervised fine-tuning. The purpose of fine-tuning is to more refine the hidden layer weight value using supervised algorithms. In this research BPNN [19–21] is used for fine tuning the pre-trained DBN.

3.2 Restricted Boltzmann Machine

An RBM is a bipartite graph in which visible units that represent observations are connected to binary stochastic hidden units using undirected weighted connections [22]. They are restricted in a sense that there are no visible-visible or hidden-hidden connections [23].

RBM has an efficient training procedure that makes it suitable for building blocks for DBN. The training set can be modeled using two layers RBM in which the visible unit connected to the input vector and the feature extractor corresponds to the hidden units

of RBM. The energy of joint configuration (v, h) of visible and hidden units expressed as follow in Eq. (1)

$$E(v,\ h,\ \theta) = \sum_{i=1}^{V} \sum_{j=1}^{H} W_{ij} v_i h_i - \sum_{i=1}^{V} b_i v_i - \sum_{j=1}^{H} a_j h_j \tag{1}$$

Where

- v_i and h_j are the state of visible unit i and hidden unit j.
- b_i and a_j are the bias of visible unit i and hidden unit j.
- W_{ij} is the weight on the connection between visible unit i and hidden unit j.

The joint probability of the visible and hidden unit of RBM is a function of the above energy function and given as in Eq. (2).

$$p(v, h) = \frac{1}{Z} e^{-E(v,h)} \tag{2}$$

Where:

- Z is the partition function and given by summing over all possible pairs of visible and hidden vectors as shown in Eq. (3).

$$Z = \sum_{v,\ h} e^{-E(v,h)} \tag{3}$$

The probability that the network assigns to a visible vector, v, is given by summing over all possible hidden vectors is given by as follows in (4).

$$\mathrm{p(v)} = \frac{1}{Z} \sum_{h} e^{-E(v,h)} \tag{4}$$

Gibbs Markov chain of the visible and hidden unit pair of variables used to estimate the gradient on the log-likelihood of RBM. It is the process of consecutive sampling h given v then v given h until the end of the chain. The chain starts from t = 0 meaning from the input vector then proceed to (v_t, h_t) where 't' is the number of sampling iteration. The derivative of the log-probability of a training vector with respect to a weight is given in Eq. (5).

$$\frac{\partial \log p(v)}{\partial w_{ij}} = \langle vihj \rangle data\ -\ \langle vihj \rangle model \tag{5}$$

The angle brackets are used to denote expectations under the distribution specified by the subscript that follows. This leads to a very simple learning rule for performing stochastic steepest ascent in the log probability of the training data:

$$\Delta w_{ij} = \varepsilon \left(\langle v_i h_j \rangle data\ -\ \langle v_i h_j \rangle model \right) \tag{6}$$

Where ε is learning rate of the weight update.

The above equation state that the weight updates is the difference of expectation of visible and hidden unit udder data and model distribution.

Since there is no direct connection between hidden units of RBM, getting the unbiased sample $\varepsilon\langle v_i h_j\rangle data$ is easy from the training vector. State of each hidden unit *hj* of the hidden layer set to 1 given the training vector can computed in the following probability Eq. (7).

$$p(h_j = 1|v) = \sigma\left(b_j + \sum_i v_i w_{ij}\right) \tag{7}$$

Also, to get the sample of each visible unit vi of the visible layer given the hidden layer can computed in the following probability Eq. (8).

$$p(v_i = 1|h) = \sigma\left(a_i + \sum_j v_i w_{ij}\right) \tag{8}$$

But getting an unbiased sample of $\langle v_i h_j\rangle model$ is not as simple as the data distribution. It could calculate using Constructive Divergence (CD) which is approximation of the gradient through alternative Gibbs sampling starting from visible layer units for some specified time. It is denoted as CD n where n denotes number of full alternative Gibbs sampling [24]. A single iteration of alternating Gibbs sampling consists of updating all of the hidden units in parallel using Eq. (7) followed by updating all of the visible units in parallel using Eq. (8).

The fast way of training as proposed by [22, 25] include setting the state of the visible units to the training vector, then sample the hidden units in parallel form the visible units using Eq. (7). The final step is the reconstruction process which is sampling the visible units in parallel from the hidden unit using Eq. (8). Then the above weight update equation can be simplified as shown in Eq. (9), (10), and (11).

$$\Delta w_{ij} = \varepsilon\left(\langle v_i h_j\rangle data - \langle v_i h_j\rangle model\right) \tag{9}$$

$$\Delta a = \varepsilon(\langle v_i\rangle data - \langle v_i\rangle model) \tag{10}$$

$$\Delta \mathrm{b} = \varepsilon\left(\langle h_j\rangle data - \langle h_j\rangle model\right) \tag{11}$$

4 Dataset

Aegean Wi-Fi Intrusion Dataset (AWID) is a collection of different wireless LAN datasets [3] for research purpose. Based on the attack class labeling there are two categories, which are high label (AWID-CLS) and finer grained (AWIDATK) [26]. The high level contains four classes namely normal, injection, impersonation and flooding in both training testing dataset. The finer grain version contains 16 classes based on the actual attack types. Then based on their size both high label and finer grained contain full dataset which is tagged as 'F' namely AWID-CLS-F and AWID-ATK-F and reduced dataset which is tagged as 'R' namely AWID-CLSR and AWID-ATK-R respectively. Each of them has separate training and testing datasets.

AWID dataset contains 156 attributes including the attack class [3]. They are composed of mainly from MAC layer information like source and destination address, initialization vector, ESSID, etc. The rest of attributes contain Radio tap information, general frame information, and frame number.

While the researchers of [3] preparing the dataset, they tested and study around 24 wireless LAN attacks based on their conceptual similarities. First, they study those attacks by grouping them using their attack purposes in to key retrieving attack, Key stream Retrieving Attacks, availability attacks, and Man-in-the-Middle Attacks. Key retrieving attacks tries to crack the Secret Key offline by monitoring specific packets.

It is passive type of attack; so, it is untraceable unless the attacker tries to execute the active version of the attack by injecting a large number of packets in the network. Key stream Retrieving Attacks focuses on retrieving key stream that can be used to create other attacks through injection. Availability attacks are commonly known as denial of service (DoS) attack since the attacks try to deny the system service by consuming or blocking the resource in the network. But since IDS system try to infer common patterns among the attack of the same class, they categorize the attacks according to the attack methodologies of execution. So, the new groups of attack are injection, flooding, impersonation and passive attack. All the attacks they studied are grouped under this list of attacks especially in the fine-gray (CLS) version of the dataset as shown in the Table 1.

Table 1. Wireless LAN Attacks and their classification

Attack class	Attack	Purpose	Target
Injection	ARP Injection, ChopChop, Fragmentation	Key Cracking	Network
Flooding	Deauthentication, Disassociation, Disassociation, Deauthentication broadcast, Disassociation broadcast, Block Acknowledge, Authentication Request, Fake Power Saving, CTS, RTS, Beacon, Probe Request, Probe Response	DoS	Client
Impersonation	Caffe Latte, Hirte, Honeypot, Evil Twin	Keystream, M-i-M	Network, client
Passive	FMS, Korek, PTW, Dictionary	Key Cracking	Network

4.1 Dataset Preprocessing

The purpose of the preprocessing step is to prepare the dataset in the way that comfortable for proposed DBN machine learning classification algorithms. The preprocessing mechanism has the following steps.

- In the dataset there are some values which are unassigned or missing values which are represented as question mark (?), and it is assigning '0' as value for those not available values.
- The SSID nominal value attack is encoded to numeric value based on their frequency in the specified attribute of the dataset. The other string valued attribute in the dataset is the attack classes. They are encoded as Normal to '1', Flooding to '2', Injection to '3' and Impersonation to '4' numeric value.
- The third step is type casting of hexadecimal values to their decimal equivalent. This involves converting hexadecimal values to their respective numeric value.
- The fourth step is scaling all the attributes between zero and one using Eq. (12).

$$Z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)} \tag{12}$$

Where:

- Z_i is scaled value.
- x_i is current attribute value from the dataset.
- min(x) is minimum value in the given attribute and
- max(x) maximum value in the given attribute
- Since DBN work on only real number valued attributes, the final step is removing NaN valued attributes from the dataset.

After the above preprocessing step out of the total attributes 52 of them have NaN. So, the resulting dataset of the preprocessing experiment has 103 attributes including the class labeling which is numeric valued and normalized between zero and one it is called version one dataset[1].

The next step in preprocessing is the attribute selection process which is implemented in two stages. The first one is removing the redundant attributes from the version one dataset by keeping single one of each attribute and its results version two of the dataset with 68 unique attributes without the class labeling. The second stage is feature selection using the entropy of each attribute for the class labeling. It is done using Weka implemented information gain feature selection algorithm. Based on their information gain value, attributes are ranked and by setting up some threshold values the most discriminant attributes are selected. To show the relationship between the number of attributes in the dataset and the performance of the system, three thresholds are used that result three different new versions of datasets as shown in Table 2.

The first version is using 0.015 threshold value and the resulting dataset has 41 attributes, the second version is using 0.05 and the resulting dataset has 34 attributes, and the last is results 25 attribute dataset version with 0.2 thresholds. So, the three new datasets are called as version three, four, and five respectively.

[1] Dataset version means nothing but it is the ways of identifying different datasets resulted from preprocessing and attribute selection procedures.

Table 2. Attributes information gain ratio value

No.	IG Threshold	Number of attributes
1	0.2	25
2	0.05	34
3	0.015	41

5 Experimental Result and Discussion

We use Dell OptiPlex 7020 desktop with the following specification and tools for system implementation and testing.

- Intel(R) Core (TM) i5-5500U CPU @ 3.70 GHz
- 8.0 GB DDR3 memory (RAM)
- Matlab R2016a
- Weka version 3.8

The first experiment is done using 102 attribute training and testing dataset for training and testing respectively. The system was able to achieve 98.55% accuracy with 100% of the training dataset. But accuracy is not the only way to measure the performance of a model especially in the case of unbalanced multiclass dataset as mentioned in [27–30] so let's discuss various performance metrics starting from the confusion matrix in Table 3.

Table 3. Confusion matrix for 100% 102 attributes training dataset

	Normal	Injection	Flooding	Impersonation
Normal	99.9717	0.0539	24.8240	30.7436
Injection	0.0034	99.9460	0	0
Flooding	0.0243	0	75.17599	0.0199
Impersonation	0.0006	0	0	69.2365

It shows that the classification accuracy of the normal and injection target classes is close to 100% which is 99.97% and 99.94% respectively, flooding is classified 75.17%, and impersonation is classified 69.23%. Based on the above confusion matrix other performance metrics are computed as shown in Table 4.

Precision or sensitivity is the number of true positive prediction of the specified class out of total positive prediction. As shown in the table based on their precision value the attack classes descending order is Impersonation, Injection, Normal, and Flooding. So, Impersonation class is the most precise than the other class and Flooding is less precise.

Table 4. Performance measure for 102 attribute model

Classes	Precision	Specificity	Recall	F-Measure
Normal	98.479	81.736	99.971	99.22
Injection	99.8925	99.996	99.946	99.919
Flooding	97.861	99.976	75.175	85.031
Impersonation	99.978	99.999	69.236	81.814

In another way, specificity is the measure of true negative prediction of the class out of the total negative prediction. Impersonation has higher specificity, Injection has the second, Flooding has the third, and Normal has the fourth and last specificity value.

Recall is the measure of the specified class true positive prediction out of that class actual true positive. From the four-class domain their recall value order is Normal, Injection, Flooding, and Impersonation. It means that in the case Normal and Injection classes, the model was able to predict truly above 99.9% to their respective class out of the given true class data and in the case of Flooding and Impersonation it is around 75.17% and 69.23% respectively.

F-Measure is the way of combining the class precision and recall using weighted harmonic mean. Injection has the highest F-measure value 99.91% and then normal with 99.22%, flooding with 85.03% and impersonation with 81.81% value follows.

5.1 Experiment of the Effect of Attribute Selection on the Performance of the System

As discussed in Sect. 4-A, there are four versions of datasets that are a result of the pre-processing and feature selection process. The system is trained independently and its performance is evaluated as shown in Fig. 3.

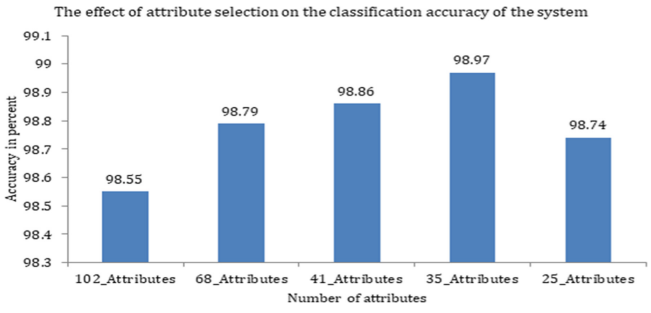

Fig. 3. Classification accuracy for different datasets

The classification accuracy of the system increases up to some point while we reduce the number of attributes by selecting the most discriminating attribute but it decreases when it decreases farther. The first accuracy improvement is from 98.55% to 98.79%

as a result of removing duplicated attributes from the dataset. Then next consecutive experiments are based on the feature selection using their IGR values. The system was able to achieve the best performance with 35 attribute set and those attributes are listed in Table 5.

Table 5. The most discriminative attributes based on classification accuracy

No	Attribute	No	Attribute
1	frame.time_epoch	18	wlan.duration
2	frame.time_delta	19	wlan.ra
3	frame.time_relative	20	wlan.da
4	frame.cap_len	21	wlan.ta
5	radiotap.flags.fcs	22	wlan.sa
6	radiotap.channel.freq	23	wlan.bssid
7	radiotap.channel.type.ofdm	24	wlan.frag
8	radiotap.channel.type.2ghz	25	wlan.seq
9	radiotap.antenna	26	wlan_mgt.fixed.timestamp
10	wlan.fc.type	27	wlan_mgt.fixed.beacon
11	wlan.fc.subtype	28	wlan_mgt.fixed.reason_code
12	wlan.fc.ds	29	wlan_mgt.ssid
13	wlan.fc.frag	30	wlan_mgt.tim.dtim_period
14	wlan.fc.retry	31	wlan.wep.iv
15	wlan.fc.pwrmgt	32	wlan.wep.icv
16	wlan.fc.moredata	33	wlan.qos.priority
17	wlan.fc.order	34	data.len

With feature reduction while the system achieves better classification accuracy decreases the training time from five hours and thirty-four minutes to three hours and thirty-five minutes.

As discussed above, accuracy is not the only measure for the performance of the system and it is clearly described that it is a good practice to make analysis on other performance metrics. The discussion is on the result of the 34-attribute set dataset. So as usual first let's see the confusion matrix of the classification result as shown in Table 6.

It shows that the system is able to classify the Normal class above 99.8%, Injection class 99.98%, Flooding class 93.49%, and Impersonation class 78.45% correctly. When compared to the previous 102 attribute result, in the case of Normal and Injection classes classification accuracy is almost similar but in the case of Flooding and Impersonation classes, there is a great improvement in their classification accuracy. Also, precision, specificity, recall, and F-measure performance metrics are computed for each class based on this confusion matrix as shown in Table 7.

Table 6. Confusion matrix for 34 attributes dataset

	Normal	Injection	Flooding	Impersonation
Normal	99.8046	0.0119	6.5042	21.4951
Injection	0.0038	99.988	0	0
Flooding	0.1650	0	93.4957	0.0548
Impersonation	0.0265	0	0	78.45012

Table 7. Performance measure of 34 attribute model

Classes	Precision	Specificity	Recall	F-Measure
Normal	98.479	81.736	99.971	99.22
Injection	99.8925	99.996	99.946	99.919
Flooding	97.861	99.976	75.175	85.031
Impersonation	99.978	99.999	69.236	81.814

As shown in the table Injection is the most precise one and followed by Impersonation, Normal, and flooding classes. In the case of specificity Injection is the most specific than the other class. The second specifically classified class is Impersonation followed by Flooding and the last specifically classified class is Normal class. Regarding recall Injection comes first and followed by Normal, Flooding and Impersonation. Also, in the case of F-measure value, which is the combination of precision and recall, Injection class has the highest value and followed by Normal, Flooding and Impersonation.

5.2 Result Comparison with Previous Works on Wireless Intrusion Detection Systems

The first research that evaluates AWID dataset using different machine learning algorithms is the research work by [3] and they were able to achieve 96.19% classification accuracy with J48 algorithm. They also select twenty attributes manually and they were able to improve the classification accuracy to 96.25%. The research work in [16] was able to achieve 95.12% using Random Tree with 41 attributes. Also, the research in [17] was able to achieve 96.32% in both full attribute set and 20 attribute set datasets. The research [15] was able to achieve 65% classification accuracy on impersonation attack with selected 35 attributes.

In this research, it was able to achieve 98.55% with 102 attributes and was able to improve this result to 98.97% with thirty-four selected attributes. When we see the individual class classification accuracy Normal and Injection was able to classify above 99.9% in both full feature and thirty-four attributes. In the case of Flooding, it was able to achieve 75.17% in 102 attribute dataset and it was able to improve to 93.45% using 34 attributes. Also, Impersonation is able to detect 69.23% while using 102 attribute and it was able to improve to 78.45% using 34 attributes. So, the above result discussion

clearly shows that the approach followed by this research is able to achieve better classification accuracy than similar previous works. The other strength of this research is, it uses only DBN to evaluate AWID dataset as a single efficient system. But in most of the researches, they use different machine learning algorithms and each algorithm has different classification accuracy in each class.

6 Conclusion

In this research wireless network IDS using DBN deep learning algorithm was proposed. For model building and performance evaluation, AWID dataset was used. The dataset was pre-reprocessed and feature selection was implemented to get the most discriminating features. Then a number of experiments have done on the implementation of the proposed system. In the first experiment, the system was able to achieve 98.55% while using 100% of the training dataset. The second Experiment is on the effect of the attribute selection on the performance of the system. Each of the four versions of datasets are evaluated and the result shows that the classification accuracy increases until 34 attributes and it falls down when the attribute set farther reduced to 25. So, using attribute selection it was able to improve the classification accuracy to 98.97%. Finally, when the proposed system is compared with similar previous works it was able to achieve better performance especially in the case of flooding and impersonation class detection. In the future, we plan to work on other versions of the dataset.

References

1. Alom, M.Z., Bontupalli, V., Taha, T.M.: Intrusion detection using deep belief networks. In: 2015 National Aerospace and Electronics Conference (NAECON), pp. 339–344. IEEE (2015)
2. Javaid, A., Niyaz, Q., Sun, W., Alam, M.: A deep learning approach for network intrusion detection system. In: Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS). ICST (Institute for Computer Sciences, Social-Informatics and . . , pp. 21–26 (2016)
3. Kolias, C., Kambourakis, G., Stavrou, A., Gritzalis, S.: Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset. IEEE Commun. Surv. Tutorials **18**(1), 184–208 (2015)
4. Waliullah, M., Gan, D.: Wireless lan security threats & vulnerabilities. Int. J. Adv. Comput. Sci. Appl. **5**(1) (2014)
5. Feng, P.: Wireless lan security issues and solutions. In: 2012 IEEE symposium on robotics and applications (ISRA), pp. 921–924. IEEE (2012)
6. Parte, S., Pandya, S.: A deep learning approach for network intrusion detection system. In: Wireless LAN: Security Issues and Solutions. International Conference on Innovation and Research in Technology for Sustainable Development 2012, pp. 104–107 (2012)
7. Salakhutdinov, R.: Learning deep generative models. Ann. Rev. Statist. Appl. **2**, 361–385 (2015)
8. Jha, J., Ragha, L.: Intrusion detection system using support vector machine. Int. J. Appl. Inf. Syst. **3**, 25–30 (2013)
9. Lakshmi, D.G.P.: Intrusion detection system using modified support vector machine. Network. Commun. Eng. **10**, 430–434 (2015)

10. Salama, M.A., Eid, H.F., Ramadan, R.A., Darwish, A., Hassanien, A.E.: Hybrid intelligent intrusion detection scheme. In: Gaspar-Cunha, A., Takahashi, R., Schaefer, G., Costa, L. (eds.) Soft computing in industrial applications, pp. 293–303. Springer Berlin Heidelberg, Berlin, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20505-7_26
11. Kaur, N., Monga, S.: Comparisons of wired and wireless networks: a review. Int. J. Adv. Eng. Technol. **5**(2), 34–35 (2014)
12. Sheldon, F.T., Weber, J.M., Yoo, S.-M., Pan, W.D.: The insecurity of wireless networks. IEEE Secur. Priv. **10**(4), 54–61 (2012)
13. Stimpson, T., Liu, L., Zhang, J., Hill, R., Liu, W., Zhan, Y.: Assessment of security and vulnerability of home wireless networks. In: 2012 9th International Conference on Fuzzy Systems and Knowledge Discovery, pp. 2133–2137. IEEE (2012)
14. Vibhuti, S.: Ieee 802.11 wep (wired equivalent privacy) Concepts And Vulnerability. San Jose State University, CA, USA, CS265 Spring (2005)
15. Aminanto, M.E., Kim, K.: Detecting impersonation attack in wifi networks using deep learning approach. In: Choi, D., Guilley, S. (eds.) Information Security Applications, pp. 136–147. Springer International Publishing, Cham (2016). https://doi.org/10.1007/978-3-319-56549-1_12
16. Thanthrige, U.S.K.P.M., Samarabandu, J., Wang, X.: Machine learning techniques for intrusion detection on public dataset. In: 2016 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), pp. 1–4. IEEE (2016)
17. Alotaibi, B., Elleithy, K.: A majority voting technique for wireless intrusion detection systems. In: 2016 IEEE Long Island Systems, Applications and Technology Conference (LISAT), pp. 1–6. IEEE (2016)
18. Hinton, G.E., Osindero, S., Teh, Y.-W.: A fast learning algorithm for deep belief nets. Neural Comput. **18**(7), 1527–1554 (2006)
19. Schalkoff, R.J.: Artificial Neural Networks. McGraw-Hill Higher Education (1997)
20. Rojas, R.: The backpropagation algorithm. In: Rojas, R. (ed.) Neural networks, pp. 149–182. Springer Berlin Heidelberg, Berlin, Heidelberg (1996). https://doi.org/10.1007/978-3-642-61068-4_7
21. Yam, Y., Chow, T.: Extended backpropagation algorithm. Electron. Lett. **29**(19), 1701–1702 (1993)
22. Hinton, G.E.: A practical guide to training restricted Boltzmann machines. In: Montavon, G., Orr, G.B., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade: Second Edition, pp. 599–619. Springer Berlin Heidelberg, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35289-8_32
23. Fischer, A., Igel, C.: An introduction to restricted boltzmann machines. In: Alvarez, L., Mejail, M., Gomez, L., Jacobo, J. (eds.) Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, pp. 14–36. Springer Berlin Heidelberg, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33275-3_2
24. Cote, M.-A., Larochelle, H.: An infinite restricted boltzmann machine. Neural Comput. **28**(7), 1265–1288 (2016)
25. Marlin, B., Swersky, K., Chen, B., Freitas, N.: Inductive principles for restricted boltzmann machine learning. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pp. 509–516 (2010)
26. Kolias, C., Kambourakis, G., Stavrou, A., Gritzalis, S.: AWID attributes, http://icsdweb.aegean.gr/awid/attributes.html. Accessed on 10 Sept 2017
27. Wang, S., Yao, X.: Relationships between diversity of classification ensembles and single-class performance measures. IEEE Trans. Knowl. Data Eng. **25**(1), 206–219 (2011)
28. Sokolova, M., Lapalme, G.: A systematic analysis of performance measures for classification tasks. Inf. Process. Manage. **45**(4), 427–437 (2009)

29. Nguyen, G.H., Bouzerdoum, A., Phung, S.L.: Learning pattern classification tasks with imbalanced data sets. In: Pattern Recognition. IntechOpen (2009)
30. Ferri, C., Hernandez-Orallo, J., Modroiu, R.: An experimental comparison of performance measures for classification. Pattern Recogn. Lett. **30**(1), 27–38 (2009)