IEEE 802.11 Network Anomaly Detection and Attack Classification: A Deep Learning Approach

Vrizlynn L. L. Thing
Cyber Security Cluster
Institute for Infocomm Research, A*STAR, Singapore
vriz@i2r.a-star.edu.sg

***Abstract*—Despite the significant advancement in wireless technologies over the years, IEEE 802.11 still emerges as the de-facto standard to achieve the required short to medium range wireless device connectivity in anywhere from offices to homes. With it being ranked the highest among all deployed wireless technologies in terms of market adoption, vulnerability exploitation and attacks targeting it have also been commonly observed. IEEE 802.11 security has thus become a key concern over the years. In this paper, we analysed the threats and attacks targeting the IEEE 802.11 network and also identified the challenges of achieving accurate threat and attack classification, especially in situations where the attacks are novel and have never been encountered by the detection and classification system before. We then proposed a solution based on anomaly detection and classification using a deep learning approach. The deep learning approach self-learns the features necessary to detect network anomalies and is able to perform attack classification accurately. In our experiments, we considered the classification as a multi-class problem (that is, legitimate traffic, flooding type attacks, injection type attacks and impersonation type attacks), and achieved an overall accuracy of 98.6688% in classifying the attacks through the proposed solution.**

***Index Terms*—IoT security, anomaly detection, attack classification, deep learning, IEEE 802.11 attacks.**

I. Introduction

With the active adoption of infocomm technologies to better support the inter-connectivity of things, we have seen significant effort put forth that results in the enhancement of efficiency in the work environment and standard of living worldwide, in both smart cities or from a wider perspective, a smart nation [1]. An important building block of a smart city is the smart home, which is essentially where people spend a major part of their lives. A key enabler which supports the wireless device connectivity in smart homes is the IEEE 802.11 standards. The IEEE 802.11 standards specify the media access control (MAC) and physical layer (PHY) for the implementation of wireless local area networks (WLANs), and provide the basis for the more commonly known technology, the Wi-Fi protocol. The IEEE 802.11 is the de-facto standard to achieve the required short to medium range wireless device connectivity in smart homes, and is ranked the most popular and has the highest market adoption as the wireless technology deployed [2].

In contrast to the wired network technologies where a device needs to be physically connected to the network to obtain network resource accessibility, a device requiring connectivity to the WLAN only needs to be within the signal range of the wireless access point. With the removal of the need for wires to be deployed and the requirement for physical access, many security issues have surfaced and security considerations become essential for the wireless network protocols.

Over the years, high profile security vulnerability exploitation and attacks have been observed across different versions of the Wi-Fi protocol. Examples of vulnerabilities and attacks that plague the protocol include those targeting the Wired Equivalent Protection (WEP) [3]–[6] and the Wi-Fi Protected Access (WPA) and WPA2 [7], [8]. As such, even though the Wi-Fi protocol provides good mobility support and connectivity compatibility due to the adoption by many smart devices, especially smart phones, tablets and laptops, the risks that accompany the wide adoption of this protocol has also been brought into question.

In light of these potential threats, attack detection has an important role to play as it can be seen that protection mechanisms are simply not adequate in defending against attacks in the IEEE 802.11 networks. In addition, it is not sufficient to only view the attack detection as a binary class problem (that is, a traffic flow constitutes an attack or not). To facilitate the attack analysis, response, mitigation and recovery stages, and potentially also attack attribution, it is also necessary to classify the attacks at a finer granularity in an accurate manner, so as to enable an easier and more streamlined effort in the subsequent stages of work for the security analysts.

In this paper, we analyse the threats and attacks targeting the IEEE 802.11 network by utilizing a publicly available dataset. The data was collected by the authors in [9] from a lab which was set up with various smart devices (e.g. smart TV, smart phones) to realistically emulate a typical SOHO infrastructure. Various attacks using different set of tools were carried out in the lab, and both attack and legitimate WI-FI signals' measurements were collected. The attacks fall into the categories of flooding, injection and impersonation. We then propose a deep learning approach which self-learns the features necessary to detect network anomalies and perform attack classification accurately. To the best of our knowledge, this is the first work that proposes a deep learning approach to perform IEEE 802.11 network anomaly detection and attack classification. In our experiments, we considered the

978-1-5090-4183-1/17/$31.00 ©2017 IEEE

classification task as a multi-class (that is, legitimate traffic, flooding type attacks, injection type attacks and impersonation type attacks) classification problem, and achieved an overall accuracy of 98.6688% in classifying the attacks accordingly.

The rest of the paper is organised as follow. A discussion on existing work is provided in Section II. Section III provides readers with the challenges of attack detection and classification, as well as the background and description of the dataset used in this work. We describe the proposed deep learning approach in Section IV, and provide the experimental results and analysis in Section V. Finally, we conclude the paper in Section VI.

II. Existing Works

There are several works [9]–[13] that explore conventional machine learning for security anomaly detection and attack classification. However, existing works on deep learning based approaches are still quite limited.

In a recent work [9], the authors applied several conventional supervised machine learning algorithms to perform the attack classification on the same dataset used in this paper. They mentioned the importance of feature and attribute selection to perform accurate wireless network intrusion detection [14], and carried out manual feature selections. The top 20 features were chosen to train 8 classifiers. The overall accuracy of their classifiers ranges from 89.43% to 96.2%. However, manual feature selection can be a very tedious and time-consuming process. In our work, we adopt a self-learning approach which is an intrinsic characteristic of deep learning to fulfil this challenging task of wireless network anomaly detection and attack classification, to attain a higher overall classification accuracy.

In [15], the authors proposed utilizing deep learning to perform intrusion detection on the NSL-KDD dataset. The NSL-KDD dataset is based on the KDD Cup 99 dataset [16], which consists of network traffic captured by the 1998 DARPA Intrusion Detection Evaluation Program. The NSL-KDD dataset is an improved version where the redundant records were removed, and the data is partitioned such that the records are grouped into various difficulty levels based on the number of learning algorithms developed thus far, that can correctly classify them. The author's proposed approach correctly classifies the test data with an accuracy of 88.39% and 79.1% for the 2-class (attack vs. normal) and 5-class problem (normal vs. 4 classes of attacks), respectively. However, the dataset is very dated, and the authors only considered the sparse auto-encoder with the classical sigmoid activation function in their work, which can be limited in performance. In addition, the data consisted of raw TCP dumps, which is different from the data we are exploring in this paper.

III. Background and Dataset

A. Challenges in Attack Detection and Classification

Attack detection and classification techniques can be broadly categorized into signature based and anomaly based approaches. Signature based solutions are widely used due to their high detection accuracy and low false positive rates. They create specific rules based on prior observations from known attacks and perform matching to determine if a traffic flow constitutes an attack or not. However, they suffer from the lack of the ability to detect new attacks that they have never encountered before. Anomaly based detection, on the other hand, detects attacks based on traffic flows that deviate from the normal profile. Thus, anomaly based detection supports the detection of novel attacks.

Machine learning has been widely utilized in security anomaly detection in recent years [9]–[13]. A severe limitation is the need to perform tedious and time consuming feature engineering to achieve a good accuracy in attack detection and classification. More often than not, the generated models are not optimal in achieving a high accuracy to address multi-class classification problems. Therefore, in this paper, we propose a deep learning approach that can self-learn features that are necessary in the accurate detection and classification of network anomalies.

B. Description of Dataset

In [9], the authors constructed a lab to emulate a SOHO infrastructure. The lab consisted of mobile and stationary devices, which include a desktop machine, two laptops, two smart phones, a tablet and a smart TV. These devices were used as legitimate clients of the network. The smart phones were used to display high mobility patterns (that is, their locations were changed frequently and made to join/leave the network throughout the duration of the experiments), while the laptops were semi-static (that is, they rarely change in locations). Different services running on the devices were producing legitimate traffic such as through web browsing and VoIP.

The network coverage was provided by an AP and protected by WEP encryption, supporting up to a transfer rate of 54 Mbps. The attacks were launched from an attack node, which is a laptop running Kali Linux 1.0.6 64-bit OS. The attack node was located outside the perimeter of the lab and "mobile" (that is, its MAC address was frequently changed while the various attacks were carried out). During the experiment duration, the attack node launched 15 unique attacks and different variations of them to achieve specific tasks, such as cracking the network key. The 15 attack types are the Fragmentation, ChopChop and ARP Injection attacks; Deauthenticaton, Authorization Request, Beacon, CTS, RTS, Disassociation, Fake Power Saving, Probe Request and Probe Response Flooding attacks; and the Evil Twin, Hirte and Caffe Latte Impersonation attacks.

Several standard wireless penetration testing tools were utilized in the experiments. They are the Aircrack suite [17], the MDK3 tool [18] and the Metasploit framework [19]. The Probe request flooding attack was launched using the File2air tool [20] while the fake power saving and disassociation attacks were launched by the authors' custom implementations using the Lorcon2 library [21]. A separate device (i.e. a desktop machine running TShark [22]), which was not associated with the network, acted as the monitor node and was used to capture the live network utilization traffic.

The attacks that have similar execution methodologies were placed into the same category. The categories are (a) injection attacks: generate a high number of correctly encrypted data frames (b) flooding attacks: generate a high volume of management frames per unit time, and (c) impersonation attacks: introduce an access point to broadcast beacon frames to advertise a pre-existing network (that is, the victim's network).

We analysed the AWID-CLS-R-Trn and AWID-CLS-R-Tst, which are the train and test dataset, respectively. The train dataset contains 1795575 records of which 1633190 are those of normal activities and 162385 are those of intrusive activities (that is, anomalies). The test dataset contains 575643 records of which 530785 are those of normal activities and 44858 are those of intrusive activities. Of the 15 attack types (with their variations), the train dataset contains 8 of the attack types, while the test dataset contains attacks of all 15 types. Thus, from the perspective of the train dataset, there are 7 novel attack types that have never been encountered before. The 7 novel attack types are the ChopChop Injection attacks; CTS, RTS, Disassociation, Fake Power Saving and Probe Request Flooding attacks; and the Hirte Impersonation attacks. In the dataset, only the class labels specifying the record category were given. They are Normal (that is, legitimate traffic), Flooding, Injection and Impersonation. Thus, we formulate this attack classification as a 4-class problem in this paper.

From the captured network traffic, each packet is represented as a vector of 155 attributes in the dataset as a record, with the last attribute being the corresponding category. Each record is composed of the MAC layer information (such as source address, destination address, initialization vector, extended service set identification) as well as Radiotap information such as the signal strength. All attributes have numeric or nominal values, except for SSID, which takes string values.

IV. Proposed Deep Learning Approach

In this work, we propose using a deep learning approach to derive the complex features with better discriminative ability to perform anomaly detection and attack classification. To achieve this, we utilized a Stacked Auto-encoder (SAE), which is a neural network built by stacking multiple layers of sparse auto-encoders. The output of each layer forms the input to the successive layer. We proposed two frameworks, which are composed of two and three hidden layers, respectively. The first layer learns the first order features from the raw inputs, while the second layer learns the features corresponding to the patterns from the first order features. The third layer in the second framework learns the features corresponding to the patterns from the second order features. The first hidden layer consists of 256 neurons, and the second hidden layer is made up of 128 neurons, while the third layer is composed of 64 neurons.

Consider a SAE with parameters W^l, b^l, denoting the parameters for the l^{th} auto-encoder. The output of the l^{th} layer with its input z^l is the auto-encoder, $a^{(l)}$. The encoding of the input feature vectors over the SAE is carried out by encoding each forward layer, as follows:

$$a^{(l)} = f(z^{(l)}) \tag{1}$$

$$z^{(l+1)} = W^{(l)}a^{(l)} + b^{(l)} \tag{2}$$

where f() is the activation function. The decoding of the SAE is then performed in the reverse order, that is, by decoding each layer backwards.

In recent years, the Rectified Linear Unit (ReLU) has been favoured over the historically widely used Sigmoid as the activation function due to its higher computational efficiency and robustness. The Sigmoid function (Equation 3 has an undesirable property of causing the gradients to tend towards zero in the event where the neuron's activation saturates. The resulting effect is that (almost) no signal will flow to the neuron's weights. ReLU, on the other hand, computes a simple thresholding function (Equation 4) and does not run into the issue of saturation. However, ReLU faces another problem of neurons being irreversibly taken off the training since the activation function goes to zero when $y < 0$ (as shown in Figure 1). In an improved version of ReLU, known as the Leaky ReLU (LReLU), the issue of neurons being taken off the training is resolved through the introduction of a small negative slope determined by a when $y < 0$, as shown in Equation 5 and Figure 2. The Parametric Rectified Linear Unit (PReLU) [23], is an enhanced version of LReLU. In PReLU, a is learnt during the training phase, while in LReLU, a is a fixed value. In [23], the authors showed that PReLU is the key factor in helping to produce results that surpass human-level performance on the Image-Net classification task.

$$\sigma(y) = \frac{1}{(1 + e^{-y})} \tag{3}$$

$$f(y) = max(0, y) \tag{4}$$

$$f(y) = ay \textit{ if } y \ < \ \textit{0 and f(y)} \ = \ y \textit{ if } y \ \geq 0 \tag{5}$$

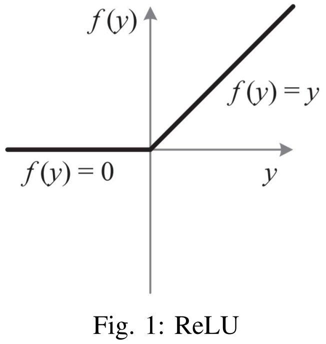

Fig. 1: ReLU

In the frameworks, the optimal SAE parameters are learnt. After which, we utilized Softmax regression (also known as multinomial logistic regression), which is a generalization of logistic regression, for the classification task. Softmax regression supports multi-class classification, which is more suited for our use case, while logistic regression only supports the binary class setting.

V. Experimental Evaluations and Discussions

Based on our preliminary dataset analysis, we applied data normalization to better represent the data and standardise the feature range, so as to facilitate the subsequent deep

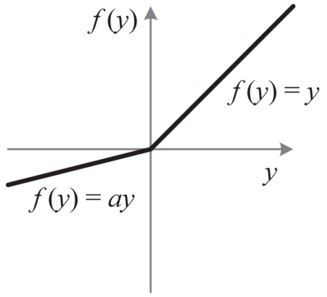

Fig. 2: Leaky ReLU and PReLU. a is fixed in Leaky ReLU, but adaptively learned in PReLU [23].

learning process. After which, we implemented our deep learning frameworks, and sent the train dataset to generate the self-learnt models. As mentioned in the previous section, PReLU is able to self-learn the optimal parameter, compared to LReLU where the parameter is a fixed value. Thus, the LReLU activation function was included here for illustrative purpose only. We randomly fixed a for the LReLU in our frameworks to be 0.001.

Based on our generated models, we performed the experimental evaluation using the test dataset, which contains all 15 attack types, and whereby 7 of them are novel. Table I shows the breakdown of the train dataset records, and Figure 3 and Figure 4 show the test dataset classification results of the 2-hidden-layer model and 3-hidden-layer model, respectively, when using different activation functions for the neurons. Table II to Table V show the test dataset classification results for the 2-hidden-layer framework with each of the activation functions, while Table VI to Table IX show the test dataset classification results for the 3-hidden-layer framework with each of the activation functions. The highest classification accuracy for each category within in each of our proposed frameworks is marked in bold.

Category	Number of Records
Normal	1633190
Injection	65379
Flooding	48484
Impersonation	48522
Total	1795575

TABLE I: Statistics of Train Dataset

Referring to Figure 3, our experimental results demonstrated that the generated 2-hidden-layer model using the PReLU activation function was able to obtain a consistently high and balanced classification rate for all categories, compared to the other three activation functions, and the prior art's best classifier using the J48 algorithm [9]. We shall refer to the prior art's best classifier as the J48 classifier from here on.

Our 2-hidden-layer PReLU based model achieved the highest overall classification accuracy of 98.6688%, which was mainly attributed to its good performance in identifying impersonation attacks when the other models, including the J48 classifier, failed to. Our 2-hidden-layer PReLU based model showed a significant improvement in the impersonation attack classification at an accuracy of 98.4959%, compared to

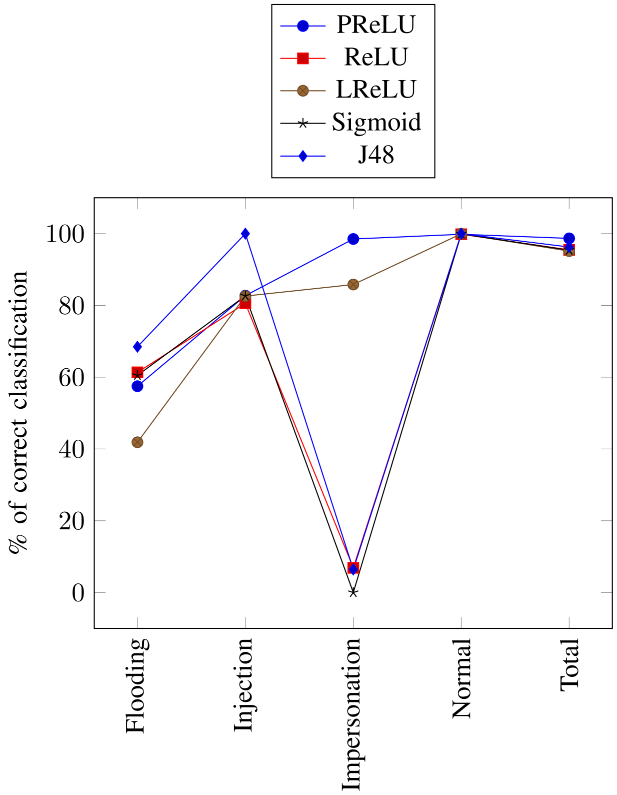

Fig. 3: Test results for the 2-hidden-layer architecture with different activation functions, and J48 classifier [9]

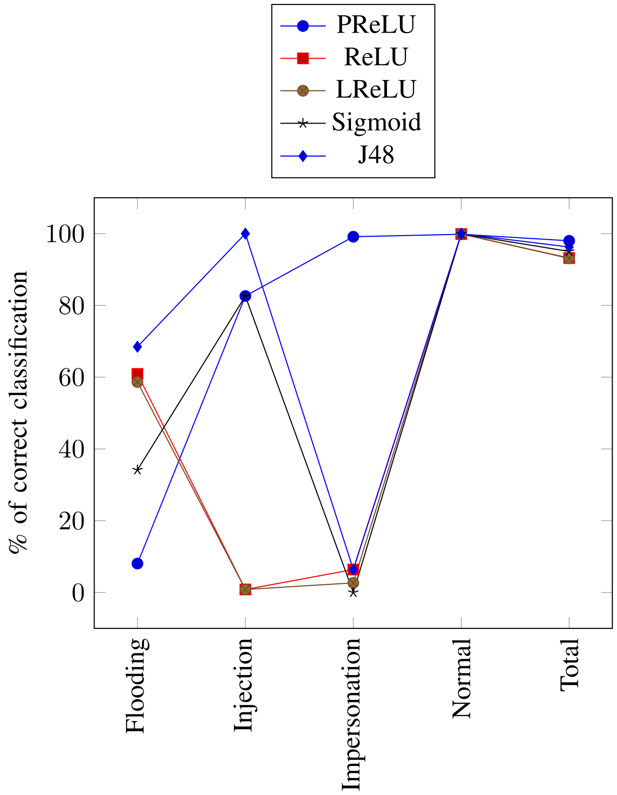

Fig. 4: Test results for the 3-hidden-layer architecture with different activation functions, and J48 classifier [9]

Category	Number of Records	Correctly Classified	Incorrectly Classified	Classification Accuracy (%)
Normal	530785	529750	1035	99.8050
Injection	16682	13799	2883	**82.7179**
Flooding	8097	4654	3443	57.478
Impersonation	20079	19777	302	**98.4959**
Total	575643	567980	7663	**98.6688**

TABLE II: Experimental Evaluation Results for the 2-hidden-layer architecture with PReLU using the Test Dataset

Category	Number of Records	Correctly Classified	Incorrectly Classified	Classification Accuracy (%)
Normal	530785	530443	342	**99.9356**
Injection	16682	13777	2905	82.5860
Flooding	8097	3389	4708	41.8550
Impersonation	20079	0	20079	0
Total	575643	547609	28034	95.1300

TABLE III: Experimental Evaluation Results for the 2-hidden-layer architecture with LReLU using the Test Dataset

Category	Number of Records	Correctly Classified	Incorrectly Classified	Classification Accuracy (%)
Normal	530785	529827	958	99.8195
Injection	16682	13441	3241	80.5719
Flooding	8097	4967	3130	**61.3437**
Impersonation	20079	1381	18698	6.8778
Total	575643	549616	26027	95.4786

TABLE IV: Experimental Evaluation Results for the 2-hidden-layer architecture with ReLU using the Test Dataset

Category	Number of Records	Correctly Classified	Incorrectly Classified	Classification Accuracy (%)
Normal	530785	530380	405	99.9237
Injection	16682	13788	2894	82.6520
Flooding	8097	4899	3198	60.5039
Impersonation	20079	0	20079	0
Total	575643	549067	26576	95.3832

TABLE V: Experimental Evaluation Results for the 2-hidden-layer architecture with Sigmoid using the Test Dataset

the LReLU, ReLU and Sigmoid activation functions, at 0%, 6.8778% and 0%, respectively. As was observed by the authors in [9], the impersonation attacks were the most challenging to detect. When they performed feature engineering and applied various conventional supervised learning algorithms, their best classifier was only able to achieve an accuracy of 6.4097% at classifying the impersonation attacks. Nonetheless, when compared to the J48 classifier, our 2-hidden-layer PReLU based model showed a drop in the classification of normal, injection and flooding traffic, by 0.1579%, 17.2701% and 10.9918%, respectively.

While the 3-hidden-layer PReLU based model is still able to achieve the highest overall accuracy at 98% compared to the other models and the J48 classifier, there is a significant drop in the flooding attack classification accuracy, as shown in Figure 4. This is due to the inability for this model to learn sufficiently good higher order features for the flooding attacks, as compared to the other classes. However, it is still able to attain a significantly high accuracy of 99.1334% in classifying the impersonation attacks, compared to the other models. This model is also able to maintain a good classification accuracy at 99.8129% and 82.6040% for the normal and injection attack traffic, respectively.

For both the 2-hidden-layer and 3-hidden-layer frameworks, the Sigmoid based models did not manage to attain the highest classification accuracy for any of the traffic class, while the LReLU and ReLU based models attained the highest classification accuracy for the normal and flooding attack traffic, respectively. The PReLU based models, on the other hand, attained the highest classification accuracies for both the injection and impersonation attacks. Taking an overall view, we observed that both the 2-hidden-layer and 3-hidden layer PReLU based models were able to provide a well-balanced classification for all four traffic types, with the 2-hidden-layer model having a better performance over the 3-hidden-layer model, as shown in Figure 3 and 4.

VI. Conclusions

In this paper, we proposed a deep learning approach for the IEEE 802.11 wireless network anomaly detection and attack classification problem. To the best of our knowledge, this is the first work that proposes a deep learning approach to perform IEEE 802.11 network anomaly classification. Our proposed frameworks are formed through the SAE architecture with two and three hidden layers. We also explored the utilization of different techniques as the activation functions for the hidden neurons. Our experimental results showed that our proposed approach is able to perform the 4-class classification, taking into consideration novel attacks, with a higher overall accuracy of 98.6688%, compared to state-of-the-art approaches.

Acknowledgement

This material is based on research work supported by the Singapore National Research Foundation under NCR Award

Category	Number of Records	Correctly Classified	Incorrectly Classified	Classification Accuracy (%)
Normal	530785	529792	993	99.8129
Injection	16682	13780	2902	**82.6040**
Flooding	8097	653	7444	8.0647
Impersonation	20079	19905	174	**99.1334**
Total	575643	564130	11513	**98.0000**

TABLE VI: Experimental Evaluation Results for the 3-hidden-layer architecture with PReLU using the Test Dataset

Category	Number of Records	Correctly Classified	Incorrectly Classified	Classification Accuracy (%)
Normal	530785	530437	348	**99.9344**
Injection	16682	140	16542	0.8392
Flooding	8097	4749	3348	58.6514
Impersonation	20079	539	19540	2.6844
Total	575643	535865	39778	93.0898

TABLE VII: Experimental Evaluation Results for the 3-hidden-layer architecture with LReLU using the Test Dataset

Category	Number of Records	Correctly Classified	Incorrectly Classified	Classification Accuracy (%)
Normal	530785	530102	683	99.8713
Injection	16682	140	16542	0.8392
Flooding	8097	4931	3166	**60.8991**
Impersonation	20079	1286	18793	6.4047
Total	575643	536459	39184	93.1930

TABLE VIII: Experimental Evaluation Results for the 3-hidden-layer architecture with ReLU using the Test Dataset

Category	Number of Records	Correctly Classified	Incorrectly Classified	Classification Accuracy (%)
Normal	530785	530433	352	99.9337
Injection	16682	13777	2905	82.5860
Flooding	8097	2765	6142	34.1485
Impersonation	20079	0	20079	0
Total	575643	546975	28668	95.0198

TABLE IX: Experimental Evaluation Results for the 3-hidden-layer architecture with Sigmoid using the Test Dataset

No. NRF2014NCR-NCR001-034. We would also like to thank the authors in [9] for making the data available to us for this research work.

REFERENCES

[1] V. L. L. Thing, “Cyber security for a smart nation,” in *IEEE Computational Intelligence and Computing Research, pp. 1-3*, 2014.

[2] M. Kuzlu, M. Pipattanasomporn, and S. Rahman, “Review of communication technologies for smart homes/building applications,” in *IEEE Innovative Smart Grid Technologies Conference*, 2015.

[3] S. R. Fluhrer, I. Mantin, and A. Shamir, “Weaknesses in the key scheduling algorithm of RC4,” in *Selected Areas in Cryptography, Vol. 2259, pp. 1-24, Lecture Notes in Computer Science, Springer*, 2001.

[4] A. Stubblefield, J. Ioannidis, and A. D. Rubin, “A key recovery attack on the 802.11b wired equivalent privacy protocol (WEP),” in *ACM Transactions on Information and System Security, Vol. 7, No. 2, pp. 319-332*, 2004.

[5] E. Tews, R.-P. Weinmann, and A. Pyshkin, “Breaking 104 bit WEP in less than 60 seconds,” in *International Conference on Information Security Applications*, 2007.

[6] A. Stubblefield, J. Ioannidis, and A. D. Rubin, “How to crack a Wi-Fi networks WEP password with BackTrack,” http://lifehacker.com/5305094/how-to-crack-a-wi-fi-networks-wep-password-with-backtrack, accessed on 12 September 2016.

[7] M. S. Ahmad, “WPA Too!” in *DEFCON 18*, 2010.

[8] A. Pash, “How to crack a Wi-Fi networks WPA password with reaver network security tools,” http://lifehacker.com/5873407/how-to-crack-a-wi-fi-networks-wpa-password-with-reaver, accessed on 12 September 2016.

[9] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, “Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset,” in *IEEE Communications Surveys and Tutorials, Vol. 18, No. 1, pp. 184-208*, 2016.

[10] T. Ahmed, B. Oreshkin, and M. Coates, “Machine learning approaches to network anomaly detection,” in *USENIX Workshop on Tackling Computer Systems Problems with Machine Learning Techniques*, 2007.

[11] M. A. Salama, H. F. Eid, R. A. Ramadan, A. Darwish, and A. E. Hassanien, “Hybrid intelligent intrusion detection scheme,” in *Soft Computing in Industrial Applications, pp. 293-303, Springer*, 2011.

[12] U. Fiore, F. Palmieri, A. Castiglione, and A. D. Santis, “Network anomaly detection with the restricted boltzmann machine,” in *Nerocomputing, Vol. 22, pp. 13-23*, 2013.

[13] D. K. Bhattacharyya and J. K. Kalita, “Network anomaly detection: a machine learning perspective,” in *CRC Press*, 2013.

[14] N. P. Neelakantan and C. Nagesh, “Role of feature selection in intrusion detection system for 802.11 networks,” in *International Journal on Smart Sensors Ad Hoc Network, Vol. 1, No. 1, pp. 98-101*, 2011.

[15] Q. Niyaz, W. Sun, A. Y. Javaid, and M. Alam, “A deep learning approach for network intrusion detection system,” in *EAI International Conference on Bio-inspired Information and Communications Technologies, pp. 21-26*, 2015.

[16] KDD Cup, “Network traffic capture dataset,” http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, accessed on 12 September 2016.

[17] Aircrack-ng, “Wi-Fi network security tools,” https://www.aircrack-ng.org, accessed on 12 September 2016.

[18] KALI Tools, “Mdk3,” http://tools.kali.org/wireless-attacks/mdk3, accessed on 12 September 2016.

[19] Metasploit, “Penetration testing software,” https://www.metasploit.com, accessed on 12 September 2016.

[20] J. Wright, “File2air,” http://www.willhackforsushi.com/?page_id=19, accessed on 12 September 2016.

[21] Lorcon2, “Wireless packet injection,” https://code.google.com/archive/p/lorcon, accessed on 12 September 2016.

[22] TShark, “Network traffic analyser,” https://www.wireshark.org, accessed on 12 September 2016.

[23] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: surpassing human-level performance on Image-Net classification,” in *IEEE International Conference on Computer Vision (ICCV)*, 2015.