An efficient deep learning model for intrusion classification and prediction in 5G and IoT networks

Shahadate Rezvy, Yuan Luo, Miltos Petridis, Aboubaker Lasebae
School of Science & Technology
Middlesex University London, UK
Email: {s.rezvy, y.luo, m.petridis, a.lasebae}@mdx.ac.uk

Tahmina Zebin
School of Computer Science & Engineering
University of Westminster, UK
Email: t.zebin@westminster.ac.uk

***Abstract*— A Network Intrusion Detection System is a critical component of every internet-connected system due to likely attacks from both external and internal sources. Such Security systems are used to detect network born attacks such as flooding, denial of service attacks, malware, and twin-evil intruders that are operating within the system. Neural networks have become an increasingly popular solution for network intrusion detection. Their capability of learning complex patterns and behaviors make them a suitable solution for differentiating between normal traffic and network attacks. In this paper, we have applied a deep autoencoded dense neural network algorithm for detecting intrusion or attacks in 5G and IoT network. We evaluated the algorithm with the benchmark Aegean Wi-Fi Intrusion dataset. Our results showed an excellent performance with an overall detection accuracy of 99.9% for Flooding, Impersonation and Injection type of attacks. We also presented a comparison with recent approaches used in literature which showed a substantial improvement in terms of accuracy and speed of detection with the proposed algorithm.**

***Index Terms*— computer network security, deep learning, intrusion detection system, autoencoder, dense neural network.**

I. Introduction

Cyber-attacks have increased by an alarming rate as the Internet of Things (IoT) are widely used now-a-days to provide e-commerce, to give online access to health-care, communication and billing systems. Experts predict that by 2020, wireless network traffic is anticipated to account for two-thirds of total Internet traffic to be generated by 50 billion Wi-Fi and cellular connected devices [1]. As technology becomes more and more integrated, security of these systems over wireless networks is becoming more important. Any hacks in banking systems, healthcare systems and many Internet of Things (IoT) devices could cause huge monetary losses every year and loss of services at crucial times. This drove to an increase in research for more secured online systems specifically in the intrusion detection systems [2]–[4]. With the majority of internet traffic occurring over wireless networks, and the domain is constantly updating with 5G and IoT technologies, there are likely many gaps in the security of these networks that can be exploited through intrusion attempts. Therefore, wireless intrusion detection systems are rapidly being developed in order to counter these potentially malicious behaviors. Recently, deep learning based methods have been successfully implemented in Network Intrusion Detection Systems (NIDS) applications. Deep learning can substitute for manually designed feature extraction to create more secure firewall or intrusion detector [5]. Modern intrusion detection systems that uses deep learning (hierarchical learning) approaches for learning traffic data representations is meant to effectively detect or prevent various kinds of intrusion in wired or wireless networks.

In this work, we have proposed a data mining based hybrid intrusion detection system for distinguishing normal and intrusive events from the AWID dataset [6]. We focused on exploring low latency models while maintaining high accuracy by proposing a deep auto-encoded dense neural network (DNN) framework for effective intrusion detection. The NIDS using deep learning did alleviate the need of feature selection or feature engineering during the detection process. In our design, the autoencoder facilitated an unsupervised pre-tarining on the data to provide compressed and less noisy representation of the input space, while the final dense neural network functioned as the supervised classifier for our experimental intrusion detection scenario.

The remainder of this paper is organized as follows. Section II introduces the background literature in the recent development in intrusion detection systems using deep learning techniques, we then provide details on our parameter settings for the implemented model in section III. The performance of the developed model for attack classification is evaluated in section IV. We also compared the results with some recent deep learning techniques appeared in the literature using the same dataset. Finally, the paper is concluded in section V along with ideas for future work.

II. Related Work

In recent years, machine learning has been widely applied to problems in detecting network attacks, particularly novel attacks. Given the landscape of the Internet, machine learning can be applied to handle the massive amounts of traffic to determine what is malicious or benign. NIDS are classifiers that differentiate unauthorized or anomalous traffic from authorized or normal traffic. Fig. 1 shows an illustration of the proposed components for the NIDS implementation. In recent years, neural networks have become an increasingly popular solution for network intrusion detection systems (NIDS). Their capability of learning complex patterns and behaviors make them a suitable solution for differentiating between normal traffic and network attacks [7]. One of the

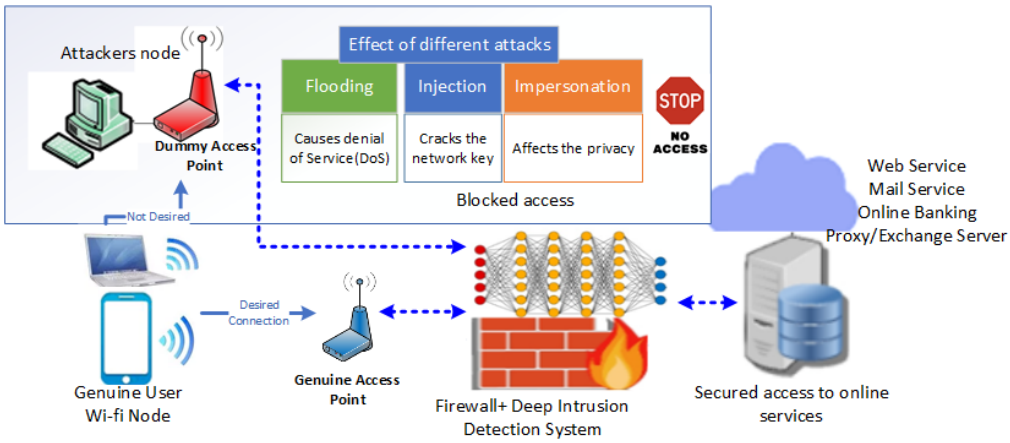

Fig. 1. Illustration of the proposed network intrusion detection system(NIDS)

TABLE I

BROAD ATTACK TYPES IN THE AWID DATASET [6], [10]

Broad attack type	Description of the attack	Sub-categories in AWID
Flooding	This category of attack is designed to cause an interruption or suspension of services of a specific host/server by flooding it with large quantities of useless traffic or external communication requests. When the Denial of Service (DoS) attack succeeds the server is not able to answer even to legitimate requests any more - this can be observed in numbers of ways: slow response of the server, slow network performance, unavailability of software or web page, inability to access data, website or other resources.	deauthentication, authentication request, amok, probe request, probe response, beacon, cts, rts, power saving, disassociation, power
Impersonation	Impersonation takes the form of device cloning, address spoofing, unauthorized access, rogue base station (or rogue access point) and replay. For example, in an evil twin setup, the client(s) unknowingly connect to them under the pretext that they are connected to a genuine access point. Once a client is connected, an attacker eavesdrops on its communication to hijack clients communication, re-direct clients to malicious websites, steal credentials of the clients connecting to it [11], [12].	evil-twin, caffe-latte, hirte
Injection	Attacker uses existing vulnerabilities in the applications to inject a code/string for execution that exceeds the allowed and expected input to the system requested, e.g. inject a client-side script onto the webpage or an SQL database	arp, fragmentation, chop chop

earliest work found in literature that used deep learning approach with Deep Belief Network (DBN) as a feature selector and Support Vector Machine (SVM) as a classifier were reported in [8].

As traditional machine learning methods depend heavily on feature engineering, extracting features is often time-consuming and complex. Thus, it is impractical to detect attacks with traditional machine learning methods in real-time applications [9].

We observed a use of Artificial neural networks (ANN) with enhanced resilient back-propagation for the design in ref [13]. In [14], the work used an unsupervised greedy learning algorithm to learn similarity representations over the nonlinear and high-dimensional data in KDD dataset.The results show that four-hidden-layer Restricted Boltzmann machines can produce the higher accuracy in comparison with SVM and ANN. In recent years, deep Neural Networks with three or more hidden layers support higher generalization capability in comparison to ANN [15]. The model is fed inputs, inputs get multiplied by weights and the passed into an activation function. The model uses backpropagation to adjust weights and increase accuracy. To be noted, most of the intrusion detection algorithms in the literature was found to be developed on on the NSL-KDD dataset, which is based on the KDD Cup 99 dataset [16].

In 2015, the Aegean Wi-Fi Intrusion Dataset (AWID) [6] was released as a comprehensive 802.11 network dataset that was derived from real Wi-Fi traffic traces. In the initial research with AWID, Kolias et al [10] applied several conventional supervised machine learning algorithms to perform the attack classification on the AWID dataset for accurate wireless network intrusion detection. They carried out manual feature selections and the top 20 features were chosen to train 8 classifiers. The overall accuracy of their classifiers ranges from 89.43% to 96.2% There are many proposed approaches in literature such as majority voting [4], multi-agent models [17], deep learning models [3], [18] where autoencoderbased models were used for detecting the attack categories in AWID dataset. However, Aminanto et al. [3] proposed an algorithm that can detect an impersonation attack by reducing the features dimensionalities and adopting stacked autoencoder at the final stage. However, they did not include flooding and injection attacks in consideration. The findings from our literature review have shown that despite the high detection accuracy being achieved, with most researchers still experimenting on combining various

algorithms (e.g. training, optimisation, activation and classification) and layering approaches to produce the most accurate and efficient solution for a specific dataset. A further accuracy-wise comparison of the methods in term of learning approach and accuracy is presented in Table II.

For this research, we focused on exploring low latency models while maintaining high accuracy by proposing a hybrid deep neural network that includes an unsupervised pre-training using autoencoders to make the model more adaptive to the changes in the network traffic. We then used a dedicated supervised dense neural network structure for the final classification. While designing, we made sure the memory or processing power to train and execute machine learning models are within the capability of the routers processing power. We hence believe the model and work presented in this paper will serve towards the real time and low latency implementation of the NIDS models.

III. Data Pre-processing and Implementation of the Model

In this section, we discuss on the technical details of our proposed deep learning based approach to increase the performance accuracy of the Aegean WiFi Intrusion Dataset (AWID). AWID is a publicly available collection of sets of data in an easily distributed format, which contain real traces of both normal and intrusive 802.11 WLAN[10]. Each record in the dataset is represented as a vector of 156 attributes, with the last attribute being class. Each record is composed mainly by MAC layer information. All attributes in the dataset have numeric or nominal values and the scales of the attributes on the dataset are heavily imbalanced. For example, a typical MAC address corresponds to an integer value of 82468889197, while a typical value of signal strength field is 33. Hence different types of encoding or normalization step would be necessary prior to applying any kind of machine learning algorithm to the dataset. There are two types of AWID dataset. The first type named CLS, has four target classes, whereas the second, named ATK, has 16 target classes. The 16 classes of the ATK dataset belong to the four attack categories in the CLS dataset.

For this research we have used the Reduced four class dataset (AWID-CLS-R-Trn, AWID-CLS-R-Tst) scenario with three attack classes (flooding, impersonation and injection) and one normal category for training and classification purposes. Further description on the features extracted in the dataset can be found in ref [6], [10]. The data contained in the AWID dataset are diverse in value, discrete, continuous, and symbolic, with a flexible value range. These data characteristics could make it difficult for the classifiers to learn the underlying patterns correctly.

A. Data Pre-processing

In training and the test dataset there is a total number of 2371,281 instances labelled data records. We used 20% of the entire dataset as test set, the remaining 80% of the data was used for training and validation purpose. As mentioned previously, the dataset contains different features with different value ranges. We conducted a detailed statistical analysis to monitor feature-wise values for minimum, maximum and standard deviation. We have replaced missing values with zeroes, dropped out the features with duplicate information and the columns with constants values as they contain no class-wise distinction. This analysis resulted in 36 unique features that was then fed to the autoencoded learning layer of our proposed model. We have also applied log encoding on the large numerical features such as source bytes, destination bytes and duration to avoid any kind of biasing on the training due to their large values. There were a number of features that needed conversion to 16-bit integer form before they could be fed to the model. As technology becomes more and more integrated, security of these systems over wireless networks is becoming more important. Over the years we have seen an increase in hacks in banking systems, healthcare systems and many Internet of Things (IoT) devices. We then performed a *standard scaler* function as a normalization operation on the feature vectors of the dataset. Finally, the output labels are *one hot encoded*. As a result, the total number of input dimension is 36 after performing the above-mentioned steps and output dimension is four (three attack classes and 1 normal class).

B. Proposed Model

Fig. 2 illustrates the work flow and the proposed deep model architecture for the intrusion detection system. Similar to most existing deep learning research, our proposed classification model was implemented using python keras library [19] with TensorFlow back end. All of our evaluations were performed on a Windows machine with an Intel Xeon 3.60GHz processor, 32 GB RAM and an NVIDIA GTX 1050 GPU.

We considered the following factors during our implementation of the IDS for the wireless realm: (a) The system should provide extremely low on false positive (FP) alerts due to the large volume of data, a FP rate of 1% may generate a great number of false alerts on daily basis. (b) It should be highly adaptive to drastic network behavioral changesdue to new events or natural changes in equipment, network behavior that once seemed normal may start looking suspicious. The system should also be able to detect novel attacksas wireless technologies change, new vulnerabilities should be added in the system on a regular basis.

We employed two main functional stages in our proposed model. An auto-encoder based unsupervised pre-training layer and a supervised dense neural network for classification of the attack types for the NIDS. We describe our intuition for using these components in the system development in the coming subsections.

1) Unsupervised pre-training with Autoencoder: An autoencoder is a type of artificial neural network used to learn efficient data representation in an unsupervised manner. In our proposed model, we have employed an autoencoder with an encoding and a decoding layer that has been trained to minimize the reconstruction error. This incorporated prior

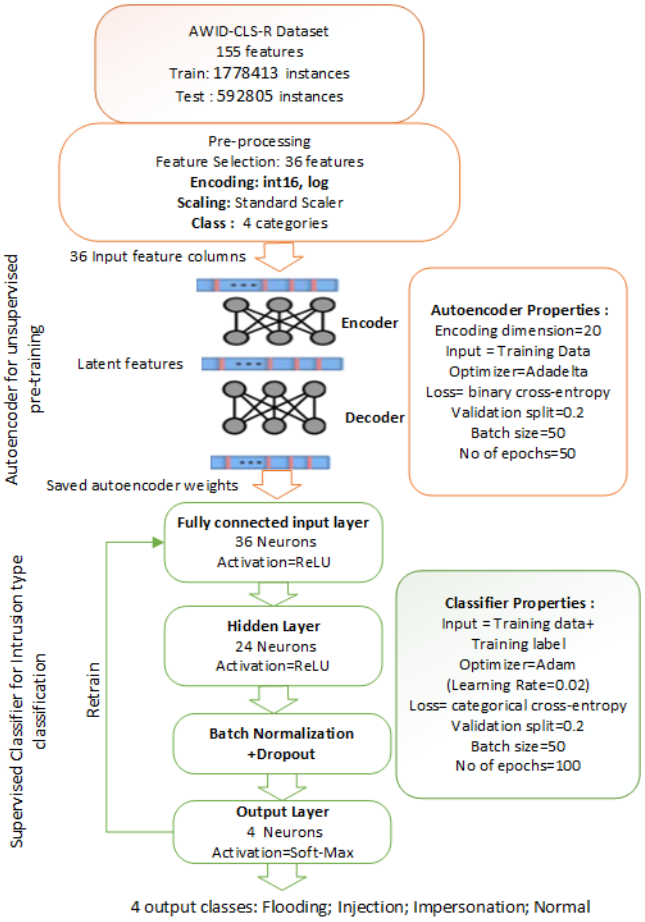

Fig. 2. Workflow and architecture of the proposed autoencoded dense neural network

knowledge from the training set to effectively learn from the data itself and provide good performance. Such pre-training allow both the data for the current task and for previous related tasks to self-organize the learning system to build the learning system in a data driven fashion. We have fed the autoencoder with the features from the training dataset without labels (unsupervised). A set of compressed and robust feature is built at the end of this step. The encoder part of the autoencoder aims to compress input data into a low-dimensional representation, and there is a decoder part that reconstructs input data based on the low-dimension representation generated by the encoder.

For a given training dataset $X = \{x_1, x_2, ..., x_m\}$ with m samples or instances, where x_n is an n-dimensional feature vector, the encoder maps the input vector x_n to a hidden representation vector h_n through a deterministic mapping f_θ as given in (2)

$$h_n = f_\theta(x_n) = \sigma(Wx_n + b) \tag{1}$$

where W is a $p \times p$, p is the number of hidden units, b is a bias vector, θ is the mapping parameter set $\theta = \{W, b\}$. σ is sigmoid activation function.

The decoder maps back the resulting hidden representation h_n to a reconstructed p-dimensional vector y_n in input space.

$$y_i = g_\theta(h_n) = \sigma(Wh_n + b) \tag{2}$$

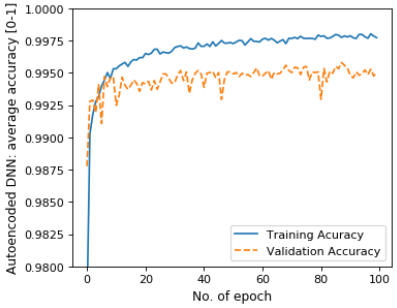

Fig. 3. Training and validation set accuracy over increasing number of epochs(training iterations)

The goal of training the autoencoder is to minimize the difference between input and output. Therefore, a error function is calculated by the following equation:

$$E(x, y) = \frac{1}{m} \left\| \sum_{i=1}^{m} (x_n - y_n) \right\|^2 \tag{3}$$

The main objective is to find the optimal parameters to minimize the difference between input and reconstructed output over the whole training set (m).

2) Supervised Classification with DNN: After the autoencoder layer, a three layer dense neural network is employed of a is trained by using the first auto-encoder,s output as inputs. This task sequence is retrained in a supervised manner with the class labels and the input feature given to the classifier. We have used a softmax activation layer as the output layer. The layer calculates the loss between the predicted values and the true values, and the weights in the network are adjusted according to the loss.

The simple *softmax* layer, which is placed at the final layer, can be defined as follows:

$$P(c|x) = \text{argmax}_{c \in C} \frac{\exp(x_{L-1} W_L + b_L)}{\sum_{k=1}^{N_C} \exp(x_{L-1} W_k)}, \tag{4}$$

where c is the number of classes, L is the last layer index, and N_C is the total number of class types including normal network connection and intrusion. After this stage, all layers are fine-tuned through back-propagation in a supervised way. In the test phase, the softmax layer outputs the probability of the predicted categories. The proposed algorithm is summarized here in Algorithm I.

3) Speeding up the training phase: Fig. 3 plots the network training process for 100 iterations. During training, we have used additional techniques such as dropout and batch normalization to avoid over fitting and also to speedup the training process. The proposed algorithm achieves approximately 99% accuracy for the training set in 20 iterations which is four times faster if no dropout and batch normalization was employed. We used a five-fold cross-validation using 20% of the training data as the validation data set. Potentially this allows a reduction in the training epochs required, and will be of vital importance for developing low-latency models and training future networks with bigger data sets.

Algorithm 1: Auto-encoded DNN training algorithm

Input: Taining dataset $X = \{x_1, x_2, ..., x_m\}$, Number of layers L

for $l \in [1, L]$ do;

Initialize $W_l = 0, W_l = 0, b_l = 0, b'_l = 0$;

Encoding layer;

Calculate encoding or hidden representation using equation(1);

$h_l = s(W_l x_{l1} + b_l)$;

Decoding layer;

while not loss==stopping criteria do;

Compute y_l using equation (2);

Compute the loss function: binary cross-entropy;

Update layer parameters $\theta = \{W, b\}$;

end while;

end for;

Classifier:Dense neural network, Soft-max activation at the output layer;

Initialize (W_{l+1}, b_{l+1}) at the supervised layer;

Calculate the labels for each sample x_n of the training dataset X;

Apply batch normalization and dropout for speeding up the calculation;

Perform back-propagation in a supervised manner to tune parameters of all layers, loss function categorical cross-entropy;

end;

Output: Class labels

IV. Model Evaluation

To prove the efficacy of our designed model, our aim is to show that the intrusion detection system will maximize attack prediction accuracy while minimizing any falsely categorized values.

A. Confusion Matrix

We presented the confusion matrix plot in Fig. 4, for our model when evaluated with the test data set. The rows correspond to the predicted class (Output Class) and the columns correspond to the true class (Target Class). If True Positive (T_P) is the number of attacks classified rightly as attack; True Negative (T_N) is the number of normal events rightly classified normal; False Positive (F_P) is the number of normal events misclassified as attacks and False Negative (F_N) is the number of attacks misclassified as normal, the diagonal cells in the confusion matrix correspond to observations that are correctly classified (T_P and T_N's). The off-diagonal cells correspond to incorrectly classified observations (F_P and F_N's). Both the number of observations and the percentage of the total number of observations are shown in each cell.

The column on the far right of the plot shows the percentages of all the examples predicted to belong to each class that are correctly and incorrectly classified. These values are often called the precision and false discovery rate respectively. To be noted, we have utilized builtin Matlab function *plotconfusion(true class, predicted class)* for generating Fig. 4. Both true class and predicted class variables were one-hot encoded and imported from the python interface as .mat file for plotting purposes.

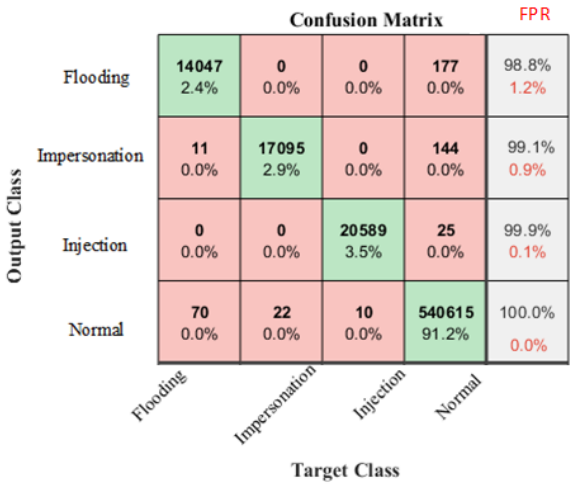

Fig. 4. Confusion matrix of the test dataset

We can also define accuracy, recall, precision and F1 values of a model using the following equations:

- Accuracy: It is an indicator of the total number of correct predictions provided by the model and defined as follows:

$$\text{Accuracy} = \frac{T_P + T_N}{T_P + T_N + F_P + F_N}. \tag{5}$$

- Recall, precision and F measure: Three of the most commonly used performance measures with F measure being the harmonic mean of recall and precision measures are defined as follows:

$$\text{Recall or True positive rate} = \frac{T_P}{T_P + F_N}. \tag{6}$$

$$\text{Precision} = \frac{T_P}{T_P + F_P}. \tag{7}$$

$$\text{F Measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision+Recall}} \tag{8}$$

We obtained a true positive rate (TPR) of above 99% for all the attack categories and normal connection. For flooding attacks, there were only 1.2% (177 wrong predictions out of 14k instances) false predictions made by the model. For the injection and impersonation type attacks, its even lower as 0.1% and 0.9% respectively (also presented in the rightmost column in Fig. 4). For the normal connection classification there were only 92 instances out of 540k test cases that was misclassified by the model.

B. Comparison with other approaches

There were several neural network and deep learning based methods presented in [3], [11], [17], [20], with variable accuracy for the under-represented attack classes. Our results showed an excellent performance with an overall detection accuracy of 99.9%. Table II summarizes the class-wise true positive rate (TPR) and overall accuracy of our proposed model with some concurrent deep learning methods, our

TABLE II

COMPARISON OF THE PROPOSED METHOD WITH OTHER MACHINE LEARNING TECHNIQUES FOR AWID FOUR-CLASS INTRUSION DETECTION

Learning method	Dataset	Flooding	Injection	Impersonation	Normal	Overall Accuracy (%)
J48 (20 features) [10]	AWID-CLS-R	99.83	100	70.55	96.14	96.2
Random Forest(20 features) [10]	AWID-CLS-R	99.97	99.42	99.92	95.44	95.6
Majority Voting (154 features) [4]	AWID-CLS-R	100	100	100	96.15	96.32
Multi-Layer Perceptron (32/7/5 features) [18]	AWID-ATK-R	-	-	-	-	96.21
3-hidden-layer deep architecture[20]	AWID-ATK-R	34.15	82.58	0	99.93	95.02
Neural Network (154/6 features)[17]	AWID-CLS-R	-	-	-	-	99.3
Stacked Autoencoder (35 features)[3]	AWID-CLS-R	-	-	-	-	99.88
Proposed Autoencoded DNN (36 features)	AWID-CLS-R	99.42	99.87	99.9	99.93	99.9

work outperforms previous related work in terms of number efficient selection of features and accuracy.

V. CONCLUSIONS

Cyber threats have become a prime concern for information security. NIDS is one of the security mechanisms used to guard these applications against attacks. In this research, we have applied a deep network intrusion detection model and evaluated the algorithm with the benchmark AIWD dataset. Currently, the algorithm is trained offline on high performance computer. Our results showed an excellent performance with an overall detection accuracy of 99.8% for Flooding or DoS, Injection and Impersonation type of attacks. We also presented a comparison with recent approaches used in literature which showed a substantial improvement in terms of accuracy and latency with proposed autoencoded DNN. In future, we will provide extensions or modifications of the proposed algorithm for larger attack types , mobile and IoT security platforms as suggested in ref [18] using intelligent agents such as soft computing and advanced unsupervised clustering algorithms. Because of the availability of deep learning libraries in python such as Keras and TensorFlow Lite, on-device machine learning inference is now possible with low latency. Hence, future models will cover a broader range of attacks, respond in real time and update itself over time. Future extension of this work will be to improve the detection accuracy and to reduce the rate of false negatives and false positives in attack detection to improve the systems performance.

REFERENCES

[1] N. Shone, T. N. Ngoc, V. D. Phai, *et al.*, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, Feb. 2018.

[2] C. Kolias, V. Kolias, and G. Kambourakis, "Termid: A distributed swarm intelligence-based approach for wireless intrusion detection," *International Journal of Information Security*, vol. 16, no. 4, pp. 401–416, 2017.

[3] M. E. Aminanto, R. Choi, H. C. Tanuwidjaja, *et al.*, "Deep abstraction and weighted feature selection for wi-fi impersonation detection," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 3, pp. 621–636, 2018.

[4] B. Alotaibi and K. Elleithy, "A majority voting technique for wireless intrusion detection systems," in *Systems, Applications and Technology Conference (LISAT), 2016 IEEE Long Island*, IEEE, 2016, pp. 1–6.

[5] B. Lee, S. Amaresh, C. Green, *et al.*, "Comparative study of deep learning models for network intrusion detection," *SMU Data Science Review, Article 8*, vol. 1, no. 1, 2018.

[6] *Awid dataset - wireless security datasets project*, 2014. [Online]. Available: `http://icsdweb.aegean.gr/awid/`.

[7] Y. Mirsky, T. Doitshman, Y. Elovici, *et al.*, "Kitsune: An ensemble of autoencoders for online network intrusion detection," *CoRR*, 2018.

[8] M. A. Salama, H. F. Eid, R. A. Ramadan, *et al.*, "Hybrid intelligent intrusion detection scheme," in *Soft Computing in Industrial Applications*, A. Gaspar-Cunha, R. Takahashi, G. Schaefer, *et al.*, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 293–303.

[9] H. Liu, B. Lang, M. Liu, *et al.*, "Cnn and rnn based payload classification methods for attack detection," *Knowledge-Based Systems*, 2018.

[10] C. Kolias, G. Kambourakis, A. Stavrou, *et al.*, "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.

[11] M. E. Aminanto and K. Kim, "Improving detection of wi-fi impersonation by fully unsupervised deep learning," in *International Workshop on Information Security Applications*, Springer, 2017, pp. 212–223.

[12] M. Agarwal, S. Biswas, and S. Nandi, "An efficient scheme to detect evil twin rogue access point attack in 802.11 wi-fi networks," *International Journal of Wireless Information Networks*, vol. 25, no. 2, pp. 130–145, Jun. 2018.

[13] R. S. Naoum, N. A. Abid, and Z. N. Al-Sultani, "An enhanced resilient backpropagation artificial neural network for intrusion detection system," in *International Journal of Computer Science and Network Security*, vol. 12, Mar. 2012.

[14] N. Gao, L. Gao, Q. Gao, *et al.*, "An intrusion detection model based on deep belief networks," in *2014 Second International Conference on Advanced Cloud and Big Data*, Nov. 2014, pp. 247–252.

[15] O. Kaynar, A. G. Yüksek, Y. Görmez, *et al.*, "Intrusion detection with autoencoder based deep learning machine," in *25th Signal Processing and Communications Applications Conference (SIU)*, May 2017, pp. 1–4.

[16] *Nsl-kdd dataset*. [Online]. Available: `http://www.unb.ca/cic/datasets/nsl.html`.

[17] D. Kaleem and K. Ferens, "A cognitive multi-agent model to detect malicious threats," in *International Conference on Advances in Applied Cognitive Computing, CSREA*, 2016, pp. 58–63.

[18] R. Abdulhammed, H. Musafer, A. Alessa, *et al.*, "Machine learning approaches for flow-based intrusion detection systems," 2018.

[19] F. Chollet. (2013). Keras: The python deep learning library, [Online]. Available: `https://keras.io/`.

[20] V. L. Thing, "Ieee 802.11 network anomaly detection and attack classification: A deep learning approach," in *Wireless Communications and Networking Conference (WCNC), 2017 IEEE*, IEEE, 2017, pp. 1–6.