Detecting Impersonation Attack in WiFi Networks Using Deep Learning Approach*

Muhamad Erza Aminanto and Kwangjo Kim

Cryptology and Information Security Lab, School of Computing,
Korea Advanced Institute of Science and Technology (KAIST)
Daejeon, Republic of Korea
{aminanto,kkj}@kaist.ac.kr

Abstract. WiFi network traffics will be expected to increase sharply in the coming years, since WiFi network is commonly used for local area connectivity. Unfortunately, there are difficulties in WiFi network research beforehand, since there is no common dataset between researchers on this area. Recently, AWID dataset was published as a comprehensive WiFi network dataset, which derived from real WiFi traces. The previous work on this AWID dataset was unable to classify Impersonation Attack sufficiently. Hence, we focus on optimizing the Impersonation Attack detection. Feature selection can overcome this problem by selecting the most important features for detecting an arbitrary class. We leverage Artificial Neural Network (ANN) for the feature selection and apply Stacked Auto Encoder (SAE), a deep learning algorithm as a classifier for AWID Dataset. Our experiments show that the reduced input features have significantly improved to detect the Impersonation Attack.

1 Introduction

In the near future, wireless network traffics will rise drastically. According to Cisco Visual Networking Index report [1], wireless traffics will account for two-thirds of total Internet traffics by 2020. Then, we expect that 66% of IP traffics come from WiFi and mobile devices. For local area connectivity, WiFi networks (so called 802.11 networks) are widely deployed. The increased usage of WiFi network will be followed by unknown attacks and vulnerabilities accordingly. There are several security protocol for WiFi network such as Wired Equivalent Privacy (WEP) and WiFi Protected Access 2 (WPA2) [2]. Our goal is to make these protocols more secure against unauthorized traffic by Intrusion Detection System (IDS) using up-to-date deep learning approach.

However, IDS research on WiFi network was difficult since there was no common dataset so for. Recently, Kolias *et al.* [2] published a comprehensive WiFi

* This work was partly supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP) (B0101-16-1270, Research on Communication Technology using Bio-Inspired Algorithm) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. NRF-2015R1A2A2A01006812)

network dataset, called Aegean WiFi Intrusion Dataset (AWID). This dataset contains 14 distinct attacks categorized by three different attacks. There are a number of previous work that use AWID dataset: Usha and Kavitha *et al.* [8], Alotaibi [4], *etc.* Unfortunately, all the previous publications were unable to improve the Impersonation Attack detection. Impersonation Attack is one of forging activity in order to take an advantage over others. Usually, it masquerades a legitimate device in a WiFi network. So, we try to improve the Impersonation Attack using AWID dataset. Major improvements of the previous IDSs can be achieved by leveraging the latest highly effective machine learning methods [5], so called deep learning. The computations for most of deep learning are heavy. Reducing the dimensionality of input features is one of light candidate solutions.

We leverage Artificial Neural Network (ANN) for the feature selection. The weight from trained models mimics the importance of the correspondence input. By selecting the important features only, the training process becomes lighter and faster than before. To validate our approach, we use *Stacked Auto Encoder* (SAE) as a classifier, which is one of popular deep learning algorithms, since this employs consecutive layers of processing stages in hierarchical manners for pattern classification and feature or representation learning [3]. Our experiment shows that the reduced input features are sufficient for SAE algorithm to achieve better detection rate for Impersonation Attack.

This paper is organized as follows: Section 2 reviews related work in brief. We describe our proposed scheme in Section 3. Section 4 reports our experimental results and analysis. Conclusion and future work of this paper will be presented in Section 5.

2 Related Work

The importance of feature selection for IDS dataset was introduced by Kayacik *et al.* [3]. They investigated the relevance of each feature in KDD 99 Dataset and provided the information gain for each feature. Their conclusion ends with the list of the most relevant features for each class label. Afterwards, there are several publications that employ feature selection method. Zaman and Karray [6] categorized the IDS based on the TCP/IP network model using a feature selection method called Enhanced Support Vector Decision Function (ESVDF). Louvieris *et al.* [7] proposed an effects-based feature identification IDS using Naive Bayes as a feature selection method.

In this paper, we focus on WiFi network. Kolias *et al.* [2] published a comprehensive WiFi network traces that becomes a public dataset for 802.11 networks. They checked various machine learning algorithms to validate their dataset in a heuristic manner. Among all the classification results, Impersonation Attack detection is the most unsatisfactory result. Hence, our goal is to improve the Impersonation Attack detection. We leverage recent deep learning algorithm published by Wang [9]. In 2015, Wang has shown that neural networks especially the deep neural networks can be used for finding features in the raw network flow data. We use AWID dataset [2] for our experimental purpose. Recently, Usha

and Kavitha *et al.* [8] leveraged AWID dataset and successfully improved the overall detection rate. However, they didn't focus on improving Impersonation Attack detection which is the one of most concerns by Kolias *et al.* [2]

3 Our Approach

In this section, we briefly describe our approach to improve Impersonation Attack detector. Basically there are two main tasks, feature selection and classification. Fig.1 shows our proposed architecture which starts with data collection task, and then training phase, ends by validation phase. We use a real WiFi networks trace, AWID [2]. The preprocessing should be conducted before using AWID dataset. The preprocessed dataset will be fed into feature selection method. Feature selection method which contains ANN learner results a list of important features to detect Impersonation Attacks. In order to validate this reduced feature list, we employ one of deep learner, SAE.

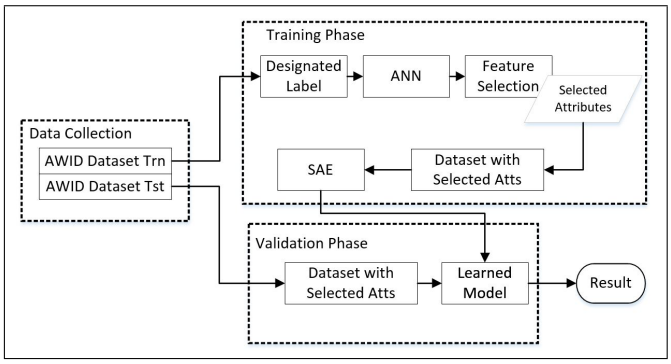

Fig. 1. Our Proposed Architecture

3.1 Data Preprocessing

AWID dataset [2] not only contains discrete type, but also consists of continuous, and symbolic types with flexible value range. This format will be difficult for most of pattern classification methods to learn [10]. The preprocessing should be conducted in advance. There are two main steps for the preprocessing: the mapping step from symbolic-valued attributes into numeric values and the normalizing step. Target class will be mapped into one of these integer-valued classes: 1 for Normal, 2 for Impersonation, 3 for Flooding and 4 for Injection Attacks. Meanwhile, symbolic attributes such as receiver, destination, transmitter, source address, *etc.*, will be mapped into integer values with scale from 1 to N where N is the number of symbols. Some attributes that have hexadecimal values such as WEP Initialization Vector (IV) and Integrity Check Value (ICV) need to be

casted into the integer values. Also, there are some attributes with continues values as the timestamp. In addition, AWID dataset also contains the question mark ("?") for those not available value on the corresponding attributes. This question mark can be assigned with zero value. After all attributes values casted into the integer values, each of the attributes linearly normalized between zero and one. Eq. (1) shows the normalizing formula.

$$z_i = \frac{x_i - min(x)}{max(x) - min(x)}, \tag{1}$$

where z_i denotes the normalized value, x_i refers to the corresponding attribute value and $min(x)$ and $max(x)$ are the minimum and maximum values of the attribute, respectively.

3.2 Feature Selection

Feature selection belongs to feature learning which contains feature extraction and feature selection [9]. Feature learning is defined as the ability to model the traffic behavior from the most characterizing raw input. Feature learning is very important to show the correlation between the detection performance and the traffic model quality [11]. However, feature extraction and selection are different terms. Feature extraction refers to deriving new features from raw feature space to be informative and non-redundant. Those features in raw feature and new generated features are usually different. On the other hand, feature selection is to select several features from the raw feature space. So, new generated features are just selected from the raw one without transformation. Both feature extraction and selection are aiming the smaller number of new generated features than the raw one. In this paper, *we adopt feature selection using ANN explicitly while feature extraction is deployed in SAE implicitly.*

We apply ANN in order to improve Impersonation Attack detection rate. By using ANN, we are able to choose some features which are important to learn the Impersonation Attack model based on the heuristic weights from ANN learning. We train our ANN with two target classes only, Normal and Impersonation Attack, instead of four target classes. Fig. 2 shows the ANN model where $b1$ and $b2$ represent the bias values for the corresponding hidden layer, respectively.

We use the first hidden layer only for feature selection and consider the weight value between the first two layers to choose the important input features. The weight represents the contribution of the input features to the first hidden layer features. Very small or even zero W_{ij} means that the corresponding input feature x_j is meaningless for further propagation. So, one hidden layer is sufficient since we consider the weights in the first hidden layer only. We define the importance value of each input feature, as expressed by Eq. (2).

$$V_j = \sum_{i=1}^{h} |W_{ij}|, \tag{2}$$

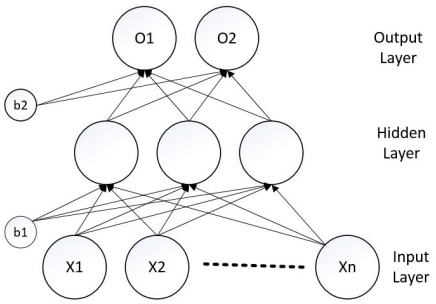

Fig. 2. ANN Model

where h is the number of neurons in the first hidden layer. In order to select the most important features, we sort the input features according to V_j value in a descending order. We pick some features that have V_j value bigger than a threshold value.

3.3 Classification

In order to validate the performance of chosen features, we utilize a deep learning algorithm. Deep learning is a class of machine learning methods, which exploits the cascaded layers of data processing stages in hierarchical structure for unsupervised feature learning and for pattern classification. We choose SAE as a classifier because SAE is able to replace original features with unsupervised approach and has hierarchical feature extraction phase. Basically, Auto Encoder (AE) model is similar to ANN as shown in Fig. 3.

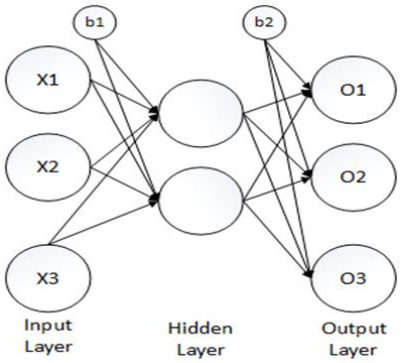

Fig. 3. AE Model

Compared with ANN model, AE model is characterized by the same number of the input and output layers. Meanwhile, the nodes in the hidden middle

layer represent new features set with lower dimension. This architecture leads to an ability that can reconstruct the data after complicated computations. Since AE aims to learn a compact set of data efficiently, it can be stacked to build deep networks. Each training results of the middle layer can be cascaded. This structure is called SAE, which can learn lots of new features in different depths [9]. Fig.4 shows the proposed SAE architecture used in this paper.

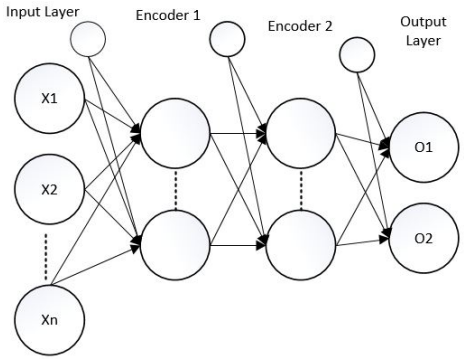

Fig. 4. Proposed SAE Architecture

We employ two hidden (encoder) layers. The features that were generated from the first encoder layer used as the training data in the second encoder layer. Meanwhile, the size of each hidden representation is decreased accordingly so that the encoder in the second encoder layer learns an even smaller representation of the input data. We complete our stacked architecture with the supervised learning approach by *softmax* regression function, which is a generalization of logistic regression for classification purposes using labels from training data. The function can take more than two possible classes and is commonly cascaded after any unsupervised learning methods.

4 Evaluation

In this section, we show the detailed steps before conducting our experiments and the results. We first explain our four Tests here and the dataset preparation followed by the experimental results and analysis.

4.1 Experiment Setup

We constructed four Tests in order to show that our proposed scheme can improve Impersonation Attack detection rate. Table 1 shows the summary of our four Tests.

Table 1. Summary of our four Tests

Test	Trn Dataset	Tst Dataset	Attributes	Feature Selection	Target Classes
(a)	Unbalanced	Unbalanced	154	N/A	4
(b)	Balanced	Balanced	154	N/A	2
(c)	Balanced	Balanced	35	ANN	2
(d)	Balanced	Unbalanced	35	ANN	2

Four Tests are described as follows:

1. Test (a): All target classes and all attributes.
   In Test (a) as shown in Fig. 5, we feed the SAE classifier with the normalized dataset without any feature selection. The aim of Test (a) is to show the basic performance before applying feature selection method. All four target classes (Normal, Injection, Impersonation and Flooding) are included and original 154 input features are involved.

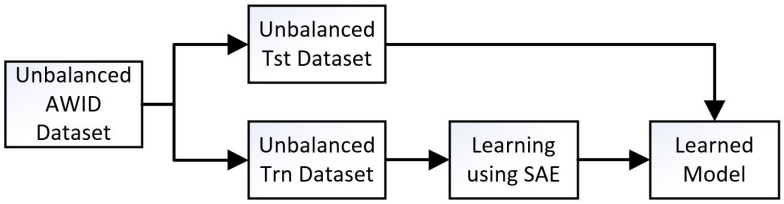

Fig. 5. Test (a)

2. Test (b): Balanced dataset and all attributes.
   Test (b) as shown in Fig. 6 is similar with Test (a). The difference is that Test (b) uses balanced dataset and considers two target classes (Normal and Impersonation) only. Test (b) aims to show the performance if we focus on detecting Impersonation Attack only with the whole original attributes. Two target classes are included and 154 input features are involved.

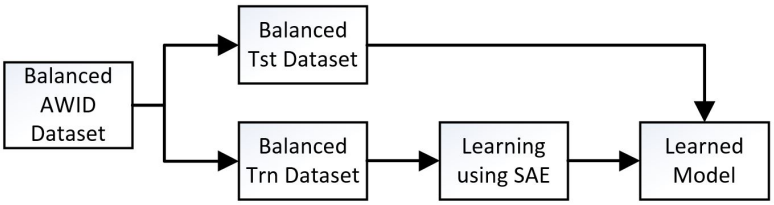

Fig. 6. Test (b)

3. Test (c): Balanced dataset and reduced attributes.
   We employ our proposed scheme in Test (c) as shown in Fig. 7. We first generate the balanced dataset that aim to detect Impersonation Attack properly. We train our ANN with the balanced dataset and output a list of selected important features. Next, balanced test dataset with selected features only is fed into SAE classifier. The aim of Test (c) is to show that our proposed scheme can outperform Test (a). Two target classes (Normal and Impersonation) are included and selected 35 input features are involved.

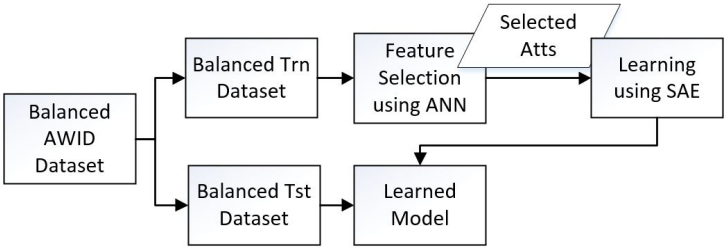

Fig. 7. Test (c)

4. Test (d): Full dataset test from learned system.
   Test (d) as shown in Fig. 8 is almost same with Test (c). The only difference is that Test (d) uses full test dataset which is unbalanced. The goal of Test (d) is to show that the learned model by Test (c) is able to work in real detection process. In Test (d), two target classes (Normal and Impersonation) are included and selected 35 input features are involved.

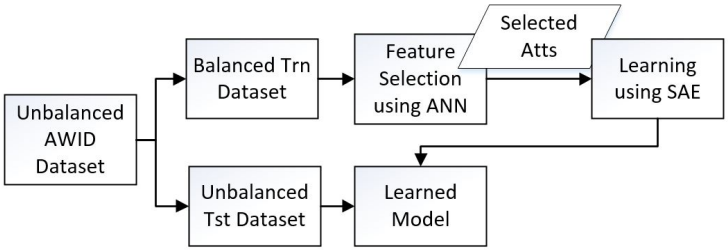

Fig. 8. Test (d)

In addition, we deploy an experiment environment: MATLAB R2016a which runs in Intel(R) Xeon(R) CPU E-3-1230v3@3.30 GHz, RAM 32 GB. Also, as the performance evaluation, we use Detection Rate (DR) which is the number of correctly detected attacks divided by the total number of attacks. On the other

hand, False Positive Rate (FPR), defined as the number of normal instances that are classified incorrectly as attacks divided by the total number of normal instances. Intuitively, we should maintain DR as high as possible and FPR as low as possible.

4.2 Dataset

The AWID dataset [2] contains real trace of WiFi traffic. There are two types of AWID dataset based on the number of target classes. The first type named "CLS" with four target classes and the second named "ATK" with 16 target classes. The 16 classes of "ATK" dataset belong to four attack categories in "CLS" dataset. As an example, *Caffe-Latte*, *Hirte*, *Honeypot* and *EvilTwin* attack types listed in "ATK" dataset, are classified as Impersonation Attack in "CLS" dataset. AWID dataset also divided into two types based on the size of data instances included, namely full and reduced dataset. In this paper, we use the reduced "CLS" AWID dataset.

We need to do balancing of the AWID dataset since it contains a huge number of normal instances compared to attack instances, especially Impersonation Attack. We need to take balancing for training purpose. Once the IDS model was successfully trained using the balanced dataset, we can validate the model using the unbalanced dataset as shown in Test (d). Table 2 shows the distribution of each classes in balanced and unbalanced AWID dataset.

Table 2. Distribution of each classes in balanced and unbalanced dataset

	Balanced		Unbalanced	
	Normal	Impersonation	Normal	Impersonation
Train	163,319	48,522	1,633,190	48,522
Test	53,078	20,079	530,785	20,079

4.3 Performance Evaluation

In this section, we provide performance results for all Tests. Table 3 shows the performance of Test (a), comparison between our scheme and original work [2]. In Test (a), we do nothing except fed the dataset into SAE learner. This approach already improves the Impersonation Attack detection with classified 13,087 (65%) instances correctly. Also, this approach has comparable performance in classifying Normal and Injection Attacks. However, the DR for Flooding Attack is still unsatisfactory.

Test (b) focuses on detecting Impersonation Attack in balanced dataset. And as expected, it shows the highest DR for both Impersonation Attack and Normal classes among all Tests as shown in Table 4. However, it took about 30 minute

Table 3. Performance comparison for Test (a)

	Normal	Impersonation	Flooding	Injection
Test (a)	530,028	13,087	2,555	16,675
Kolias *et al.* [2]	530,765	4,419	5,974	16,680

Table 4. Performance comparison for Tests (b), (c) and (d)

	Normal	DR	FPR	Duration
Kolias *et al.* [2]	530,765 (99.9%)	4,419 (22%)	14,187 (2.75%)	-
Test (b)	52,427 (98.6%)	18,613 (92.7%)	651 (1.23%)	30 min
Test (c)	51,826 (97.6%)	17,033 (85%)	1,252 (2.36%)	5 min
Test (d)	518,237 (97.6%)	17,033 (85%)	12,548 (2.36%)	5 min

for training and validating process. Meanwhile, Test (c) and (d) took about 5 minute only.

In Test (c), we first train our ANN learner as a feature selection method as expressed in Fig. 7. We consider weights on first hidden layer only. Fig. 9 shows the distribution of each input feature's weight. In order to get selected important features, we set the threshold value to 15 due to most of attributes have weight value lower than 15. We may adjust the threshold value in order to get optimal result. Hence, 35 features are selected, as listed in Table 5.

We use the balanced dataset in Test (c) in order to learn the Impersonation Attack properties properly. We select 35 attributes only based on previous result, and fed it into SAE learner. Table 4 shows the performance comparison for Tests (b), (c) and (d). The DR for Impersonation Attack in Test (b) is significantly improved compared to Kolias *et al.* [2]. Also, the DR for Normal class is slightly below the original work. Test (b) also shows the significant improvement on FPR.

Test (d) is the same Test (c) with different test dataset. We use the unbalanced dataset which contains full test instances from AWID dataset [2]. By this approach, we demonstrate that our learned model is able to be used for detecting Impersonation Attack properly in real network traffic. Table 4 shows that the DR for both classes are the same for Tests (c) and (d). This result supports our claim that the learned model by Test (c) can be implemented in real network traffic.

Table 5. Selected features for Impersonation Attack detector

Index	Feature Name	Description
4	frame.time_epoch	Epoch Time
7	frame.time_relative	Time since reference or first frame
8	frame.len	Frame length on the wire
29	radiotap.present.rxflags	RX flags
38	radiotap.mactime	MAC timestamp
47	radiotap.datarate	Data rate (Mb/s)
62	radiotap.antenna	Antenna
66	wlan.fc.type	Type
67	wlan.fc.subtype	Subtype
68	wlan.fc.ds	DS status
70	wlan.fc.retry	Retry
72	wlan.fc.moredata	More Data
73	wlan.fc.protected	Protected flag
77	wlan.da	Destination address
79	wlan.sa	Source address
80	wlan.bssid	BSS Id
82	wlan.seq	Sequence number
88	wlan.ba.bm	Block Ack Bitmap
93	wlan_mgt.fixed.capabilities.privacy	Privacy
94	wlan_mgt.fixed.capabilities.preamble	Short Preamble
98	wlan_mgt.fixed.capabilities.short_slot_time	Short Slot Time
104	wlan_mgt.fixed.listen_ival	Listen Interval
107	wlan_mgt.fixed.timestamp	Timestamp
108	wlan_mgt.fixed.beacon	Beacon Interval
112	wlan_mgt.fixed.auth_seq	Authentication SEQ
113	wlan_mgt.fixed.category_code	Category code
122	wlan_mgt.tim.dtim_period	DTIM period
125	wlan_mgt.country_info.environment	Environment
126	wlan_mgt.rsn.version	RSN Version
127	wlan_mgt.rsn.gcs.type	Group Cipher Suite type
140	wlan.wep.iv	Initialization Vector
141	wlan.wep.key	Key Index
142	wlan.wep.icv	WEP ICV
144	wlan.ccmp.extiv	CCMP Ext. Initialization Vector
148	wlan.qos.ack	Ack Policy

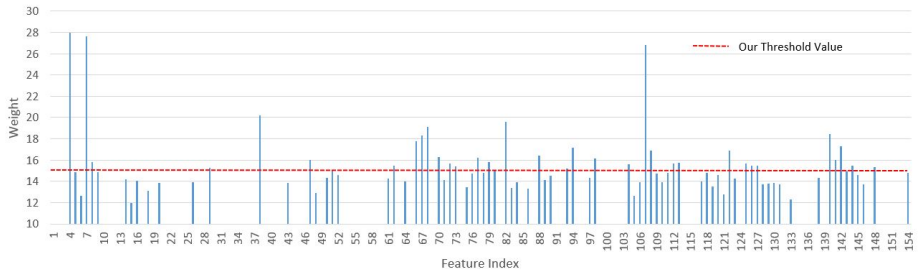

Fig. 9. Distribution of Each Input Feature's Weight

5 Conclusion and Future Work

We address the limitation of the previous work which was unable to detect Impersonation Attack properly. We employ feature selection method in order to select the most important attributes for detecting Impersonation Attack. We selected 35 attributes based on our ANN learning. Our experiments using SAE learner with selected features show significant improvements compared to Kolias *et al.* [2].

In the near future, we will conduct experiments for all attack classes of AWID dataset. In addition, combining several learning methods as an ensemble learning is a challenging issue in order to achieve optimal IDS in WiFi network.

Acknowledgment

The authors are very grateful to anonymous reviewers for their valuable feedbacks and suggestions.

References

1. Cisco Visual Networking Index: Forecast and Methodology, 2015-2020, `http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html`
2. Kolias, C., Kambourakis, G., Stavrou, A., Gritzalis, S.: Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset. IEEE Communications Surveys and Tutorials, 18(1), pp. 184-208. IEEE (2015)
3. Kayacik H., Zincir-Heywood A.N., Heywood M.I.: Selecting features for intrusion detection: A feature relevance analysis on KDD 99 intrusion detection datasets. Proceedings of the 3rd annual conference on privacy, security and trust, PST (2005)
4. Alotaibi, B.: A majority voting technique for wireless intrusion detection systems, University of Bridgeport (2016)
5. Sommer, R., Paxson, V.:Outside the closed world: On using machine learning for network intrusion detection. Security and Privacy, 2010 IEEE Symposium on (2010)
6. Zaman, S., Karray, F.: Lightweight IDS based on features selection and IDS classification scheme. In IEEE Computational Science and Engineering, CSE'09, International Conference on, 3, pp. 365-370. IEEE (2009)
7. Louvieris, P., Clewley, N. Liu, X.: Effects-based feature identification for network intrusion detection. Neurocomputing, Elsevier, 121, pp.265–273. IEEE (2013)
8. Usha, M., Kavitha, P.: Anomaly based intrusion detection for 802.11 networks with optimal features using SVM classifier. Springer Wireless Networks, The Journal of Mobile Comm., Computation and Information, 22, pp. 1–16. Springer (2016)
9. Wang, Z.: The application of deep learning on traffic identification. BlackHat USA (2015)
10. Sabhnani, M. and Serpen, G.: Application of machine learning algorithms to KDD intrusion detection dataset within misuse detection context. In Proceedings of the International Conference on Machine Learning: Models, Technologies, and Applications. pp. 209-215. (2003)
11. Palmieri, F., Fiore, U., Castiglione, A.: A distributed approach to network anomaly detection based on independent component analysis. In Concurrency Computation Practice and Experience Bd. 26, Nr. 6, S. pp. 1113-1129. Wiley (2013)