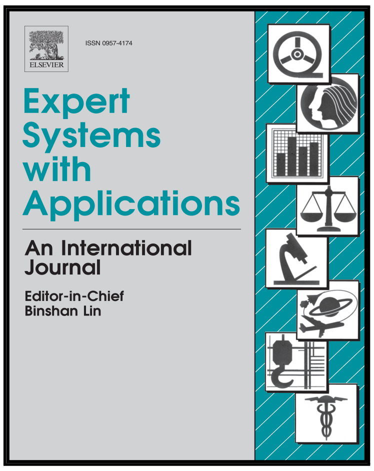


An Anomaly-based Intrusion Detection System in Presence of Benign Outliers with Visualization Capabilities

Amin Karami

PII: S0957-4174(18)30276-8
DOI: 10.1016/j.eswa.2018.04.038
Reference: ESWA 11952

To appear in: *Expert Systems With Applications*

Received date: 18 December 2017
Revised date: 15 April 2018
Accepted date: 28 April 2018

Highlights

- Taking into consideration benign outliers into IDS design.
- Improving SOM for anomaly visualization in presence of benign outliers.
- Introducing a new visualization by SOM without additional computational costs.
- Several datasets are evaluated to show the detection efficiency and accuracy.
- Comparison with several existing methods has been done.

An Anomaly-based Intrusion Detection System in Presence of Benign Outliers with Visualization Capabilities

Amin Karami[1]

Computer Science & Informatics (CS&I) department, University of East London (UEL), University Way, E16 2RD, UK

Abstract

Abnormal network traffic analysis through Intrusion Detection Systems (IDSs) and visualization techniques has considerably become an important research topic to protect computer networks from intruders. It has been still challenging to design an accurate and a robust IDS with visualization capabilities to discover security threats due to the high volume of network traffic. This research work introduces and describes a novel anomaly-based intrusion detection system in presence of long-range independence data called benign outliers, using a neural projection architecture by a modified Self-Organizing Map (SOM) to not only detect attacks and anomalies accurately, but also provide visualized information and insights to end users. The proposed approach enables better analysis by merging the large amount of network traffic into an easy-to-understand 2D format and a simple user interaction. To show the performance and validate the proposed visualization-based IDS, it has been trained and tested over synthetic and real benchmarking datasets (NSL-KDD, UNSW-NB15, AAGM and VPN-nonVPN) that are widely applied in this domain. The results of the conducted experimental study confirm the advantages and effectiveness of the proposed approach.

Keywords: Anomaly Detection, Intrusion Detection System, Benign Outlier, Visualization, Self-Organizing Map

1. Introduction

Intrusion Detection Systems (IDSs) have considerably become a required asset to the computer security ianfrastructure almost in all organizations connected to the Internet. With the growth of Internet and computer networks technologies, infrastructures, applications, and protocols, network traffic analysis for detecting unauthorized accesses, misbehaving and anomalous traffic has recently become one of the hot research topics in network security Jia et al. (2016); Ahmed et al. (2016); Karami (2015a). IDS can differentiate between normal (well-behaved) traffic and abnormal (misbehaved or malicious) traffic or violations through monitoring a network or system. Most analysis approaches in IDS are designed to detect intrusions by misuse detection or anomaly detection. The misuse detection algorithms discover attacks based on the pattern extracted from known intrusions (predefined signatures), while anomaly detection algorithms discover attacks and abnormal traffic patterns based on the deviations from the established profile that contains normal patterns of well-behaved traffic Luo & Xia (2014); Yu et al. (2010). Misuse detection techniques propose very low false positive rate and very high detection rate for known attacks that have already been defined in the signature database, but cannot detect new types of attacks which do not exist in the signature database. In contrast, anomaly detection techniques

Email address: `a.karami@uel.ac.uk` (Amin Karami)
URL: `https://www.uel.ac.uk/staff/k/amin-karami` (Amin Karami)
[1]Corresponding author, Telephone: +44 20 8223 3347

provide high performance by detecting unknown attacks. Nevertheless, the major drawbacks of multi-class anomaly-based IDSs exist in terms of lower detection precision and higher false positive rate Chen et al. (2016); Jabez & Muthukumar (2015); Powersa & He (2008).

To overcome the anomaly-based IDS weaknesses, various artificial and computational intelligence algorithms either integrating with meta-heuristic optimization approaches or without them are investigated, such as fuzzy logic Karami & Guerrero-Zapata (2015b); Feizollah et al. (2013), Support Vector Machine (SVM) Kabir et al. (2017); Bao & Wang (2016), Radial Basis Function (RBF) Karami & Guerrero-Zapata (2015c); Bi et al. (2009), Artificial Neural Network (ANN) Subba et al. (2016); Hodo et al. (2016), Self-Organizing Map (SOM) la Hoz et al. (2015); Karami & Guerrero-Zapata (2014); dong Wang et al. (2007), Adaptive Neuro-Fuzzy Inference System (ANFIS) Devi et al. (2017); Karami & Guerrero-Zapata (2015a), and Principle Component Analysis (PCA) An & Weber (2017); Khalid et al. (2015). Nevertheless, the major drawbacks of anomaly-based IDSs exist in terms of the lower detection precision and the higher false positive rate in presence of low-frequent patterns called *outliers*, resulting in weaker detection stability Karami & Guerrero-Zapata (2015b); Jabez & Muthukumar (2015); Luo & Xia (2014).

The low-frequent data patterns are mostly *malignant outliers*, which adversely affect the clustering quality because they are clearly so far from main data distribution. Detecting and removing such malignant outliers improves the clustering accuracy. Much of the research on clustering attempts to remove these using ”outlier removal techniques” Salim & Razak (2016); Hachmi et al. (2015); He et al. (2003) or combine meta-heuristic optimization algorithms with machine learning techniques Zhang et al. (2016); Karami & Guerrero-Zapata (2015c); Chen et al. (2015); Karami & Guerrero-Zapata (2015b). In contrast, *benign outliers* have been getting less attention, most approaches simply ignore them during training Karami & Guerrero-Zapata (2014); Luo & Xia (2014). Benign outliers are the long-term independent data points, that are mostly not included in a 90%-95% confidence interval of normally distributed data Jach & Kokoszka (2008). Since benign outliers are inherently a part of original data (see Figure 1 in Section 2) and they must be available for training purposes (i.e., they are not severe or malignant outliers), removing them might result in inappropriate training, unstable and diverge modelling. It means that, benign outliers exist in all the datasets and the user should decide to take into consideration a small portion of data (i.e., out of 95% of normal distribution) as benign or a large portion (i.e., out of 90% of normal distribution) as benign. In our research, we have particularly considered benign outliers to improve the stability and the robustness of anomaly-based intrusion detection systems in terms of the higher detection rate and the lower false alarm rate at the same time.

On the other hand, the biggest challenge for network administrators is visualization capabilities of IDS Abdullah et al. (2005) to demonstrate useful information and insights about network traffic. End users expect to understand a greater amount of data in shorter time to consider an applied IDS as an useful system Elhenawy et al. (2011). Due to the limitations in human cognitive and perceptual ability within discovering a large amount of knowledge by IDS, visualization strategies must be taken into consideration. Visualization of intrusion detection enables better analysis and response because an intrusion is recognized intuitively rather than alert flooding to network administrator via involving a network administrator in the analysis Etoty & Erbacher (2014); Elhenawy et al. (2011). Hence, visualization strategies provide several advantages, mainly merging huge volumes of data into simple and effective graphs and providing easy-to-understand analysis format Ibrahim et al. (2017); Karami (2015b); Luo & Xia (2014); Karami & Johansson (2014b).

The purpose of this paper is to develop a novel anomaly-based intrusion detection system that provides higher detection and lower false alarm precision at the same time in presence of benign outliers alongside visualization capabilities. To do so, we applied a modified Self-Organizing Map (SOM) to be able to satisfy both mentioned objectives. SOM is a widely used unsupervised clustering algorithm with visualization capabilities. Its neurons are arranged in a lattice (output space) according to a given topology (often, grid or hexagonal) in order to visualize multi-dimensional inputs usually into a 2D format da Silva & Wunsch (2017); Faigl & Hollinger (2017). There are various research works using SOM for developing IDSs such as De la Hoz et al. (2014); Olszewski (2014); Obimbo & Jones (2012); Pachghare et al. (2009); Powers & He (2008); Kayacik & Zincir-Heywood (2006); Mitrokotsa & Douligeris (2005); Jirapummin et al. (2002). However, they are not fully taking into consideration benign outliers into modelling as well as not generating a comprehensive visualization driven by SOM.

The contribution of this paper is summarized in four objectives. The first objective is taking into consideration benign outliers into IDS design. The second objective is particularly improving SOM method in presence of benign outliers. The evaluation through an extensive implementation and analysis over 2D synthetic and multi-dimensional benchmarking datasets shows that the proposed method outperformed effectively several existing approaches in terms of the applied performance metrics. We firstly evaluated the proposed method over 2D synthetic datasets to observe that how the SOM lattice is adjusting among data points labelled as benign outlier, then into all input vectors (either benign or normal). The proposed method could successfully outperformed the standalone SOM and Fuzzy SOM techniques that have been widely applied into learning systems. Afterwards, we employed the proposed method over four IDS benchmarking datasets (NSL-KDD, UNSW-NB15, AAGM and VPN-nonVPN) as the principal reason for this contribution to be able to design a robust IDS. The third objective is considering and introducing new visualization capabilities for multi-class datasets by SOM without additional computational costs. Finally, we used a usability test to assess the proposed visualization-based IDS from experts' point of view in terms of the learnability and the satisfaction.

The rest of this paper is organized as follows. Section 2 discusses benign outliers. Section 3 presents the SOM algorithm. The proposed method is given in Section 4. Section 5 provides the details of experimental results over synthetic and real benchmarking datasets. The embedded visualization approach in the proposed method is critically considered in Sections 6 and 8, respectively. Section 7 presents the effectiveness of the proposed IDS method through usability test. Section 9 investigates the proposed method on the recent and custom network traffic datasets. Finally, Section 10 concludes the paper and discusses future work.

2. Benign Outliers

Outliers are data points that are distant from other data and may indicate experimental error, often resulting in exclusion from the data set. Outliers may occur due to several reasons, such as, measurement error, incidental systematic error, or by chance. It is often not trivial to ascertain the cause of an outlier, resulting in no straightforward way to express rules for their removal. For instance, a person with an IQ of 130 is not an outlier. Outliers may or may not be a problem depending on several factors Marr (2015):

- Some statistical tests are robust and can accommodate outliers, others may be severely influenced by outliers.
- Some data types will naturally contain extreme values which are entirely inherent.
- The presence of outliers may, in fact, be of interest.

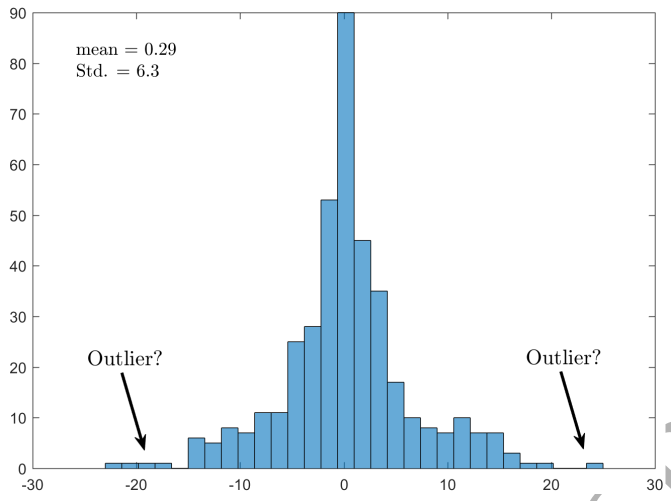


Figure 1: A sample of normal distribution

Figure 1 depicts a sample of data distribution with a set of data samples that are far from the 90%-95% of the normal distribution; however, they are not malignant or severe outliers. Hence, we cannot remove them because they are inherently a main part of the original data. In our research, we call them *Benign Outlier*, that removing them might result in inappropriate training, unstable and diverge data modelling. To be able to deal accurately with benign outliers, we would initially need to identify them. To do so, we employ Hotelling's T-squared distribution technique Yi et al. (2016).

2.1. Hotelling's T-squared distribution

The $Hotelling's\ T^2$ distribution is a multivariate generalization of the $Student\ t-test$. The form of the $Hotelling's\ T^2$ is as follows:

$$T^2 = (X - \bar{X})W^{-1}(X - \bar{X}) \tag{1}$$

where, X is the original data matrix, $\bar{X}$ is the mean of the dataset, and W is the covariance matrix of X. The $Hotelling's\ T^2$ statistic is approximately F-distributed as follows:

$$F_{p,n,\alpha} \sim T^2 \frac{(n-a)}{a(n-1)} \tag{2}$$

Any sample that has a F-value that exceeds the critical F-value can be considered as an outlier.

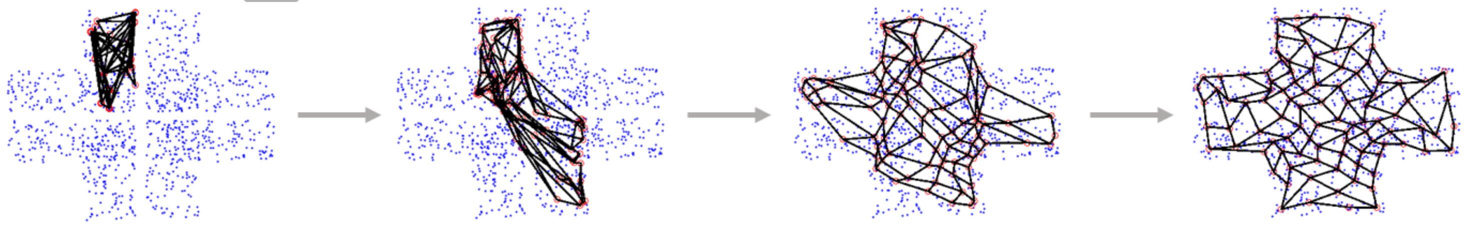


Figure 2: An illustration of the training of a self-organizing map

3. Self-Organizing Map (SOM)

Self-Organizing Map (SOM, also known as Kohonen network) Kohonen & Honkela (2007) is an unsupervised competitive learning algorithm. The goal of SOM is to convert a complex high-dimensional input data into a simpler low-dimensional (usually 2D) discrete map, building a topologically preserving map. Figure 2 illustrates an example of SOM training. At first (left) the SOM nodes are arbitrarily positioned in the data space. One SOM node which is nearest to a randomly picked training datum is selected. It is moved towards the training datum and absorbed its neighbours towards picked training datum on the grid (two middle illustrations). After many iterations the grid tends to approximate the data distribution (right). To do so, SOM includes four processes in a row: initialization, competition, cooperation, and adaptation.

- Initialization: all the connection weights are initialized with small random values.
- Competition: one training data is randomly selected and calculated a distance-based similarity (Often by Euclidean distance) to each neuron weight to find the Best Matching Unit (BMU) as the winner neuron.

$$\|X(t) - W_j(t)\| = \sqrt{\sum_{j=1}^{k} (X(t) - W_j(t))^2} \tag{3}$$

  where, at iteration t, $X(t)$ is the random selected data, $W(t)$ is all the neuron weights, and k is the number of SOM neurons.

- Cooperation: the closest neighbours of winner neuron are willing to relocate more than far neighbours alongside with winner neuron towards the data point $X(t)$. This is a topological neighbourhood that decays with distance. For the size of the neighbourhood, we employed the Gaussian function that shrinks on each iteration until eventually the neighbourhood is just the BMU itself, as follows:

$$T_{I(w),j} = exp(\frac{-d^2_{I(w),j}}{2\sigma^2}) \tag{4}$$

  where, $I(w)$ is the index of the winning neuron, j is the list of all neurons, $d_{I(w),j}$ is distance between winning neuron and all the neurons, and σ is the neighbourhood size which is decreased with time.
  In this paper, we apply a temporal scaling function to decrease the σ over time:

$$Value_i + Tfrac * (Value_f - Value_i); \tag{5}$$

  where, $Value_i$ is the initial value of σ, $Value_f$ is the final value of σ, $Tfrac$ is the time fraction which is calculated by $\frac{iter}{T}$, $iter$ is the current iteration of the running algorithm, and T is the maximum number of iteration.

- Adaptation: All the nuerons will be updated according to:

$$\Delta W_{ji} = \eta(t)\ T_{I(w),j}(t)\ (X(t) - W_j(t)) \tag{6}$$

  where, ηt is the learning rate which is decreased over time. We apply the same strategy given in Eq. 5.

The last three phases are repeated, until the maximum number of iterations is reached or the changes become smaller than a predefined threshold.

Algorithm 1 The pseudocode of the proposed visualization-based IDS (The time complexity is $O(N + 2T)$)

```
Input: Training dataset
Output: Well-separated clusters with visualization capabilities
 1: $N$; The number of training data
 2: $T$; The maximum number of iteration
 3: $Iter$; The counter
 4: Set initial parameters for SOM
    Phase 1: Identify Benign Outliers
 5: for ($iter = 1\ to\ N; iter++$) do
 6:     Identify $Benign.Outliers$ (Eqs. 1 and 2)
 7: end for
    Phase 2: Run SOM over Benign Outliers
 8: for ($iter = 1\ to\ T; iter++$) do
 9:     Run SOM algorithm (Section 3) over $Data_{Benign.Outliers}$
10: end for
    Phase 3: Run SOM based on Roulette Wheel (RW)
11: Define roulette wheel for both $Benign.Outliers$ and rested $normalities$
12: for ($iter = 1\ to\ T; iter++$) do
13:     Run SOM algorithm (Section 3) over picked data points by RW mechanism
14: end for
15: return Well-separated clusters
16: return Visualize 2D output of the designed multi-class IDS by SOM capabilities (A sample of visualizations is in Section 6).
```

4. The Proposed Detection Method for IDS Design

This section presents the details of the proposed detection method for IDS design using a modified Self-Organizing Map (SOM). The algorithm pseudocode is shown in Algorithm 1. Basically, the proposed training algorithm consists of three stages. The first stage is identification of benign outliers ($O(N)$). Second stage is running SOM over benign outliers ($O(T)$). Finally, the third stage is re-running SOM over those data picked by Roulette Wheel (RW) selection method ($O(T)$). With the RW-based data point selection, the SOM can search more in sparse area (low-frequent patterns or benign outliers) rather than select all data points randomly with the same selection probability. To do so, we give a pre-defined percentage of roulette space to those data labelled as benign outliers, and rested for normalities. For instance, if the predefined roulette size goes 60% for benign outlier data points (i.e., the entire 60% are equally divided between them), the rested 40% goes for rested normalities (i.e., the entire 40% portion are equally divided between them). Figure 3 illustrates the concept of the roulette wheel selection method. The time complexity of the proposed method is $O(N + 2T)$ which is linear and affordable. The proposed method consists of two phases: offline training and online detection. Algorithm 2 shows the proposed training method and Algorithm 3 shows the threshold-based detection technique for monitoring new data pattern (including known and unknown) which may not exist in the training phase. Once the optimal

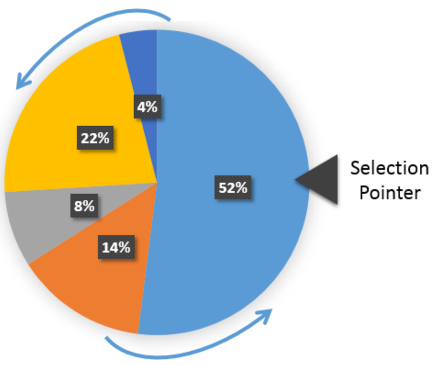


Figure 3: The concept of roulette wheel selection

placement of neurons (cluster centroids) from the training phase is constructed, they are sent to the detection phase for anomaly detection purposes when new monitoring data enters the system. If a new monitoring data is close to a normal cluster with less than a predefined threshold value (we selected $Threshold = 0.4$), we label it as normal traffic; otherwise in case it is bigger than the predefined threshold, it is considered as anomaly. Similarly, if a new monitoring data is close to a cluster labelled attack, it is considered as a known attack.

5. Experimental Results

The experiments have been firstly conducted on several synthetic datasets to show the effectiveness of the proposed method in 2D datasets. Furthermore, we applied four computer network intrusion detection benchmarking datasets as NSL-KDD NSL-KDD (2014), UNSW-NB15 Moustafa & Slay (2015), AAGM Lashkari et al. (2017), and VPN-nonVPN Draper-Gil et al. (2016) to assess the performance and robustness of the proposed method over real network data.

Table 1: Datasets' characteristics

Data set	classes	size
Crescent Full Moon	4	1200
Half Kernel	4	1000
Pin Wheel	5	1000
Aggregation	7	788
Compound	6	399
D31	31	3100
Flame	2	240
Jain	2	373

5.1. Synthetic datasets

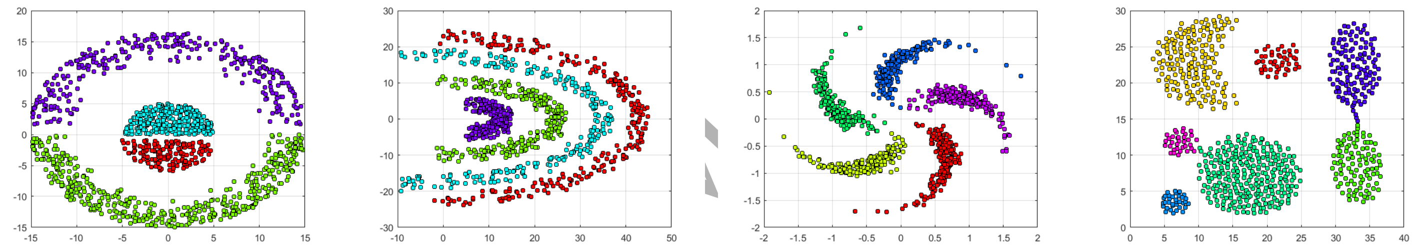
To assess the performance of the proposed method, we firstly applied four 2D artificial data sets from Karami & Johansson (2014a) and rested four from Fränti (2015) which include rare and low-frequent patterns as well as very close data samples

Algorithm 2 Training phase of the proposed detection method for IDS

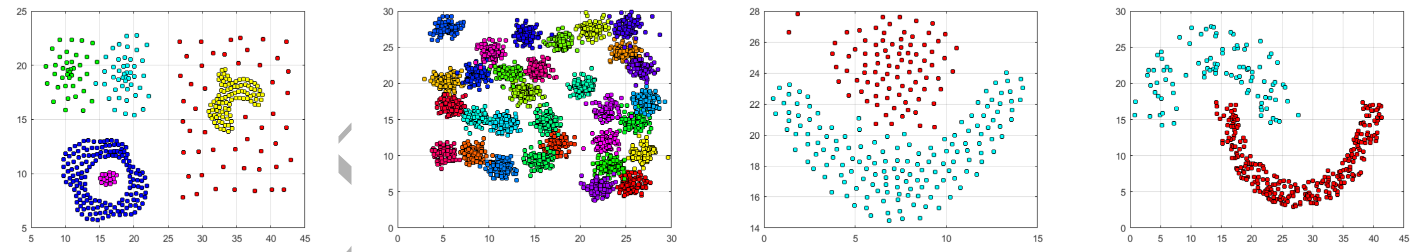
```
Input: Training dataset
Output: Well-separated cluster centroids
    STEP 1: Initialization:
 1: Load dataset for training, $N =$ the number of input data
 2: Initialize lattice (2D) and neurons' weight, $T$ is the number of iteration
 3: Declare initial learning rate ($\mu_i$) and final learning rate ($\mu_f$)
 4: Declare initial Sigma (attraction rate between points) ($\sigma_i$) and final Sigma ($\sigma_f$)
 5: Define the roulette size between [0 100] ($RouletteSize_{Benign.Outliers}$) for $Benign.Outliers$, while rested goes for normalities ($RouletteSize_{Normalities}$)
    STEP 2: Identify Benign Outliers:
 6: Declare $F$-value as threshold for outlier detection
 7: for ($iter = 1$ to $N; iter++$) do
 8:     Identify $Benign.Outliers$ by Eqs. 1 and 2
 9: end for
    STEP 3: Run SOM over Benign Outliers:
10: for ($iter = 1$ to $T; iter++$) do
11:     Run Competition phase by Eq. 3
12:     Scale $\mu$ and $\sigma$ by Eq. 5
13:     Run Cooperation phase by Eq. 4
14:     Run Adaptation phase by Eq. 6
15: end for
    STEP 4: Run SOM over all data points:
16: for ($iter = 1$ to $T; iter++$) do
17:     Pick randomly a data point by Roulette Wheel (Fig. 3)
18:     Run Competition phase by Eq. 3
19:     Scale $\mu$ and $\sigma$ by Eq. 5
20:     Run Cooperation phase by Eq. 4
21:     Run Adaptation phase by Eq. 6
22: end for
23: return Well-separated cluster centroids with visualization capabilities
```

from different classes. The characteristics of the artificial data sets are summarized in Table 1 and depicted in Figure 4. All the experiments were performed on Intel(R) Core(TM) i5-6300HQ CPU 2.30 GHz with 16 GB RAM on the platform Microsoft Windows 10. We have conducted this experiment in MATLAB R2016b. To compare the performance of the proposed algorithm with some pre-existing algorithms, we used several metrics, such as Detection Rate (DR), False Positive Rate (FPR), Mean of errors

```
Algorithm 3 Detection phase for new monitoring data
Input: Cluster centroids from the training phase
Output: Data type detection, either normal, attack or anomaly
  $Threshold$ = The threshold value for anomaly detection
  while monitoring new data $Data_{New}$ do
    if distance $Data_{New}$ is closer to one of the attack labelled clusters then
      label $Data_{New}$ as ATTACK_CLASS
    else if distance is smaller than $Threshold$ to a normal cluster then
      label $Data_{New}$ as NORMAL
    else
      label $Data_{New}$ as ANOMALY
    end if
  end while
  return Detecting three types of data: normal, anomaly or multi-class attack
```



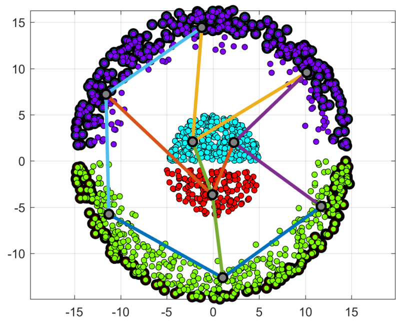
(a) Crescent Full Moon (b) Half Kernel (c) Pin Wheel (d) Aggregation

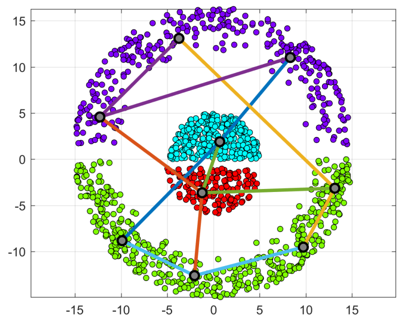


(e) Compound (f) D31 (g) Flame (h) Jain's toy problem

Figure 4: Synthetic datasets from Karami & Johansson (2014a) and Fränti (2015)

between data target and training output, Standard Deviation (Std.) of the errors, Confidence Interval (CI 95%), and Topographic Error (TE). TE is a quantitative measure of mapping quality. The TE gives the percentage of the data vectors for which the first BMU and the second BMU are not neighbouring units. Lower TE values indicate better mapping quality.


(a) Proposed method


(b) Fuzzy SOM

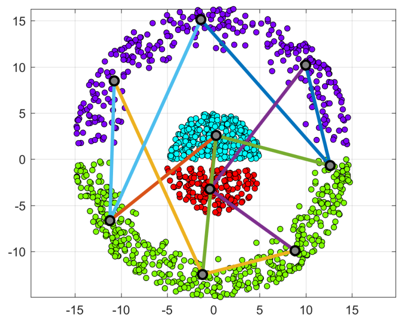
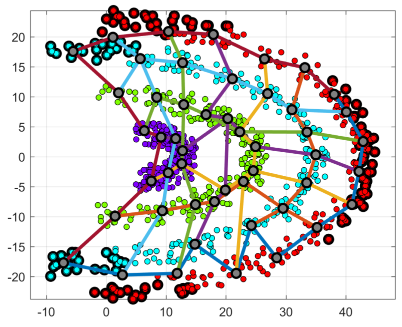
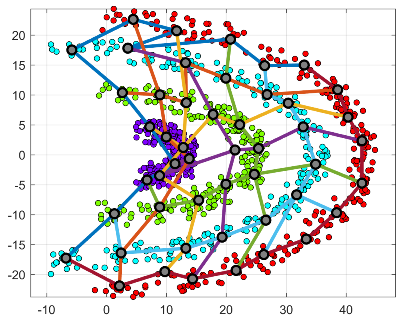

(c) Standalone SOM

Figure 5: The Lattice of three training algorithms over Crescent Full Moon dataset

Table 2: The results for Crescent Full Moon dataset

Lattice	Methods	DR (%)	FPR (%)	Mean	Std.	CI (95%)	TE
3x3	Proposed Method	98.61	0.96	3.074	1.776	[2.973, 3.174]	0.074
	Fuzzy SOM	91.20	10.13	3.105	1.636	[3.198, 3.105]	0.12
	SOM	90.04	8.94	3.129	1.671	[3.035, 3.224]	0.14
4x4	Proposed Method	98.99	0.61	1.962	1.031	[1.962, 2.079]	0.042
	Fuzzy SOM	95.45	1.50	2.122	1.164	[2.056, 2.188]	0.081
	SOM	93.88	1.84	2.143	1.183	[2.076, 2.21]	0.092
5x5	Proposed Method	99.45	0.11	1.576	0.905	[1.525, 1.627]	0.012
	Fuzzy SOM	98.88	0.36	1.654	0.891	[1.603, 1.704]	0.028
	SOM	99.00	0.26	1.572	0.772	[1.529, 1.616]	0.021


(a) Proposed Method


(b) Fuzzy SOM

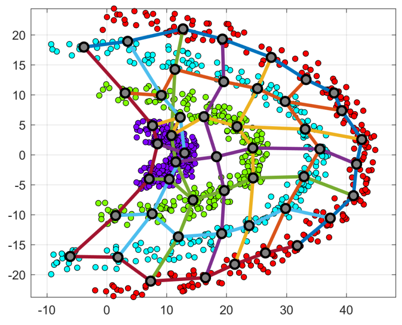

(c) Standalone SOM

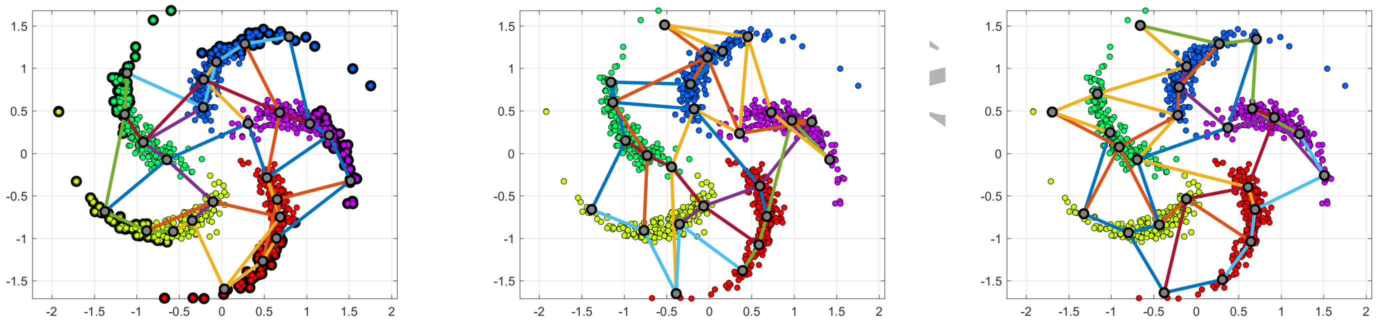
Figure 6: The Lattice of three training algorithms over Half Kernel dataset

5.1.1. *The experimental results over synthetic data*

Figures 5-12 demonstrates the results of SOM lattice adjustment towards the input vectors. The lattice visualization clearly shows how the SOM nodes (neurons) are positioned into the 2D output according to three applied algorithms. Figures 5a-12a

Table 3: The results for Half Kernel dataset

Lattice	Methods	DR (%)	FPR (%)	Mean	Std.	CI (95%)	TE
6x6	Proposed Method	94.77	1.53	2.194	1.29	[2.114, 2.274]	0.072
	Fuzzy SOM	92.66	2.38	2.196	1.201	[2.121, 2.27]	0.089
	SOM	92.31	1.84	2.254	1.221	[2.178, 2.33]	0.076
7x7	Proposed Method	97.57	0.80	1.75	0.984	[1.689, 1.811]	0.051
	Fuzzy SOM	97.02	0.84	1.81	0.99	[1.749, 1.872]	0.055
	SOM	93.12	2.01	1.924	1.085	[1.857, 1.992]	0.056
8x8	Proposed Method	99.02	0.37	1.45	0.799	[1.401, 1.5]	0.042
	Fuzzy SOM	98.95	0.41	1.505	0.807	[1.455, 1.555]	0.066
	SOM	98.18	0.77	1.491	0.827	[1.439, 1.542]	0.071



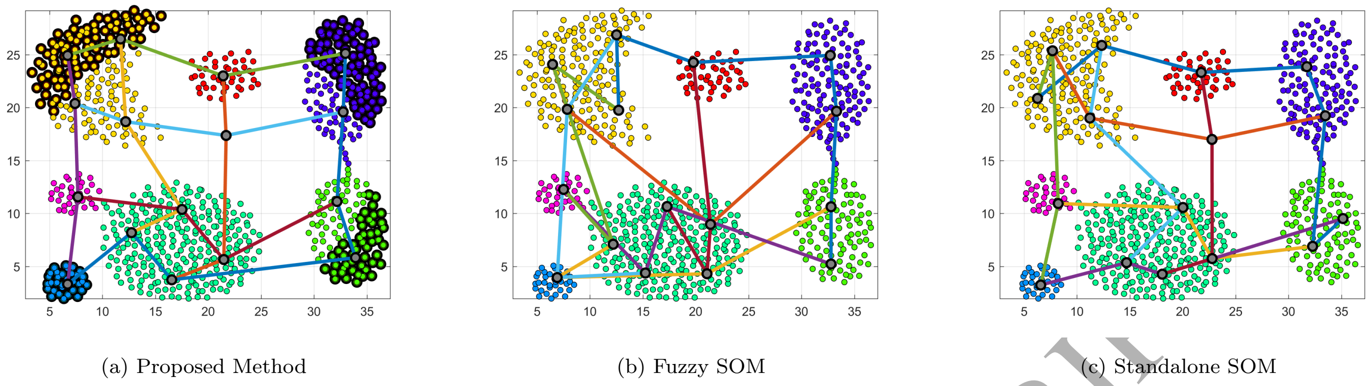
(a) Proposed Method (b) Fuzzy SOM (c) Standalone SOM

Figure 7: The Lattice of three training algorithms over Pin Wheel dataset

Table 4: The results for Pin Wheel dataset

Lattice	Methods	DR (%)	FPR (%)	Mean	Std.	CI (95%)	TE
5x5	Proposed Method	99.41	0.28	0.124	0.081	[0.119, 0.129]	0.017
	Fuzzy SOM	98.38	0.31	0.137	0.082	[0.131, 0.142]	0.048
	SOM	96.67	0.32	0.134	0.077	[0.129, 0.138]	0.05
6x6	Proposed Method	99.48	0.09	0.101	0.065	[0.097, 0.105]	0.013
	Fuzzy SOM	99.11	0.16	0.104	0.061	[0.1, 0.108]	0.042
	SOM	98.83	0.09	0.12	0.059	[0.103, 0.11]	0.05
7x7	Proposed Method	99.83	0.042	0.088	0.054	[0.085, 0.091]	0.011
	Fuzzy SOM	99.71	0.052	0.089	0.053	[0.086, 0.093]	0.039
	SOM	99.68	0.049	0.086	0.051	[0.084, 0.09]	0.047

shows the results of the proposed algorithm based on finding benign outliers. The bold and highlighted input vectors are the benign outliers that the modified SOM described in Section 4 firstly distributes SOM nodes across them, afterwards it adjusts SOM nodes and lattice over normalities together with benign outliers with the incorporated Roulette Wheel (RW) mechanism. As shown in Figures 5-12, the proposed method provides better lattice adjustment with a few overlapped connections among neighbours and well-separated nodes in the sensitive spots (highlighted inputs). The obtained results in terms of the connections between nodes and

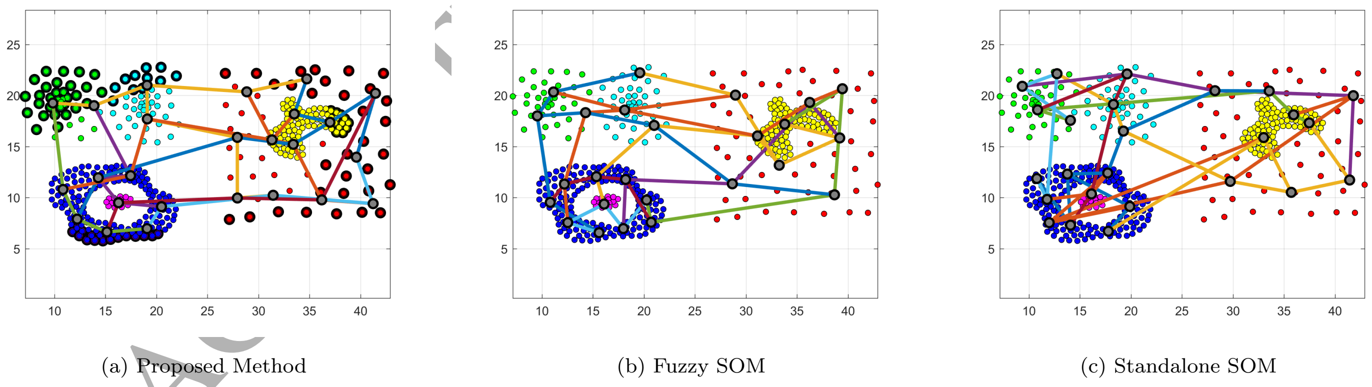


(a) Proposed Method (b) Fuzzy SOM (c) Standalone SOM

Figure 8: The Lattice of three training algorithms over Aggregation dataset

Table 5: The results for Aggregation dataset

Lattice	Methods	DR (%)	FPR (%)	Mean	Std.	CI (95%)	TE
4x4	Proposed Method	99.54	0.19	2.159	0.921	[2.095, 2.224]	0.056
	Fuzzy SOM	99.39	0.22	2.224	0.942	[2.158, 2.29]	0.089
	SOM	98.3	0.91	2.267	1.051	[2.193, 2.34]	0.094
5x5	Proposed Method	99.77	0.19	1.66	0.73	[1.609, 1.711]	0.068
	Fuzzy SOM	99.65	0.22	1.724	0.788	[1.669, 1.779]	0.081
	SOM	99.37	0.27	1.724	0.76	[1.671, 1.777]	0.086
6x6	Proposed Method	99.94	0.004	1.371	0.549	[1.333, 1.41]	0.043
	Fuzzy SOM	99.56	0.084	1.424	0.657	[1.378, 1.47]	0.055
	SOM	99.63	0.11	1.429	0.633	[1.384, 1.473]	0.06



(a) Proposed Method (b) Fuzzy SOM (c) Standalone SOM

Figure 9: The Lattice of three training algorithms over Compound dataset

their neighbours from two other methods are somehow disordered, because they have no preliminary knowledge of benign outliers to be able to perform well. Tables 2-9 show the best result out of 10 times independent running algorithms with different lattice sizes. The proposed method significantly outperformed fuzzy SOM and standalone SOM in terms of the higher detection rate,

Table 6: The results for Compound dataset

Lattice	Methods	DR (%)	FPR (%)	Mean	Std.	CI (95%)	TE
5x5	Proposed Method	97.78	1.12	1.345	0.692	[1.277, 1.413]	0.072
	Fuzzy SOM	94.65	1.25	1.345	0.0775	[1.269, 1.421]	0.091
	SOM	95.94	1.98	1.381	0.748	[1.308, 1.455]	0.086
6x6	Proposed Method	98.24	0.59	1.096	0.555	[1.041, 1.15]	0.6
	Fuzzy SOM	97.18	0.99	1.085	0.603	[1.026, 1.144]	0.79
	SOM	98.19	1.23	1.123	0.6	[1.064, 1.182]	0.88
7x7	Proposed Method	99.27	0.23	0.936	0.487	[0.889, 0.984]	0.039
	Fuzzy SOM	98.81	0.64	0.904	0.501	[0.855, 0.953]	0.048
	SOM	98.87	0.31	0.956	0.533	[0.903, 1.008]	0.5

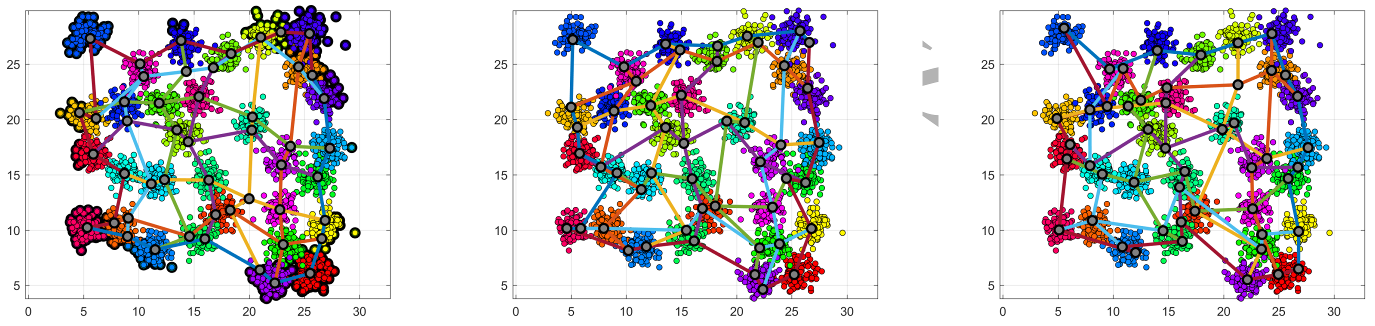

(a) Proposed Method (b) Fuzzy SOM (c) Standalone SOM

Figure 10: The Lattice of three training algorithms over D31 dataset

Table 7: The results for D31 dataset

Lattice	Methods	DR (%)	FPR (%)	Mean	Std.	CI (95%)	TE
6x6	Proposed Method	96.71	0.96	0.971	0.535	[0.952, 0.99]	0.101
	Fuzzy SOM	96.67	1.16	0.921	0.482	[0.904, 0.938]	0.142
	SOM	95.52	1.39	0.972	0.509	[0.954, 0.99]	0.149
7x7	Proposed Method	93.65	1.74	0.864	0.437	[0.848, 0.879]	0.087
	Fuzzy SOM	92.98	2.99	0.897	0.476	[0.88, 0.914]	0.096
	SOM	91.78	2.88	0.905	0.468	[0.888, 0.921]	0.099
8x8	Proposed Method	95.87	1.81	0.77	0.41	[0.756, 0.785]	0.097
	Fuzzy SOM	95.08	2.62	0.75	0.397	[0.736, 0.764]	0.115
	SOM	94.51	1.97	0.796	0.417	[0.781, 0.811]	0.129

the lower false positive rate, the better mapping quality (lower TE), and the better estimation of 95% confidence interval at the same time. Figures 13a-13h demonstrate the average MSE from 10 times independently run over applied datasets. The proposed method starts off with more attention to some particular data points (i.e., benign outliers) which are normally far from densities. As predicted, the MSE value of the proposed method within preliminary iterations are higher than other methods. Finally in the last iterations, the proposed method could successfully outperform fuzzy SOM and standalone SOM in terms of the MSE. It can

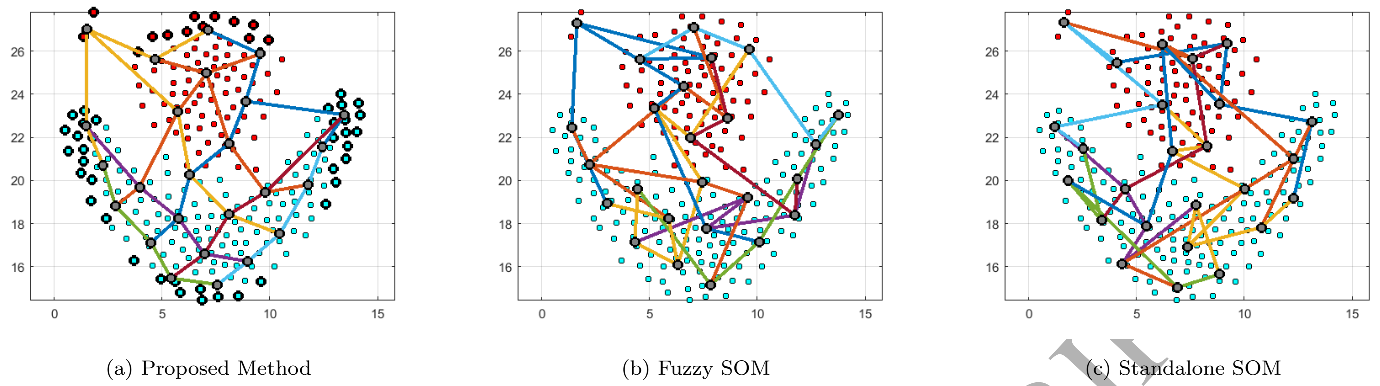

(a) Proposed Method (b) Fuzzy SOM (c) Standalone SOM

Figure 11: The Lattice of three training algorithms over Flame dataset

Table 8: The results for Flame dataset

Lattice	Methods	DR (%)	FPR (%)	Mean	Std.	CI (95%)	TE
3x3	Proposed Method	99.64	0.37	1.348	0.602	[1.271, 1.424]	0.033
	Fuzzy SOM	95.66	2.55	1.424	0.62	[1.346, 1.503]	0.058
	SOM	97.22	0.79	1.443	0.643	[1.361, 1.524]	0.069
4x4	Proposed Method	99.1	0.55	1.406	0.413	[0.988, 1.092]	0.028
	Fuzzy SOM	98.47	0.83	1.094	0.491	[1.032, 1.156]	0.044
	SOM	96.26	0.98	1.091	0.466	[1.032, 1.15]	0.049
5x5	Proposed Method	98.04	0.87	0.814	0.331	[0.773, 0.856]	0.021
	Fuzzy SOM	96.41	0.91	0.83	0.318	[0.79, 0.87]	0.046
	SOM	97.27	0.94	0.845	0.35	[0.801, 0.889]	0.049

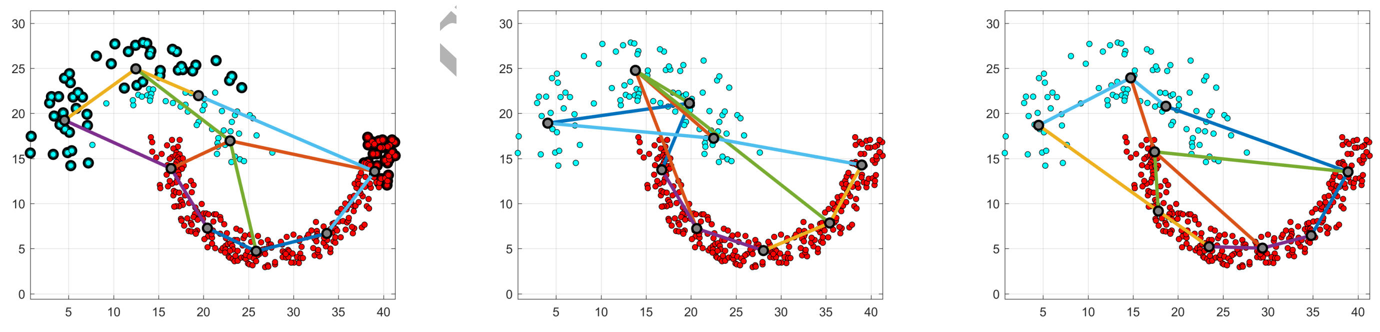

(a) Proposed Method (b) Fuzzy SOM (c) Standalone SOM

Figure 12: The Lattice of three training algorithms over Jain dataset

confirm that the proposed method has knowingly and accurately considered all data points with different degrees of importance.

Table 9: The results for Jain dataset

Lattice	Methods	DR (%)	FPR (%)	Mean	Std.	CI (95%)	TE
2x2	Proposed Method	91.1	4.69	4.438	2.167	[4.218, 4.658]	0.45
	Fuzzy SOM	89.83	6.17	4.529	2.203	[4.305, 4.752]	0.67
	SOM	87.62	13.67	4.766	2.356	[4.527, 5.005]	0.71
3x3	Proposed Method	100.00	0.00	2.391	1.097	[2.279, 2.502]	0.009
	Fuzzy SOM	100.00	0.00	2.567	1.253	[2.44, 2.695]	0.011
	SOM	99.01	0.37	2.417	1.372	[2.277, 2.556]	0.024
4x4	Proposed Method	100.00	0.00	1.664	0.819	[1.581, 1.748]	0.012
	Fuzzy SOM	100.00	0.00	1.664	0.821	[1.591, 1.757]	0.018
	SOM	100.00	0.00	1.72	0.872	[1.632, 1.809]	0.021

5.2. NSL-KDD dataset for intrusion detection

NSL-KDD data set is publicly used for intrusion detection purposes Tavallaee et al. (2009); NSL-KDD (2014). Table 10 shows the characteristics of the NSL-KDD. It is derived from KDDCUP'99 dataset that suffers from redundant data, which causes the learning algorithm to be biased towards the frequent patterns. The NSL-KDD dataset includes a training set (125,973 records) and a testing set (22,544 records). It includes 41 attributes with one normal data and twenty-four attack data samples, which is shown in details in Table 11. These attack data samples are divided into four major classes including Denial-of-Service (DoS), unauthorized access to local supervisor privileges (U2R), unauthorized access from a remote machine (R2L), and scanning network to find known vulnerabilities (Probing). The details of attack types are shown in Table 12.

5.3. UNSW-NB15 dataset for intrusion detection

The KDDCup'99 and NSL-KDD datasets have been commonly used for setting up and evaluating intrusion detection systems; however, they are relatively old datasets and many of normal and attack traffic have been obviously changed during the last decade. To be able to verify the proposed method with a new benchmarking IDS dataset, we employed a new IDS dataset called UNSW-NB15 created by the IXIA PerfectStorm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) for generating a hybrid of real modern normal activities and synthetic contemporary attack behaviours Moustafa & Slay (2015). The summary of the UNSW-NB15 dataset is provided in Table 13. This dataset has nine types of attacks namely, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms. This dataset contains 49 features and around 2,540,044 data instances which is shown in details in Table 14. There are a subset of training and testing sets, which contain 175,341 and 82,332 records, respectively. This dataset is quite different from KDDCup and NSL-KDD, which reflects a more contemporary and complex threat environment. For instance, this new dataset introduces the increased number of attack types and highly imbalanced records that present a significant challenge for hybrid intelligent algorithms designed specifically for intrusion detection and prevention.

Table 11: The 41 features of NSL-KDD dataset

ID	Feature name	Data type	Description
1	duration	numeric	Length of the connection
2	protocol_type	nominal	Connection protocol
3	service	nominal	Destination service
4	flag	nominal	Status flag of the connection
5	src_bytes	numeric	Bytes send from source to destination
6	dst_bytes	numeric	Bytes sent from destination to source
7	land	binary	1 if is from/to the same host/port; otherwise 0
8	wrong_fragment	numeric	Number of wrong fragment
9	urgent	numeric	Number of urgent packets
10	hot	numeric	Number of hot indicators
11	num_failed_logins	numeric	Number of failed login in attempts
12	logged_in	binary	1 if successful logged in; otherwise 0
13	num_compromised	numeric	Number of compromised conditions
14	root_shell	binary	1 if root shell is obtained; otherwise 0
15	su_attempted	binary	1 if *su root* command attempted; otherwise 0
16	num_root	numeric	Number of root accesses
17	num_file_creations	numeric	Number of file creation operations
18	num_shells	numeric	Number of shell prompts
19	num_access_files	numeric	Number of operations on access control files
20	num_outbound_cmds	numeric	Number of outbound commands in an ftp session
21	is_host_login	binary	1 if the login belongs to the host list; otherwise 0
22	is_guest_login	binary	1 if the login is a guest login; otherwise 0
23	count	numeric	Number of connections to the same host as the current connection in the past two seconds
24	srv_count	numeric	Number of connections to the same service as the current connection in the past two seconds
25	serror_rate	numeric	% of connections that have SYN error (same-host connections)
26	srv_serror_rate	numeric	% of connections that have SYN error (same-service connections)
27	rerror_rate	numeric	% of connections that have REJ error (same-host connections)
28	srv_rerror_rate	numeric	% of connections that have REJ error (same-service connections)
29	same_srv_rate	numeric	% of connections to the same service (same-service connections)
30	diff_srv_rate	numeric	% of connections to the different services
31	srv_diff_host_rate	numeric	% of connections to the different hosts (same-service connections)
32	dst_host_count	numeric	% Count of connections having the same destination host
33	dst_host_srv_count	numeric	Count of connections having the same destination host and using the same service
34	dst_host_same_srv_rate	numeric	% of connections having the same destination host and using the same service
35	dst_host_diff_srv_rate	numeric	% of different services on the current host
36	dst_host_same_src_port_rate	numeric	% of connections to the current host having the ame port
37	dst_host_srv_diff_host_rate	numeric	% of connections to the same service coming from different hosts
38	dst_host_serror_rate	numeric	% of connection to the current host that have a SO error
39	dst_host_srv_serror_rate	numeric	% of connection to the current host and specified service that have a SO error
40	dst_host_rerror_rate	numeric	% of connections to the current host that have a RST error
41	dst_host_srv_rerror_rate	numeric	% of connection to the current host and specified service that have a RST error

Table 12: Four attack classes in the NSL-KDD intrusion detection dataset

Classes	Intrusion types
DoS	back, land, neptune, pod, smurf, teardrop, apache2*,udpstorm*, mailbomb*, processtable*
U2R	buffer_overflow, loadmodule, perl, rootkit, sqlattack*, mscan*, httptunnel*, ps*, xterm*
R2L	ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient, warezmaster, snmpgetattack*, snmpguess*, named*, sendmail*, worm*, xlock*, xsnoop*
Probe	ipsweep, nmap, portsweep, satan, saint*
	* unknown attacks available in testing dataset as anomalies for training dataset

Table 13: The data types in UNSW-NB15

Traffic type	Training set	Testing set
	No. of records	No. of records
Normal	56000	37000
Generic	40000	18871
Exploits	33393	11132
Fuzzers	18184	6062
DoS	12264	4089
Reconnaissance	10491	3496
Analysis	2000	677
Backdoor	1746	583
Shellcode	1133	378
Worms	130	44
Total:	175,341	82,332

5.4. Performance metrics

To assess the effectiveness and the efficiency of the proposed method, we use several performance criteria as follows:
The detection rate is the number of intrusions detected by the system (it is also called sensitivity or True Positive Rate (TPR)):

$$DR\,(Recall) = \frac{TP}{TP + FN} \tag{7}$$

The false positive rate is the number of normal traffic that was incorrectly classified as intrusion:

$$FPR = \frac{FP}{FP + TN} \tag{8}$$

Precision or Positive Predictive Value (PPV) measures the exactness and quality of the correct prediction:

$$Precision = \frac{TP}{TP + FP} \tag{9}$$

F-measure (or F1) is the weighted harmonic mean of precision (positive predictive value) in Eq. 9 and recall (detection rate) in Eq. 7:

$$F1 - Score = 2 \times \frac{Precision\ \times\ Recall}{Precision + Recall} \tag{10}$$

Accuracy (ACC) is the proportion of the true results (True Positives (TP) and True Negatives (TN)) among the total number of cases examined.

$$Accuracy = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

5.5. The experimental results on NSL-KDD dataset

We randomly sampled the NSL-KDD dataset to create two subsets for training and testing. The training set is approximately 20% of the original training instances. We trained the derived dataset 10 times independently to be able to assess the performance of the proposed method. The training accuracy with and without taking benign outliers into consideration is summarized in Table 15. This table shows the number of benign outliers per class. According to the proposed algorithm in Section 4, those data samples labelled as benign outlier are initially trained for the preliminary SOM lattice placement, then rested data come into the training phase by the RW selection strategy to improve the training accuracy. The obtained results from Table 15 show that

Table 14: The 49 features of UNSW-NB15 dataset

No.	Name	Type	Description
1	srcip	nominal	Source IP address
2	sport	integer	Source port number
3	dstip	nominal	Destination IP address
4	dsport	integer	Destination port number
5	proto	nominal	Transaction protocol
6	state	nominal	"Indicates to the state and its dependent protocol, e.g. ACC, CLO, CON, ECO, ECR, FIN, INT, MAS, PAR, REQ, RST, TST, TXD, URH, URN and
7	dur	Float	Record total duration
8	sbytes	Integer	Source to destination transaction bytes
9	dbytes	Integer	Destination to source transaction bytes
10	sttl	Integer	Source to destination time to live value
11	dttl	Integer	Destination to source time to live value
12	sloss	Integer	Source packets retransmitted or dropped
13	dloss	Integer	Destination packets retransmitted or dropped
14	service	nominal	"http, ftp, smtp, ssh, dns, ftp-data, irc and (-) if not much used service"
15	Sload	Float	Source bits per second
16	Dload	Float	Destination bits per second
17	Spkts	integer	Source to destination packet count
18	Dpkts	integer	Destination to source packet count
19	swin	integer	Source TCP window advertisement value
20	dwin	integer	Destination TCP window advertisement value
21	stcpb	integer	Source TCP base sequence number
22	dtcpb	integer	Destination TCP base sequence number
23	smeansz	integer	Mean of the packet size transmitted by the src
24	dmeansz	integer	Mean of the packet size transmitted by the dst
25	trans_depth	integer	Represents the pipelined depth into the connection of http request/response transaction
26	res_bdy_len	integer	Actual uncompressed content size of the data transferred from the servers http service.
27	Sjit	Float	Source jitter (mSec)
28	Djit	Float	Destination jitter (mSec)
29	Stime	Timestamp	record start time
30	Ltime	Timestamp	record last time
31	Sintpkt	Float	Source interpacket arrival time (mSec)
32	Dintpkt	Float	Destination interpacket arrival time (mSec)
33	tcprtt	Float	"TCP connection setup round-trip time, the sum of 'synack' and 'ackdat'."
34	synack	Float	"TCP connection setup time, the time between the SYN and the SYN_ACK packets."
35	ackdat	Float	"TCP connection setup time, the time between the SYN_ACK and the ACK packets."
36	is_sm_ips_ports	Binary	"If source (1) and destination (3)IP addresses equal and port numbers (2)(4) equal then, this variable takes value 1 else 0"
37	ct_state_ttl	Integer	No. for each state (6) according to specific range of values for source/destination time to live (10) (11).
38	ct_flw_http_mthd	Integer	No. of flows that has methods such as Get and Post in http service.
39	is_ftp_login	Binary	If the ftp session is accessed by user and password then 1 else 0.
40	ct_ftp_cmd	integer	No of flows that has a command in ftp session.
41	ct_srv_src	integer	No. of connections that contain the same service (14) and source address (1) in 100 connections according to the last time (26).
42	ct_srv_dst	integer	No. of connections that contain the same service (14) and destination address (3) in 100 connections according to the last time (26).
43	ct_dst_ltm	integer	No. of connections of the same destination address (3) in 100 connections according to the last time (26).
44	ct_src_ltm	integer	No. of connections of the same source address (1) in 100 connections according to the last time (26).
45	ct_src_dport_ltm	integer	No of connections of the same source address (1) and the destination port (4) in 100 connections according to the last time (26).
46	ct_dst_sport_ltm	integer	No of connections of the same destination address (3) and the source port (2) in 100 connections according to the last time (26).
47	ct_dst_src_ltm	integer	No of connections of the same source (1) and the destination (3) address in in 100 connections according to the last time (26).
48	attack_cat	nominal	"The name of each attack category. In this data set, nine categories e.g. Fuzzers, Analysis, Backdoors, DoS Exploits, Generic, Reconnaissance, Shellco
49	Label	binary	0 for normal and 1 for attack records

U2R and Probing classes have the highest representations of benign outliers by 78.43% and 72.24% (for F-Value=30), respectively. In contrast, R2L and DoS classes have the lowest benign outliers by 23.12% and 11.29%, respectively. On the other hand, by decreasing the sensitivity of benign outlier threshold to 35 (F-Value=35), Probing and U2R classes have the highest rate of benign outliers by 96.19% and 82.36%, respectively. It confirms that Probing and U2R data samples are the most important bunches of benign outliers that are mostly out of 90%-95% of the main distribution. R2L and DoS classes by 23.12% and 11.29% have much dissimilarities to benign outliers.

Table 16 illustrates the confusion matrix for the training phase based on the recall and the precision metrics per class. Table 17 presents the comparative results of recall and precision for various studies. As seen in this table the proposed method outperformed the other methods in the most cases. The only drawback of the proposed method based on the recall is its performance for DoS

Table 15: The training accuracy of IDS with (without) Benign Outliers Consideration in NSL-KDD

			The number of benign outliers per class					
Lattice	F-Value	Benign Outliers	Normal	DoS	U2R	R2L	Probing	Accuracy
20 × 20								99.49 (92.19)
			3229	1037	40	184	1684	
	30	23.87%	23.97%	11.29%	78.43%	23.12%	72.24%	
18 × 18								98.95 (90.15)
20 × 20								97.88 (91.03)
			4150	1558	42	194	2018	
	20	34.68%	34.24%	18.85%	82.36%	21.65%	96.19%	
18 × 18								95.19 (90.44)

and R2L classes where their values are a little less than Ramp-KSVCR algorithm. In terms of the precision results, the proposed method clearly outperformed other methods, except for Normal and R2L classes that are a little less than Ramp-KSVCR algorithm. However, the proposed method scores over these methods in the majority of cases. The additional important comparison is given in Table 18 based on the total accuracy, detection rate (DR), false alarm rate (FPR) and F1-Score. It can be seen that the applied performance metrics on the proposed method is higher than several existing algorithms except for overall DR that is in the second rank. It can be concluded that the proposed method is better considering its performance levels across both attack and non-attack classes and the advantages incurred by discovering the number of data in each class acting as benign outlier that degrade considerably the training accuracy.

Table 16: Confusion Matrix of training process for NSL-KDD

	Normal	DoS	U2R	R2L	Probing	Recall (%)
Normal	7083	6	0	6	7	99.73
DoS	67	4868	0	0	3	98.58
U2R	5	1	45	5	0	80.36
R2L	19	0	0	94	0	83.19
Probing	46	6	0	0	1180	95.78
Precision (%)	98.10	99.73	100	89.52	99.16	

Table 17: The training results on NSL-KDD based on each class

Method		Normal	DoS	U2R	R2L	Probing
Proposed method (20 × 20)	Recall (%)	**99.73**	98.58	**80.36**	83.19	**95.78**
	Precision (%)	98.10	**99.73**	**100**	89.52	**99.16**
Proposed method (18 × 18)	Recall (%)	98.11	98.19	69.54	79.57	94.79
	Precision (%)	97.20	99.02	90.88	77.96	95.10
Fuzzy SOM (20 × 20) Karami & Guerrero-Zapata (2014)	Recall (%)	96.19	96.34	63.89	82.12	94.45
	Precision (%)	95.67	98.89	87.11	80.14	96.21
SOM (20 × 20)	Recall (%)	94.44	93.19	59.34	77.19	89.45
	Precision (%)	93.12	91.02	80.87	70.18	89.16
Ramp-KSVCR Bamakan et al. (2017)	Recall (%)	99.14	**99.49**	68.75	**91.09**	93.58
	Precision (%)	**98.69**	98.94	86.84	**90.64**	98.31

Table 18: The results of overall training performance (%) on NSL-KDD

Method	Accuracy	DR	FPR	F1-Score
Proposed method (20 × 20)	**99.49**	91.53	**0.7**	**92.32**
Proposed method (18 × 18)	97.88	88.04	0.92	89.62
Fuzzy SOM (20 × 20) Karami & Guerrero-Zapata (2014)	95.87	86.60	3.18	88.43
SOM (20 × 20)	91.72	82.72	5.73	83.47
Ramp-KSVCR Bamakan et al. (2017)	98.68	90.41	0.86	N/A
LMDRT-SVM2 Wang et al. (2017)	98.47	**99.85**	2.96	N/A
HG-GA SVM Raman et al. (2017)	96.72	97.14	0.83	N/A

5.6. The experimental results on UNSW-NB15 dataset

We randomly sampled the UNSW-NB15 dataset to create two subsets for training and testing processes. The training set is approximately 15% of the original training instances. This configuration is applied for each of 9 classes except for the "Worms" class, where all the instances were used for training. The reason is the less number of Worms instances in the original training dataset. We trained the derived dataset 10 times independently to be able to assess the performance of the proposed method. The training accuracy with and without taking benign outliers into consideration is summarized in Table 19. This table clearly shows the number of benign outliers per class. According to the proposed algorithm, the labelled data as benign outliers are initially trained for the preliminary lattice placement (i.e., the second and the third steps in Algorithm 2), then rested data come into the training phase (i.e., the fourth step in Algorithm 2) by the RW selection strategy to improve the training accuracy. The obtained results from Table 19 show that Exploits and Generic classes have the highest representations of benign outliers by 84.64% and 72.92% (for F-Value=30), respectively. In contrast, Reconnaissance and Shellcode classes have the lowest benign outliers by 10.10% and 9%, respectively. On the other hand, by decreasing the sensitivity of benign outlier threshold to 35 (F-Value=35), Exploits and Normal classes have the highest rate as benign outliers by 70.38% and 58.10%, respectively. Reconnaissance and Shellcode classes by 7.85% and 7.71% have much dissimilarities to benign outliers. Consequently, Exploits, Generic and Normal classes have the highest degree of importance on the training accuracy; however, Reconnaissance and Shellcode classes effects on the training accuracy with the lowest degree of importance. On the other hand, a few attack classes such as Fuzzers, Reconnaissance, and DoS with the largest number of data for training have had the lowest number of benign outliers that confirm these classes do not degrade significantly training accuracy.

Table 19: The training accuracy of IDS with (without) Benign Outliers Consideration in UNSW-NB15

			The number of benign outliers per class										
Lattice	F-Value	Benign Outliers	Normal	Backdoor	Analysis	Fuzzers	Shellcode	Reconn.	Exploits	DoS	Worms	Generic	Accuracy
25 × 25													98.26 (89.91
			1008	55	87	1067	26	265	2220	451	34	875	
	30	26.7%	60%	12.59%	17.40%	23.49%	9%	10.10%	84.64%	15.77%	26.15%	72.92%	
20 × 20													92.54 (89.74
25 × 25													94.12 (89.02
			976	50	79	991	22	206	1846	380	34	491	
	35	22.24 %	58.10%	11.44%	15.80%	21.81%	7.61%	7.85%	70.38%	13.29%	26.15%	40.92%	
20 × 20													91.94 (86.11

Table 20 presents the confusion matrix for the training phase based on the recall and the precision metrics per class. Table 21 presents the comparative results of recall and precision for various studies. As seen in this table the proposed method outperformed the other methods in the most cases. The only drawback of the proposed method based on the recall is its performance for Normal and Exploits classes where their values are a little less than Ramp-KSVCR algorithm. As the precision point of view, the proposed method clearly outperformed other methods, except for Backdoor, Fuzzers, and DoS classes that are lower than Ramp-KSVCR. However, the proposed method scores over these methods in the majority of cases. On the other hands, the additional important comparison is given in Table 22 based on the total accuracy, detection rate (DR), false alarm rate (FPR) and F1-Score. It can be seen that the applied performance metrics on the proposed method is higher than several existing algorithms except for overall DR that is in the second rank. Overall, it can be concluded that the proposed method is better considering its performance levels across both attack and non-attack classes and the advantages incurred by discovering the number of data in each class acting as benign outliers that affect negatively on the training accuracy.

Table 20: Confusion Matrix of training process for UNSW-NB15

	Normal	Backdoor	Analysis	Fuzzers	Shellcode	Reconn.	Exploits	DoS	Worms	Generic	Recall (%)
Normal	1593	1	0	73	0	1	12	0	0	0	94.82
Backdoor	0	302	2	9	0	1	108	15	0	0	71.11
Analysis	0	1	446	0	0	0	49	4	0	0	89.20
Fuzzers	0	11	2	4330	3	55	126	15	0	1	95.31
Shellcode	0	0	0	19	246	23	1	0	0	0	85.12
Reconn.	0	16	2	152	0	2331	112	10	0	0	88.87
Exploits	0	42	9	249	1	28	8113	101	2	0	94.94
DoS	0	30	4	36	1	11	300	2484	0	0	86.67
Worms	0	1	0	4	0	4	33	0	88	0	67.69
Generic	0	0	0	1	0	0	3	0	0	1196	99.67
Precision (%)	100	74.75	96.91	88.86	98.01	94.99	91.60	94.48	97.78	99.92	

Table 21: The training results on UNSW-NB15 based on each class

Method		Normal	Backdoor	Analysis	Fuzzers	Shellcode	Reconn.	Exploits	DoS	Worms	Generic
Proposed method (25 × 25)	Recall (%)	94.82	**71.11**	**89.20**	**95.31**	**85.12**	**88.87**	94.94	**86.67**	**67.69**	**99.67**
	Precision (%)	**100**	74.75	**96.91**	88.86	**98.01**	**94.99**	**91.06**	94.48	**97.78**	**99.92**
Proposed method (20 × 20)	Recall (%)	92.38	67.35	71.08	92.38	64.41	79.17	77.81	86	62.31	98.75
	Precision (%)	99.95	75.40	83.39	80.32	93.97	79.40	88.13	73.82	97.59	99.50
Fuzzy SOM (25 × 25) Karami & Guerrero-Zapata (2014)	Recall (%)	91.78	55.81	52.35	83.99	60.15	70.25	81.42	73.50	50	98.84
	Precision (%)	96.82	91.47	91.89	70.61	66.61	86.26	78.39	76.91	96.33	98.88
SOM (25 × 25)	Recall (%)	91.03	44.99	47.83	85.35	49.51	61.69	91.77	55.84	54.81	98.39
	Precision (%)	97.37	78.45	67.43	63.41	86.32	86.37	72.18	83	91.36	98.82
Ramp-KSVCR Bamakan et al. (2017)	Recall (%)	**97.38**	70.44	69.83	87.5	58.2	83.8	**95.61**	84.81	38.24	97.81
	Precision (%)	97.54	**97.33**	96.89	**91.93**	97.93	66.18	90.73	**98.51**	81.25	98.60

6. Visualization Capacities in the proposed IDS

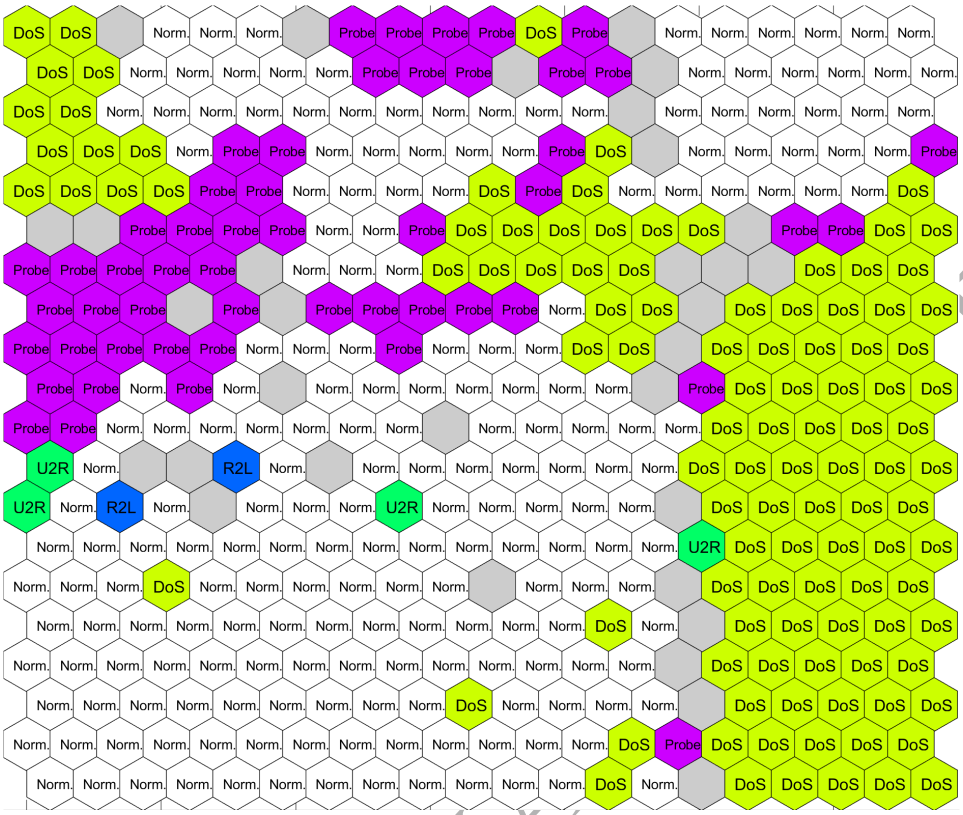
One of the common ways to assess the performance and effectiveness of the IDS from end users' point of view is visualization capabilities. Up to present, there have been no inspiring visualization-based IDSs to propose all the possible knowledge required from multi-dimensional and big datasets applied for intrusion detection purposes with less computational costs. For instance, Principle Component Analysis (PCA) projection in Jia et al. (2016); la Hoz et al. (2015); Corchado & Álvaro Herrero (2011);

Table 22: The results of overall training performance (%) on UNSW-NB15

Method	Accuracy	DR	FPR	F1-Score
Proposed method (25 × 25)	**98.26**	87.44	**1.95**	**89.17**
Proposed method (20 × 20)	94.12	79.17	5.38	80.62
Fuzzy SOM (25 × 25) Karami & Guerrero-Zapata (2014)	90.61	71.81	9.42	75.93
SOM (25 × 25)	89.20	68.12	12.67	73.14
Ramp-KSVCR Bamakan et al. (2017)	93.52	78.44	2.46	N/A
PSI-NetVisor Mishra et al. (2017)	94.54	N/A	2.81	N/A
GAA-ADS Moustafa et al. (2017)	92.8	**91.3**	5.1	N/A

Kiziloren & Germen (2009) and Curvilinear Component Analysis (CCA) projection in Corchado & Álvaro Herrero (2011); Herrero & Corchado (2011); Shakhatreh & Bakar (2011) have been frequently used and incorporated into SOM algorithm for IDS visualization. However, they add additional computational costs into IDS. In our research, we used SOM visualization capabilities to demonstrate important and useful information to be easy for interpretation by end users who need to get well enough knowledge timely. Among the conducted experiments with different lattice sizes, the results of developed IDS with the lattice sizes of 20×20 and 25×25 are depicted in Figures 14 and 17 for NSL-KDD and UNSW-NB15, respectively. Figures 14a and 17a show the SOM neurons' labels for both IDS datasets. The hexagonal shapes with grey colour are non-selected and empty nodes. The neighbour of each neuron is intuitively visible in order to find the neurons (nodes) with attack labels in the neighbourhood of normal neurons. Figures 14b and 17b show the purity of each SOM nodes for both IDS datasets. A pure black colour means a pure cluster (100% similar data points within a cluster) while a pure red colours mean an impure cluster (All the data points are dissimilar and have different labels). There is a hue ranges between red and black to express the percentage of a pure cluster. White hexagonal shapes are empty nodes.

An additional useful graphical representation based on the 2D SOM lattice is shown in Figures 15 and 18 for NSL-KDD and UNSW-NB15 datasets, respectively. The left sub-figures show the 2D lattice adjustment derived from 41D NSL-KDD and 49D UNSW-NB15, respectively. The user can zoom in to some regions, such as very dense nodes to be able to have a deeper vision into clusters. Moreover, the user can select each cluster (neuron) to get a useful message in the same screen, such as the label and the purity of the cluster. This design displays a high-level view of entire 41D for NSL-KDD and 49D for UNSW-NB15 input vectors. Preliminary results can confirm that the proposed method produces satisfactory outcomes. In addition, end users can see the results of anomaly detection from new monitoring data (i.e., testing set) graphically in Figures 16 and 19, together with the numerical results for classification accuracy displayed in Tables 23 and 24 for NSL-KDD and UNSW-NB15, respectively.
Figures 16 and 19 illustrate the graphical representation of normal and attack traffic, as well as anomaly detection from new monitoring data (testing set) for NSL-KDD and UNSW-NB15, respectively. The black hexagons are normal classes that act as anomalies because of some new attack data samples from testing set are closer to them than attack classes (i.e., detection phase in Algorithm 3). The associated labels can visually help end users for understanding the classification results. The neurons with grey colour are empty. Simultaneously, the users can figure out the purity of each clusters (hexagonal nodes) for new monitoring data intuitively in Figures 16b and 19b for NSL-KDD and UNSW-NB15, respectively. These graphical representations through neuron projections steer the user's attention towards the most reliable clusters storing either normal or attack data points to drill down anomalous data. End users can visually see the results of these figures in parallel, to be able to figure out some hidden knowledge in each cluster.



(a) Neurons' labels

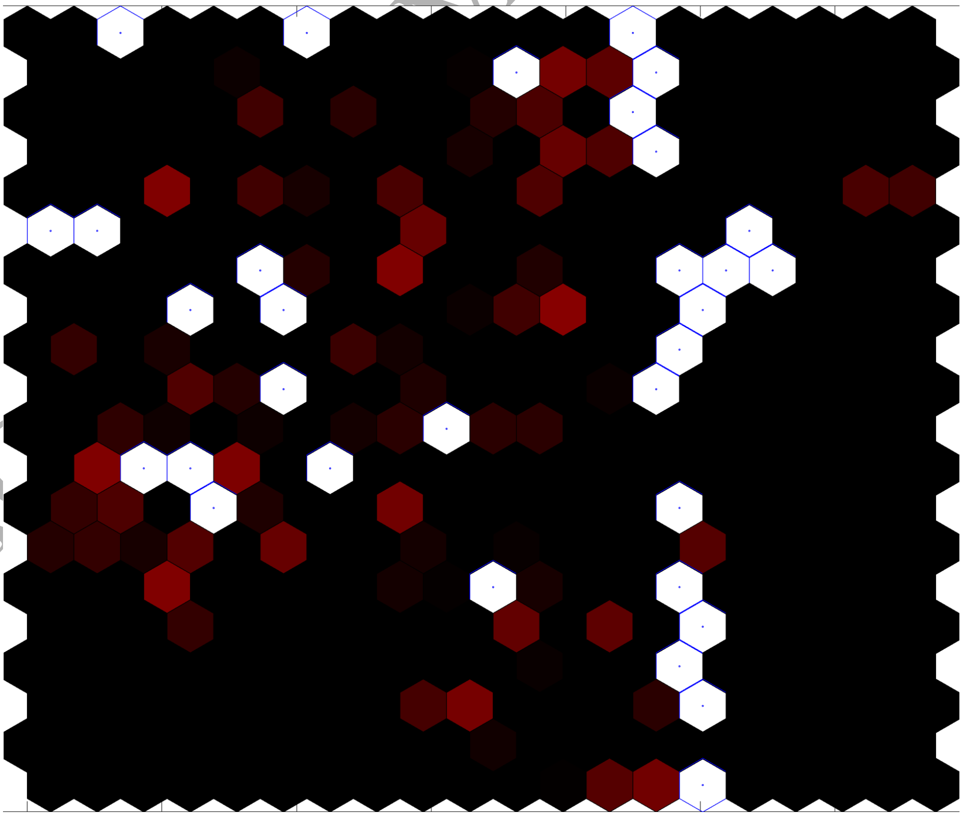

(b) Neurons' purity

Figure 14: Visualization results of the proposed method for NSL-KDD (lattice size 20 × 20)

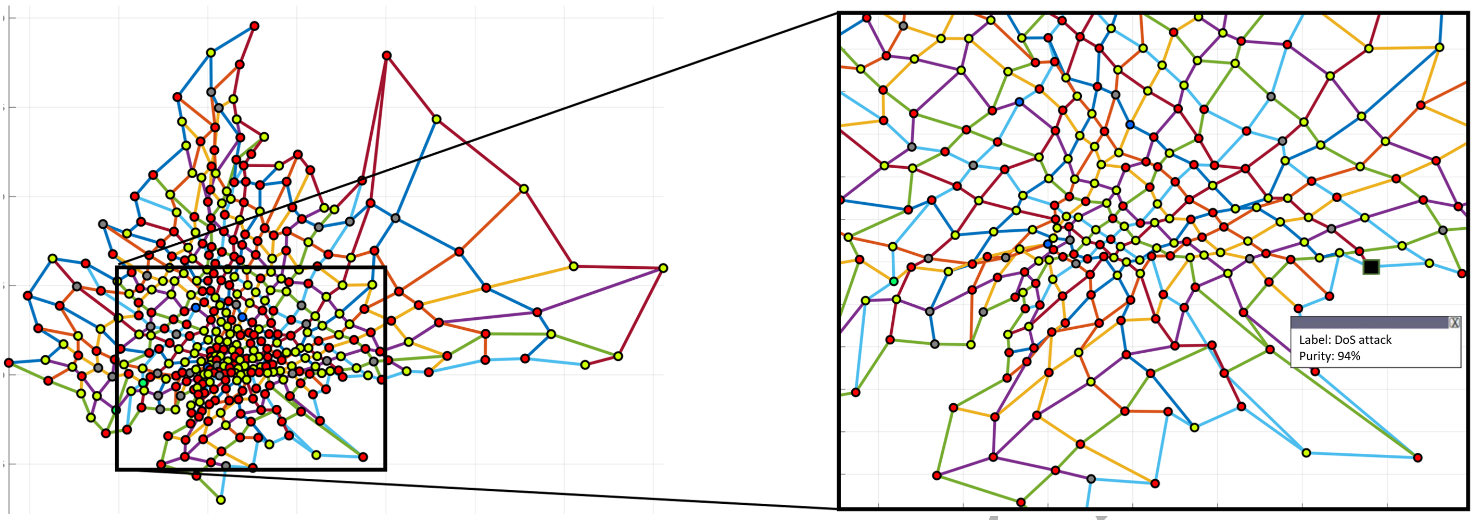


Figure 15: A 20 × 20 lattice visualization with user interaction capabilities in NSL-KDD

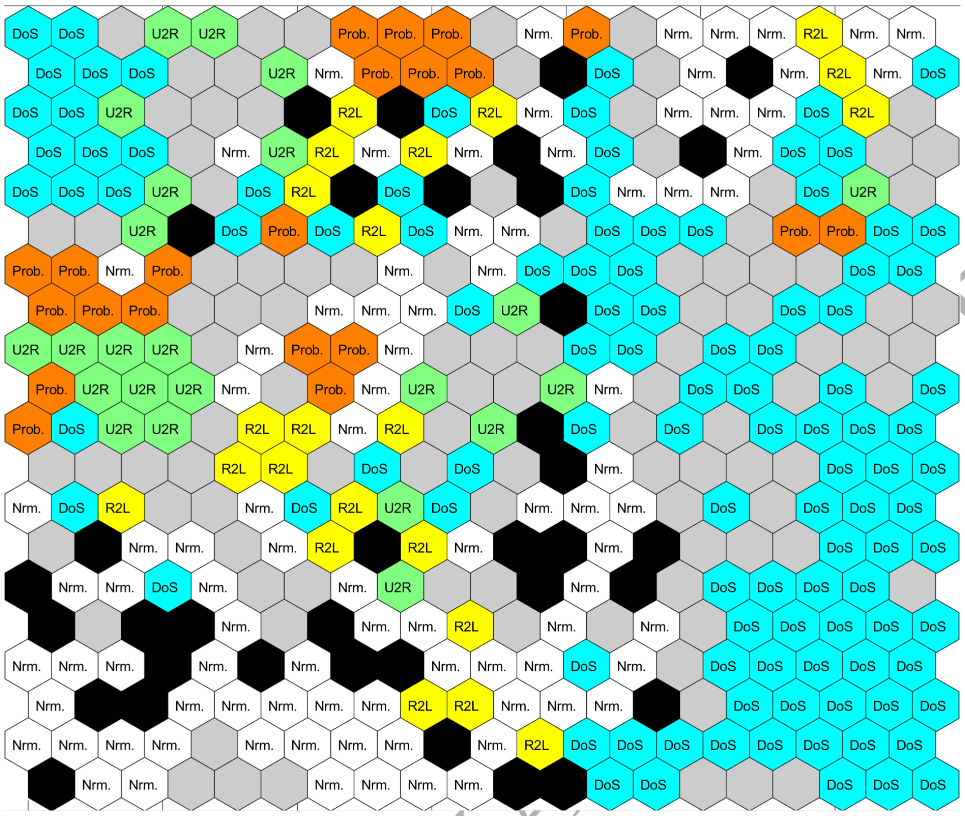
Finally, a classification is performed on testing sets to determine exact normal and attack categories with a five-fold cross validation. We randomly created 30% of the testing set in each validation step. The classification accuracy with the exact number of classes in each validation step is presented in Tables 23 and 24 for NSL-KDD and UNSW-NB15, respectively. Table 23 shows the number of sampled data together with the percentage of misbehaving samples as anomaly per class in NSL-KDD testing set. Obviously, only attack classes misbehave as anomaly, where some of them are closer (more similar) to normal clusters than attack ones. According to the experimental results, Probing and U2R have more misbehaved data samples with the average of 11.41% and 9.46%, respectively. Overall, the average accuracy and F1-score for new monitoring data (testing set) are 95.45% and 89.03%, respectively.

Table 24 shows the number of sampled data together with the percentage of misbehaving samples as anomaly per class in UNSW-NB15 testing set. According to the experimental results, Exploits, Generic, and Fuzzers have more misbehaved data samples with the average of 16.48%, 15.53%, and 11.49% respectively. In contrast, Reconnaissance and Shellcode have less misbehaved data samples with the average of 4.15% and 2%, respectively. Overall, the average accuracy and F1-score for new monitoring data (testing set) are 95.24% and 81.50%, respectively. A future work is needed in the proposed detection phase (see Algorithm 3) by replacing a new intelligent reaction mechanism with the threshold-based solution to improve the accuracy and the harmonic mean of precision and recall (F1-score) of classification.

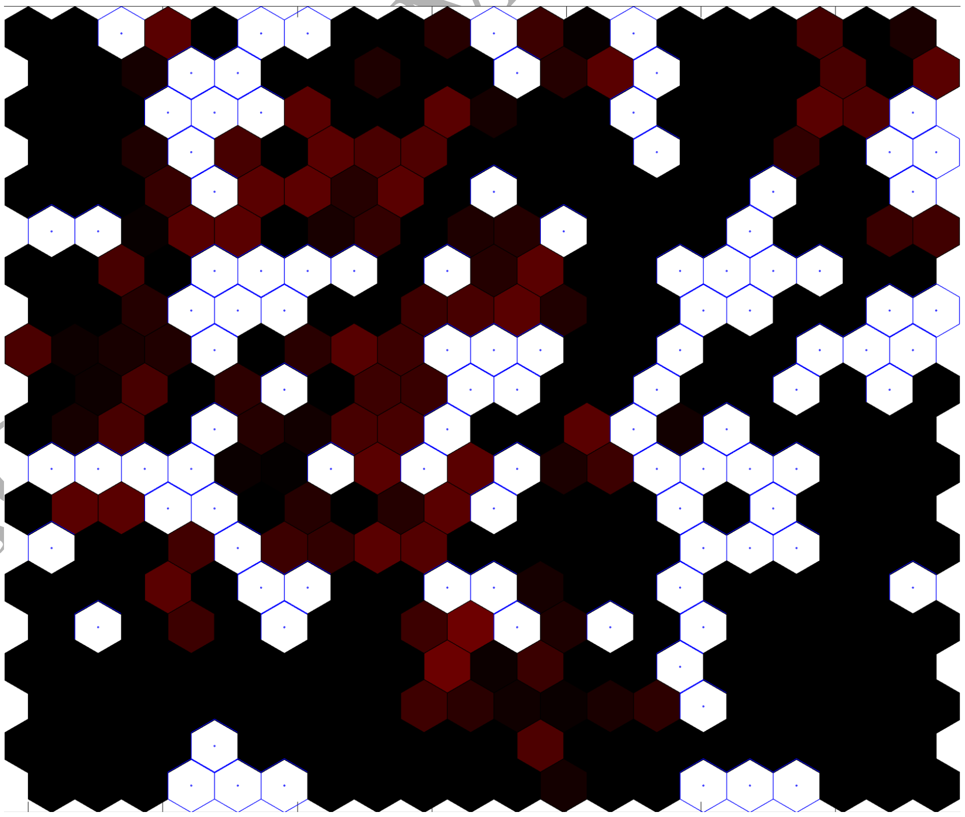
Table 23: The classification performance results on NSL-KDD

			The number of data in each class (anomalous data (%))				
Testing set	Accuracy	F1-Score	Normal	DoS	U2R	R2L	Probing
Test 1	95.39%	88.84%	2833	2286(5.10%)	363(10.41%)	868(10.09%)	414(11.31%)
Test 2	95.57%	89.61%	2903	2233(4.93%)	366(9.92%)	830(9.12%)	432(12.40%)
Test 3	95.33%	88.07%	2929	2218(4.76%)	349(9.34%)	846(7.93%)	422(9.81%)
Test 4	95.55%	89.39%	2887	2255(7.12%)	361(8.41%)	839(6.76%)	422(13.34%)
Test 5	95.39%	89.25%	2896	2213(6.19%)	362(9.23%)	835(10.02%)	458(10.18%)

There is an interesting finding within the experimental results. For instance in NSL-KDD testing set, Probing and U2R classes

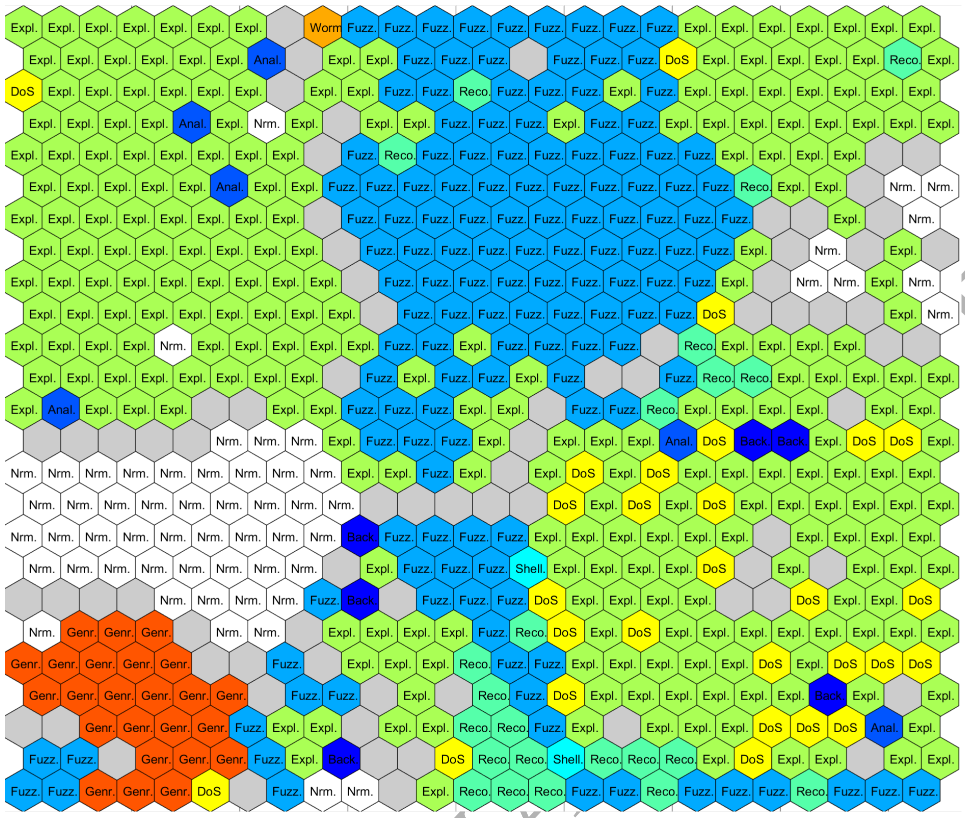


(a) Neurons' labels

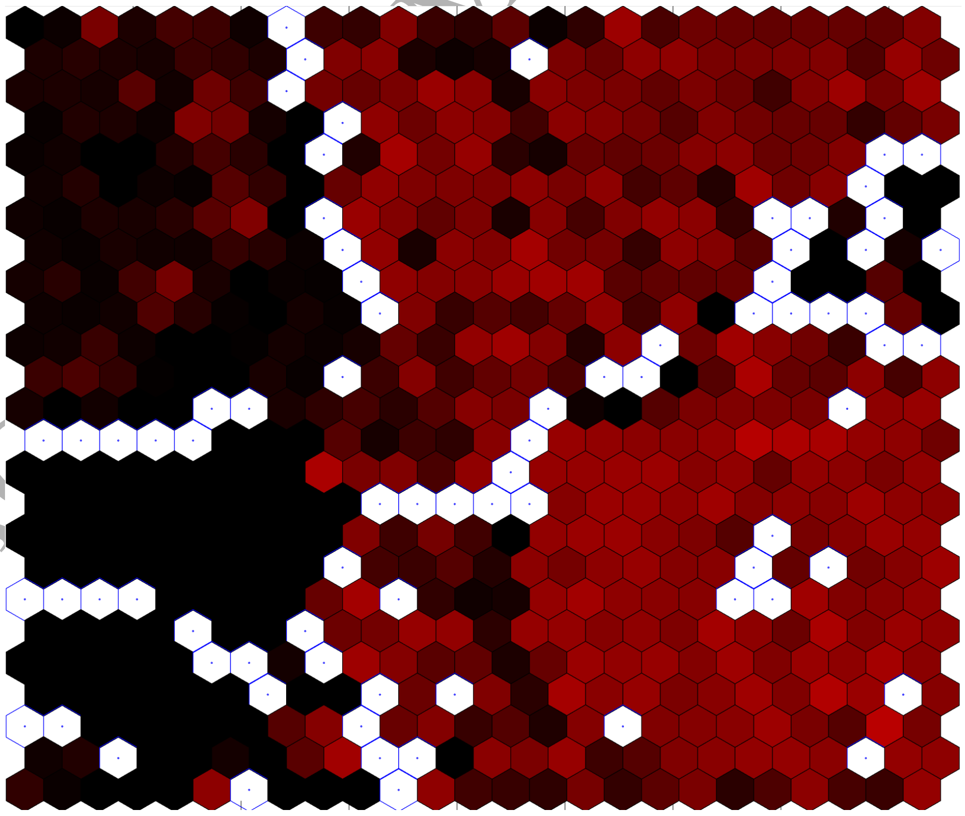


(b) Neurons' purity

Figure 16: A neuron visualization of Anomalous neurons (centres) for new monitoring data (lattice size 20 × 20) in NSL-KDD. Black hexagons in (a) are anomalous centres.



(a) Neurons' labels



(b) Neurons' purity

Figure 17: Visualization results of the proposed method for UNSW-NB15 (lattice size 25 × 25)

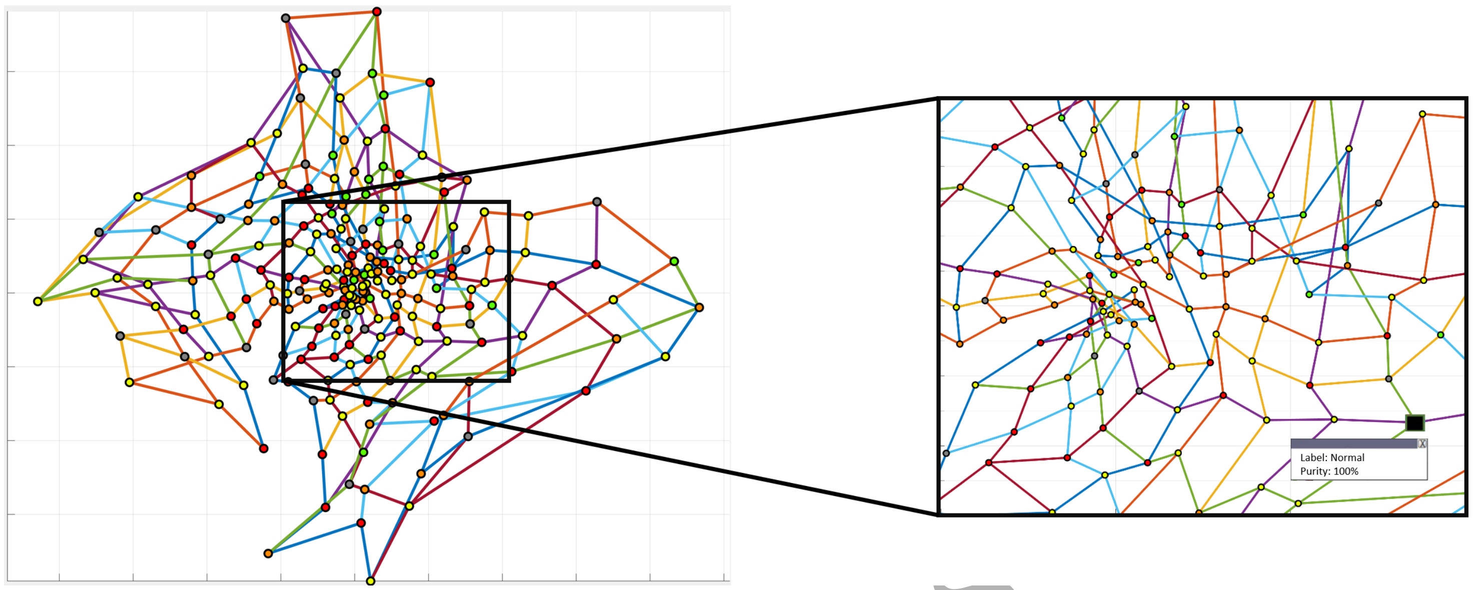


Figure 18: A 15 × 15 lattice visualization with user interaction capabilities in UNSW-NB15

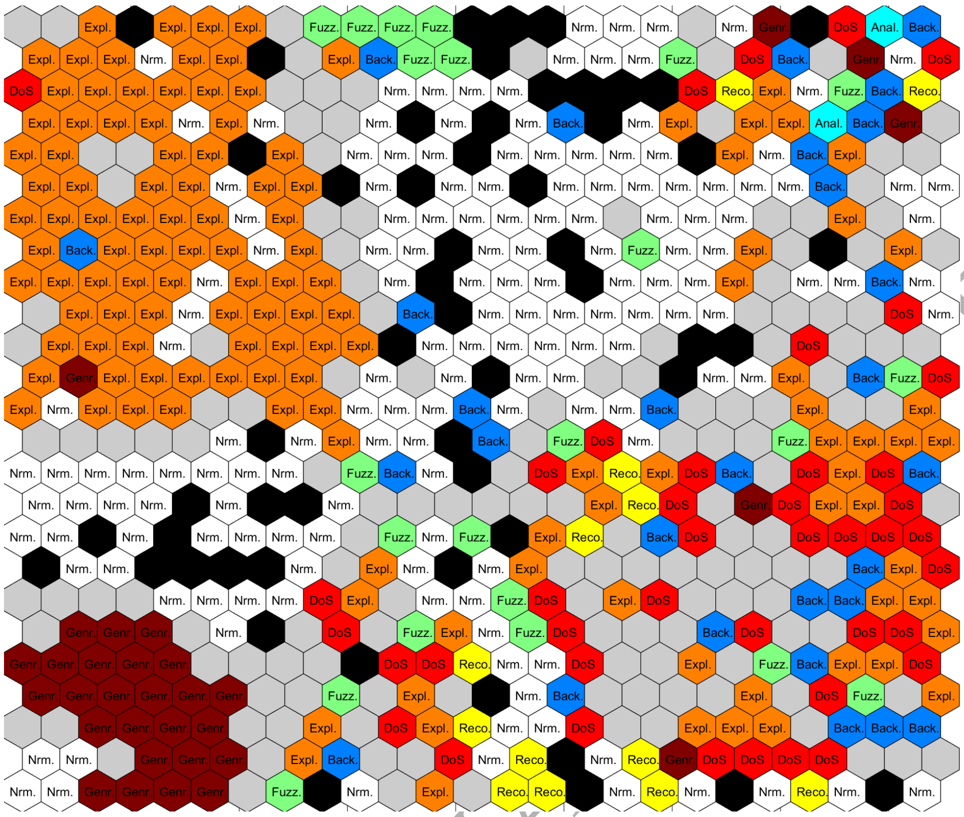
Table 24: The classification performance results on UNSW-NB15

			The number of data in each class (anomalous data(%))									
Testing set	Accuracy	F1-Score	Normal	Backdoor	Analysis	Fuzzers	Shellcode	Reconn.	Exploits	DoS	Worms	Generic
Test 1	95.25%	82.96%	11100	161(6.11%)	227(5.12%)	1797(13.72%)	125(1.12%)	1031(3.32%)	3327(15.18%)	1174(4.11%)	12(10.13%)	5746(14.17%)
Test 2	95.24%	80.25%	11054	167(6.24%)	209(4.92%)	1757(12.81%)	126(2.56%)	1021(4.09%)	3391(14.19%)	1281(4.44%)	7(12.02%)	5687(16.12%)
Test 3	95.19%	80.37%	11116	177(7.05%)	204(5.66%)	1872(12.02%)	122(1.04%)	1036(4.55%)	3270(17.73%)	1235(5.12%)	12(7.18%)	5656(15.04%)
Test 4	95.31%	82.58%	11136	166(5.41%)	184(6.81%)	1818(10.29%)	104(2.20%)	1053(3.93%)	3330(16.66%)	1230(5.08%)	10(9.93%)	5669(14.55%)
Test 5	95.21%	81.35%	11002	214(6.83%)	195(4.12%)	1821(8.61%)	118(3.11%)	1046(4.87%)	3320(18.62%)	1229(4.43%)	13(10.07%)	5742(17.78%)

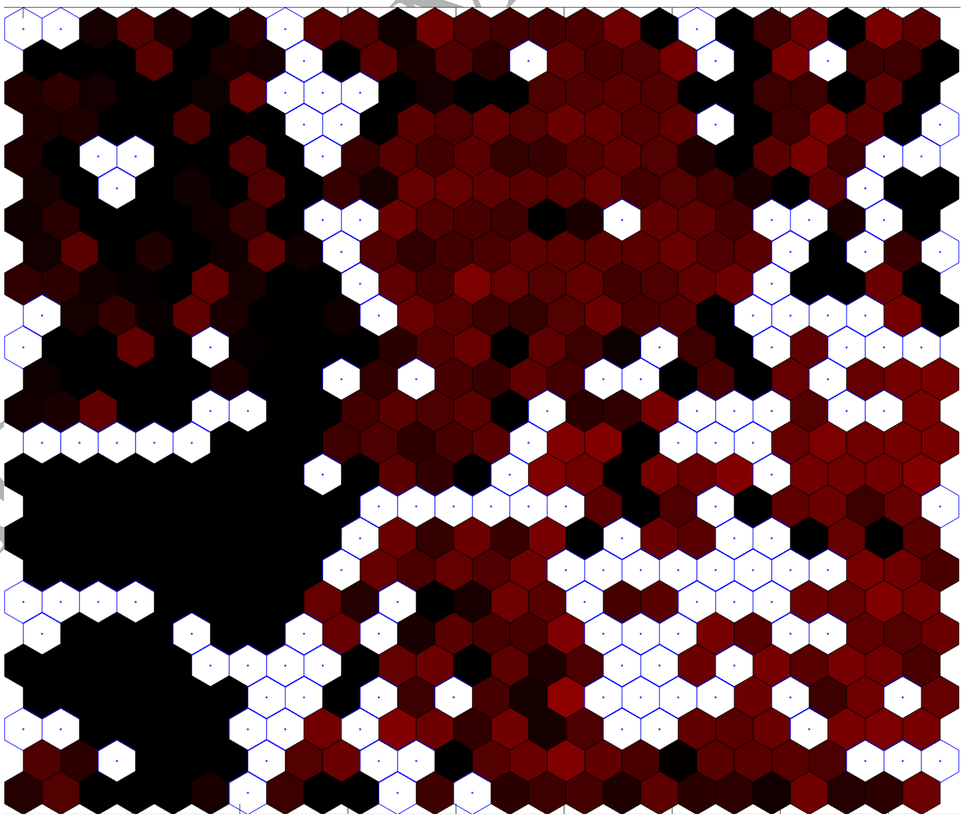
with the highest ratio of benign outliers during training (see Table 15) have more anomalies in the testing sets. In contrast, DoS class with the lowest ratio of benign outlier has relatively the lowest anomalous data samples. On the other hand for UNSW-NB15 testing sets, Exploits and Generic classes with the largest portion of benign outliers during training (see Table 19) have the most anomalies. In contrast, Shellcode and Reconnaissance classes with the least amount of benign outliers during training, have the lowest amount of anomalies in the testing sets.

7. The Effectiveness of the Proposed IDS through Usability Test

The main contribution of this research is finding a better way for detecting and visualizing intrusions and anomalies. End users would prefer to work with usable IDSs to be able to observe and track the status of the clusters labelled as normal or attack together with anomalies when new monitoring data enter. To ensure that an IDS can work properly in terms of the visualization capabilities without confusing the user, usability test should be performed. To measure the performance of the usability, we are specifically interested in two parameters: learnability (i.e., How easy is it for users to accomplish tasks the first time they encounter the design?) and satisfaction (i.e., How pleasant is it to use the design?) Adhy et al. (2017). These two usability parameters can meet our goal in this research as visualizing normal and attack traffic, and anomalies in a proper manner to end users. In our research, we gathered ten participators that are experts in IDS design and network security. The learnability and the satisfaction parameters are measured by giving out the questionnaire to participators, then counting the amount of which user can use, learn,



(a) Neurons' labels



(b) Neurons' purity

Figure 19: A neuron visualization of Anomalous neurons (centres) for new monitoring data (lattice size 25 × 25) in UNSW-NB15. Black hexagons in (a) are anomalous centres.

and understand the proposed IDS easily. We prepared the questionnaire with 14 questions (see Table 25), in which the first four questions are about learnability and the rested for satisfaction.

Table 25: Questionnaires for Usability Testing

No.	Question
Learnability Questions Adhy et al. (2017):	
1	This system design is learnable.
2	I can use this system design without any help from technician or developer.
3	The information and analyses about nodes have been provided well and very understandable.
4	The graphic about data received in every node has been provided well and very understandable.
Satisfaction Questions by System Usability Scale (SUS) method Brooke et al. (1996):	
5	I think that I would like to use this system frequently.
6	I found the system unnecessarily complex.
7	I thought the system was easy to use.
8	I think that I would need the support of a technical person to be able to use this system.
9	I found the various functions in this system were well integrated.
10	I thought there was too much inconsistency in this system.
11	I would imagine that most people would learn to use this system very quickly.
12	I found the system very cumbersome to use.
13	I felt very confident using the system.
14	I needed to learn a lot of things before I could get going with this system.

To analyse the results, we used the seven points of Likert scale including 1 = strongly disagree, 2 = disagree, 3 = almost disagree, 4 = neutral, 5 = almost agree, 6 = agree, and 7 = strongly agree. Both learnability and satisfaction criteria are measured using the ideal and the actual scores as follows Adhy et al. (2017):

$$Ideal\ Score = The\ biggest\ score\ scale \times The\ number\ of\ respond \tag{12}$$

$$Actual\ Score = The\ number\ of\ score\ of\ each\ question \tag{13}$$

$$Percent = \frac{Actural\ Score}{Ideal\ Score} \times 100\% \tag{14}$$

The questionnaires results are summarized in Table 26. From the first four questions, learnability is calculated as $(95.71 + 92.85 + 100 + 80)/4 = 92.14\%$. Similarly, the satisfaction is calculated based on the last ten questions as $(92.85 + 77.15 + 87.14 + 84.28 + 82.80 + 72.86 + 88.57 + 72.86 + 84.28 + 81.42)/10 = 82.42\%$. The collected scores for learnability show that the 92.14% is reasonable outcome for this design. The only weak score goes for question 4 by 80% about the abstract information and knowledge in each cluster. It is suggested that information about clusters would be better to develop informatively, such as some descriptive and inferential analyses on each. The participants also expressed that we have to embed a simple and straightforward dashboard to present the numerical results about the number of benign outliers and anomalies beside graphical representations. It was demanded more when we had many clusters (neurons) that entirely occupied the screen. The collected scores for satisfaction show that the 82.42% is a good achievement for this first design. The lowest scores are for questions 6, 10, and 12. The important comments from participators for satisfaction is about reading and understanding the 2D visualization for anomaly detection by user interaction capabilities. If the number of visualized nodes are reasonably low, they can interact better with the system. In contrast, the large number of nodes are not user friendly to drill down in each. The participants suggested to use some pop-up messages for proposing additional useful knowledge for each classes, in particular when the number of anomalous clusters and the misbehaved new monitoring data samples are increasing over the time. It helps end users to understand and learn the trend of classification (prediction) for new monitoring data samples. At end, a few participators asked for an advanced visualization with a proper connection between neurons' label, neurons' purity, 2D lattice visualization and anomalous data samples. For instance, they like to get more insights by zooming into Figure 15 or 18 and select one neuron, then this neuron be fired and highlighted in some other figures.

Table 26: Questionnaires Result

Question	Participants										Actual Score	Percent (%)
	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th		
1	7	7	7	6	7	6	7	7	6	7	67	95.71
2	7	7	7	5	6	7	7	6	6	7	65	92.85
3	7	7	7	7	7	7	7	7	7	7	70	100
4	6	5	6	5	6	6	6	5	5	6	56	80
5	7	6	6	7	7	7	6	6	6	7	65	92.85
6	1	1	1	2	2	2	1	1	3	2	16	77.15
7	6	7	7	7	7	6	5	5	6	5	61	87.14
8	5	6	5	6	7	5	6	7	7	5	59	84.28
9	5	6	6	7	6	6	5	5	6	6	58	82.80
10	1	1	2	2	3	3	2	2	2	1	19	72.86
11	6	6	6	7	7	7	6	5	5	7	62	88.57
12	2	3	3	2	2	1	1	2	2	1	19	72.86
13	5	5	6	6	6	7	7	5	6	6	59	84.28
14	5	5	6	6	7	7	5	5	6	5	57	81.42

8. Discussion on the proposed visualization-based IDS

From the visualization point of view, the proposed method offers a complete and intuitive visualization of network traffic with a variety of patterns (normal, known and unknown attacks) by depicting the general overview of each formed clusters, the type of clusters, the quality of each neuron (cluster) by purity metric, the distribution of 2D lattice and the neighbours' similarities and dissimilarities in one useful plot. The proposed visualization-based IDS could successfully provide the network administrator with a snapshot of network traffic into separated categories in order to identify the intrusions and anomalous network traffic visually rather than sending a massive amount of alerts to administrators. On the other hand, the proposed graphical representation of the multi-dimensional and the high volume of network traffic provides less computational costs as compared to dimensionality reduction techniques that are widely applied. To do so, we employed SOM capabilities for visualization without additional computational costs. Moreover, the proposed graphical representation tools (Figures 14, 15, and 16 for NSL-KDD dataset and Figures 17, 18, and 19 for UNSW-NB15 dataset) can provide a general, easy and understandable overview of the traffic within a network for even an inexperienced network administrator to identify categories (clusters), normal and anomalous traffic data just having a quick look at the proposed projections. Moreover, a numerical analysis has been done (Tables 23 and 24 for NSL-KDD and UNSW-NB15, respectively) to add more knowledge beside the graphical representations. Table 27 compares the characteristics of the visualization-based IDSs. The challenge of large-data visualization, which involves both human and machine limitations, will

remain relevant in the foreseeable future. As a future work, visual analytic features are needed to perform an interactive visual analysis on the network traffic data, such as through highlighting, brushing, and filtering of data or dimensions.
Convergence of the proposed visualization-based classifier is studied for several frequently used synthetic datasets detailed in Table 1 and Figure 4 in Section 5.1.1. Afterwards, we applied the proposed method for the benchmarking datasets in intrusion detection problems in Sections 5.2 and 5.3 for NSL-KDD and UNSW-NB15, respectively. To complete our extensive analysis, we employed two new benchmarking datasets (AAGM and VPN-nonVPN) in the next Section (refer to Section 9). Experimental results confirm the accuracy and the robustness of the proposed approach. On the other hand, the feasibility and efficiency of the proposed method was compared with some existing algorithms. The obtained results in Section 5 express that the proposed method is able to construct more accurate and well-tuned SOM neurons. Finally, Sections 6 and 7 helped us to discover and figure out users feelings and experiences while they were interacting with. It introduces new research directions and challenges within user-centered IDS design.

Table 27: Comparing Visualization Capabilities for IDS design

Contribution	Visualization method	Visualizing Anomalous Nodes	2D Lattice Visualization	Intrusion Relationships	User Interaction with 2D plot	Extra Cost for Visual.	Convergence
Jia et al. (2016)	PCA	✗	✗	✓	less	✓	NSL-KDD
la Hoz et al. (2015)	PCA+FDR+PSOM	✗	✗	✗	No	✓	NSL-KDD
Luo & Xia (2014)	FASVFG Classifier	✗	✗	✓	No	✓	KDDcup99
Corchado & Álvaro Herrero (2011)	SOM+CMLHL+CCA	✓	✓	✗	less	✓	GICAP-IDS, DARPA
Kiziloren & Germen (2009)	SOM+PCA	✗	✓	✗	No	✓	KDDCup99
Proposed method	A modified SOM	✓	✓	✓	middle	✗	NSL-KDD, UNSW-NB15 AAGM, VPN-nonVPN

9. Analysis on the recent and custom network traffic datasets

There has been much effort from IDS/IPS research communities to construct new datasets to present the specific and new types of intrusions. The main reason is that some current benchmarking datasets such as KDD and DARPA are suffering from the lack of traffic diversity and volumes, and do not cover the variety of known attacks Hamed et al. (2018). NSL-KDD and UNSW-NB15 IDS datasets could significantly improve the old IDS benchmarking datasets in terms of traffic volumes and diversity. However, some other researchers have been trying to generate and introduce novel datasets due to the nature of computer networks and Internet that they most likely bring new patterns over time. In this section, we employed two recently used network traffic datasets, generated and published by University of New Brunswick (UNB) UNB (2018) including Android Adware and General Malware (AAGM), and VPN-nonVPN datasets.

9.1. Android Adware and General Malware (AAGM)

AAGM is a new network traffic set with nine flow-based network traffic features for characterizing three types of malwares: benign, adware, and general malware Lashkari et al. (2017). AAGM dataset is captured by installing the Android apps on the real smartphones on NEXUS 5. The dataset is generated from 1900 applications with the following three categories:

1. Benign (1500 apps): GooglePlay market (top free popular and top free new) in 2015 and 2016.
2. Adware (250 apps): Airpush, Dowgin, Kemoge, Mobidash, and Shuanet
3. General Malware (150 apps): AVpass, FakeAV, FakeFlash/FakePlayer, GGtracker, and Penetho

This dataset includes 75 features with apps collected between 2008 and 2016. For more information about this dataset and the process of traffic generation refer to the original paper in Lashkari et al. (2017). There is a dataset including 631,955 instances with 471,597 benign, 155,613 adware, and 4,745 general malware instances. We extracted one-fifth (126,391 instances) of this big dataset randomly for modelling, then we divided this derived dataset to 70% (55,297 instances) for training and 30% (23,698 instances) for testing. We implemented a 10-fold cross validation in our experiment to propose the best achieved outcomes. A comparison with the proposed classifier in Lashkari et al. (2017) is given in Table 28 based on the total accuracy, DR, FPR and F1-Score, as well as the number of benign outliers for each class.

Table 28: The results of overall training performance (%) on AAGM

					The number of benign outliers		
Method	Accuracy	Precision	FPR	F1-Score	Benign	Adware	General Malware
Proposed method (15 × 15)	**93.35**	**93.11**	3.73	**88.65**	5.15 %	28.65 %	39.66 %
Lashkari et al. (2017)	91.41	91.24	**0.085**	N/A	–	–	–

Figures 20a and 20b show the SOM neurons' labels and purity after training. Similarly, Figures 20c and 20d show the labels and purity for anomaly detection during testing phase. The way of interpreting these figures were explained in details in Section 5.5 and 5.6. A graphical representation based on the 2D SOM lattice is shown in Figure 22a for AAGM dataset. This illustration shows the 2D lattice adjustment derived from 79D dataset. We assumed Benign instances as normal and others as abnormal because the benign instances are less misbehaved than others. The user can zoom into some regions, such as very dense and sparse nodes to have a deeper vision on each cluster. In the end, Table 29 presents the classification performance results on testing data.

Table 29: The classification performance results on AAGM

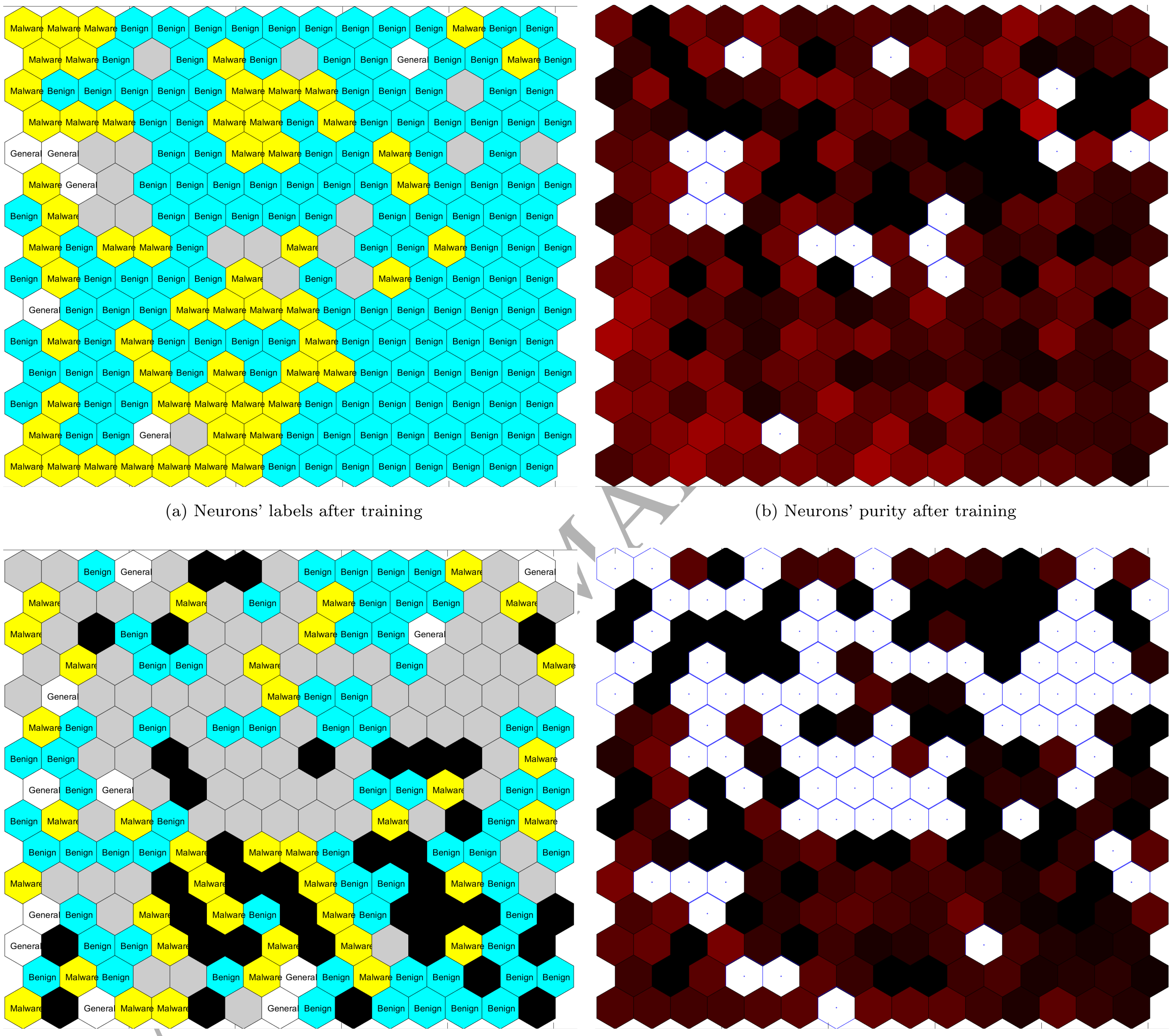
Criteria	AAGM's labels		
	Benign	Adware	General Malware
Accuracy (%)	92.92	87.65	88.51
Precision (%)	90.33	83.67	87.53
FPR (%)	1.29	5.52	4.04
F1-Score (%)	85.39	83.77	82.14

9.2. ISCX VPN-nonVPN

We use ISCX VPN-nonVPN traffic dataset Draper-Gil et al. (2016) that consists of captured traffic of different applications with 24 features. The captured protocols and applications are Web Browsing, Email, Chat, Streaming, File Transfer, VOIP, and P2P. For each traffic types (e.g., VOIP, P2P, etc.), there are VPN and non-VPN traffic categories. For more information about this dataset and the process of traffic generation refer to the original paper in Draper-Gil et al. (2016). We used two different datasets as follows:

1. 15-sec VPN and non-VPN instances: 9,793 VPN and 8,965 non-VPN samples (total: 18,758).
2. 120-sec VPN and non-VPN instances: 5,631 VPN and 5,151 non-VPN samples (total: 10,782).

We divided these datasets to 70% for training and 30% for testing. After that, we implemented a 10-fold cross validation in our experiment to propose the best achieved outcomes. A comparison with the proposed classifier in Draper-Gil et al. (2016) is given in Table 30 based on the total Recall, Precision, FPR and Accuracy, as well as the number of benign outliers for each class.



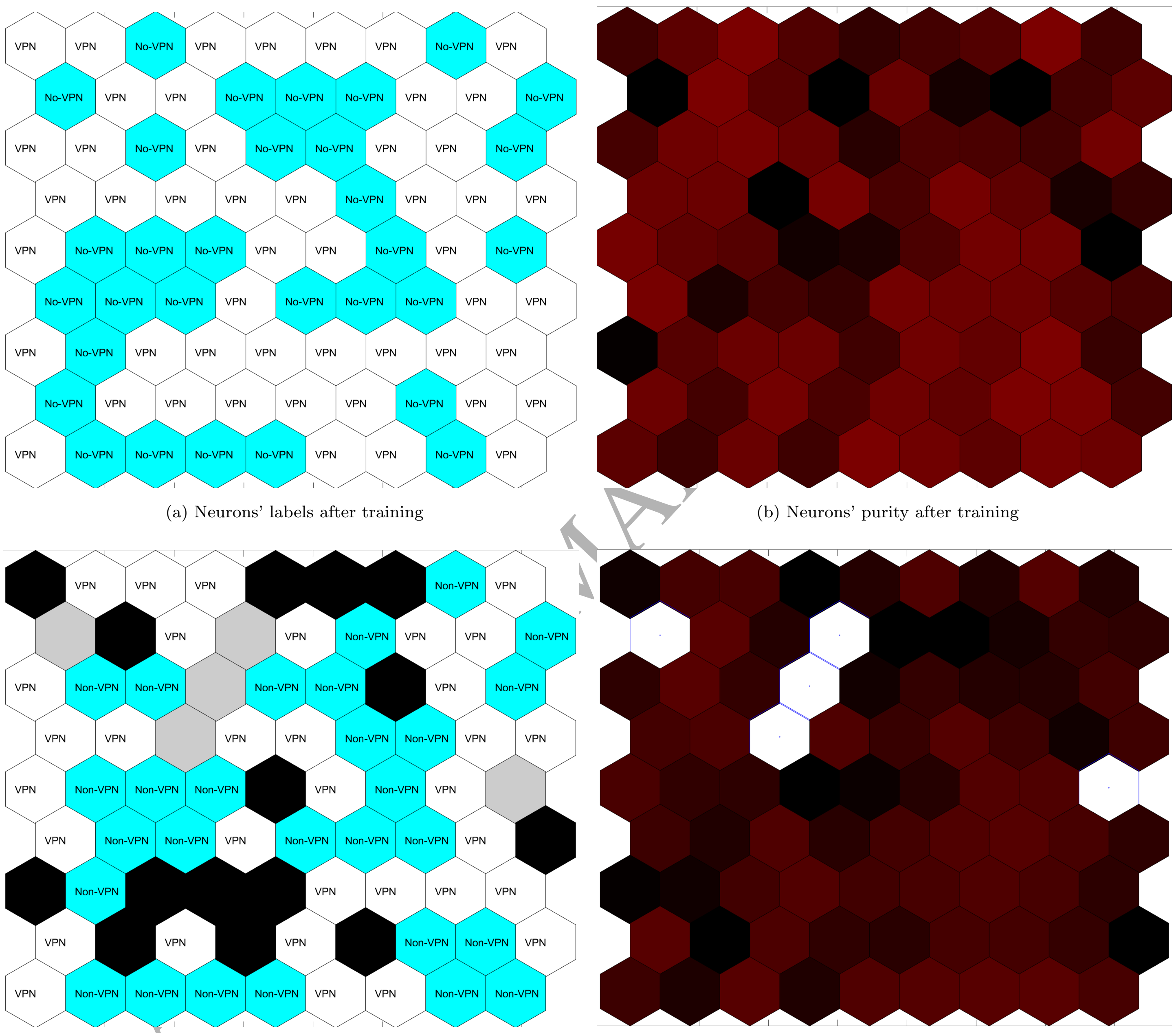
(a) Neurons' labels after training

(b) Neurons' purity after training

(c) Neurons' labels for anomaly detection

(d) Neurons' purity for anomaly detection

Figure 20: A neuron visualization for AAGM dataset (lattice size 15 × 15).



(a) Neurons' labels after training

(b) Neurons' purity after training

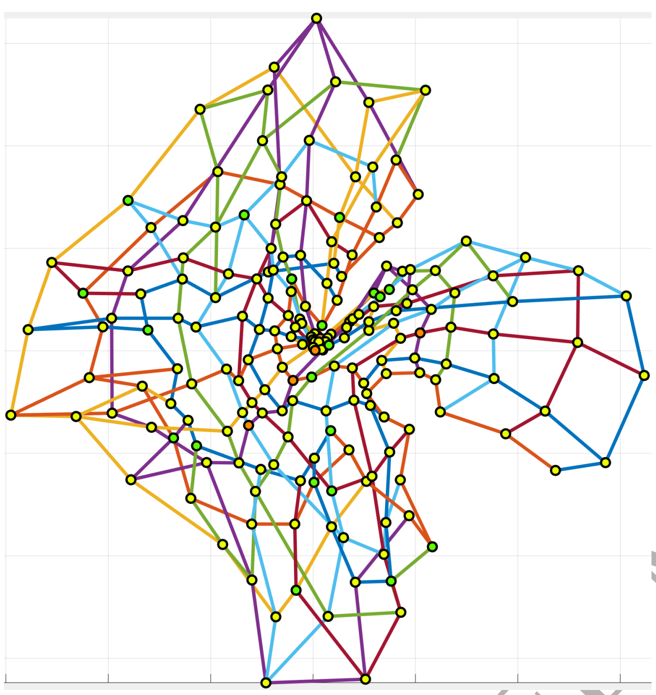
(c) Neurons' labels for anomaly detection

(d) Neurons' purity for anomaly detection

Figure 21: A neuron visualization for VPN-nonVPN (15-sec) dataset (lattice size 9 × 9).

Table 30: The results of overall training performance (%) on ISCX VPN-nonVPN

Method	Recall	Precision	FPR	Accuracy	The number of benign outliers	
					VPN	non-VPN
Proposed method (9 × 9) (15-sec)	**91.96**	**92.62**	**6.04**	**90.15**	23.89 %	47.11 %
Draper-Gil et al. (2016) (15-sec)	87.45	89.8	N/A	N/A	–	–
Proposed method (9 × 9) (120-sec)	**91.15**	**91.01**	**5.31**	**90.19**	38.46 %	57.02 %
Draper-Gil et al. (2016) (120-sec)	85.5	87.35	N/A	N/A	–	–


(a) A 15 × 15 lattice visualization15 for AAGM dataset

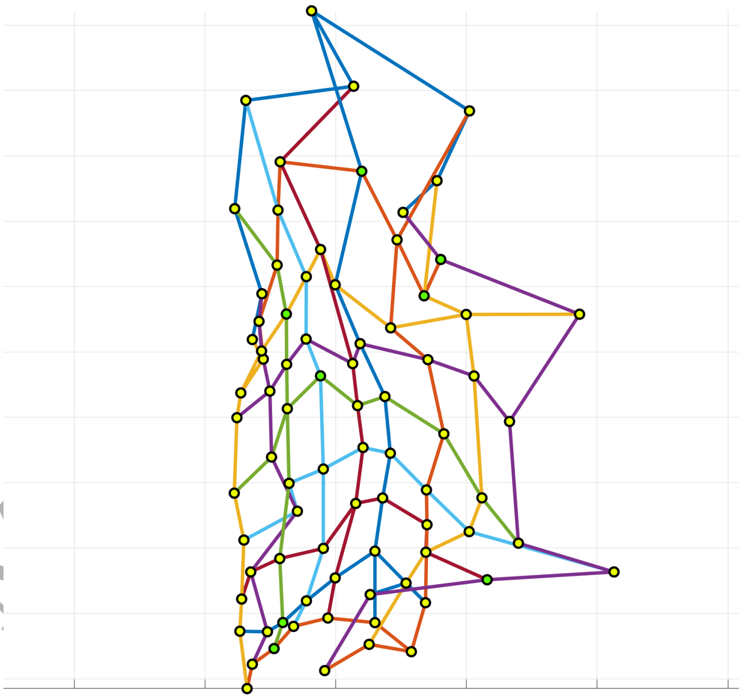

(b) A 9 × 9 lattice visualization for VPN-nonVPN dataset (15-sec)

Figure 22: 2D lattice visualizations with user interactions for AAGM and VPN-nonVPN datasets

Figures 21a and 21b show the SOM neurons' labels and purity after training. Similarly, Figures 21c and 21d show the labels and purity for anomaly detection during testing phase. A graphical representation based on the 2D SOM lattice is shown in Figure 22b for VPN-nonVPN dataset. This illustration shows the 2D lattice adjustment derived from 24D dataset. We assumed VPN instances as normal and nonVPN instances as abnormal. In the end, Table 31 presents the classification performance results on testing data.

Table 31: The classification performance results on VPN-nonVPN

Criteria	VPN-nonVPN's labels	
	VPN	non-VPN
Accuracy (%)	91.06	88.12
Precision (%)	90.36	84.80
FPR (%)	4.37	2.78
F1-Score (%)	88.13	89.14

10. Conclusions

This research work presented a novel anomaly-based intrusion detection system by visualization capabilities using modified Self-Organizing Map (SOM) in the presence of benign outliers. The benign outliers refer to low-frequent data patterns resulting in weaker detection stability and robustness. To deal accurately with low-frequent patterns to not affect negatively on IDS performance, the proposed method considered benign outliers and rested normalities separately in the training phase. The experimental results show that the proposed approach performs well and effectively as compared to some frequently used existing approaches. Consequently, the proposed method visualizes useful information and insights about training and testing results. The proposed visualization capabilities enable better analysis and response intuitively by considering the limitations in human cognitive ability when dealing with IDS including the large volumes of information which are not possible to fit all the requirements into one screen.
We can build larger and higher-resolution visual representations, however the limitations of human visual capabilities prevent the effectiveness of extreme-scale visual analytics. We would consider the challenges of large-scale visualization in IDS in future work.

References

Abdullah, K., Lee, C., Conti, G., & Copeland, J. A. (2005). Visualizing network data for intrusion detection. In *Workshop on Information Assurance and Security United States Military Academy* (pp. 100 – 107).

Adhy, S., Noranita, B., Kusumaningrum, R., Wirawan, P. W., Prasetya, D. D., & Zaki, F. (2017). Usability testing of weather monitoring on a web application. In *1st International Conference on Informatics and Computational Sciences (ICICoS)* (pp. 131–136).

Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, *60*, 19 – 31.

An, N., & Weber, S. (2017). Impact of sample size on false alarm and missed detection rates in pca-based anomaly detection. In *51st Annual Conference on Information Sciences and Systems (CISS)* (pp. 1 – 6).

Bamakan, S. M. H., Wang, H., & Shi, Y. (2017). Ramp loss k-support vector classification-regression; a robust and sparse multi-class approach to the intrusion detection problem. *Knowledge-Based Systems*, *126*, 113–126.

Bao, H., & Wang, Y. (2016). A c-svm based anomaly detection method for multi-dimensional sequence over data stream. In *IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS)* (pp. 948 – 955).

Bi, J., Zhang, K., & Cheng, X. (2009). Intrusion detection based on rbf neural network. In *International Symposium on Information Engineering and Electronic Commerce* (pp. 357 – 360).

Brooke, J. et al. (1996). Sus-a quick and dirty usability scale. *Usability evaluation in industry*, *189*, 4–7.

Chen, G., Zhang, X., & Li, Z. J. W. F. (2015). Robust support vector data description for outlier detection with noise or uncertain data. *Knowledge-Based Systems*, *90*, 129 – 137.

Chen, M.-H., Chang, P.-C., & Wu, J.-L. (2016). a population-baed incremental learning approach with artificial immune system for network intrusion detection. *Engineering Applications of Artificial Intelligence*, *51*, 171 – 181.

Corchado, E., & Álvaro Herrero (2011). Neural visualization of network traffic data for intrusion detection. *Applied Soft Computing*, *11*, 2042 – 2056.

Devi, R., Jha, R. K., Gupta, A., Jain, S., & Kumar, P. (2017). Implementation of intrusion detection system using adaptive neuro-fuzzy inference system for 5g wireless communication network. *AEU - International Journal of Electronics and Communications*, *74*, 94 – 106.

Draper-Gil, G., Lashkari, A. H., Mamun, M. S. I., & Ghorbani, A. A. (2016). Characterization of encrypted and vpn traffic using time-related. In *In Proceedings of the 2nd International Conference on Information Systems Security and Privacy(ICISSP)* (pp. 407–414).

Elhenawy, I., Riad, A. E.-D., Hassan, A., & Awadallah, N. (2011). Visualization techniques for intrusion detection - a survey. *International Journal of Computer Science and Engineering Survey (IJCSES)*, *2*, 107 – 116.

Etoty, R. E., & Erbacher, R. F. (2014). *A Survey of Visualization Tools Assessed for Anomaly-Based Intrusion Detection Analysis*. Technical Report No. ARL-TR-6891. Army Research Lab Adelphi MD Computational and Information Science Directorate.

Faigl, J., & Hollinger, G. A. (2017). Autonomous data collection using a self-organizing map. *IEEE Transactions on Neural Networks and Learning Systems*, *PP*, 1 – 13.

Feizollah, A., Shamshirband, S., Anuar, N. B., Salleh, R., & Kiah", M. L. M. (2013). Anomaly detection using cooperative fuzzy logic controller. In *Intelligent Robotics Systems: Inspiring the NEXT: 16th FIRA RoboWorld Congress, FIRA 2013, Kuala Lumpur, Malaysia* (pp. 220 – 231). Springer Berlin Heidelberg.

Fränti, P. (2015). Clustering datasets.

Hachmi, F., Boujenfa, K., & Limam, M. (2015). A three-stage process to detect outliers and false positives generated by intrusion detection systems. In *IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing* (pp. 1749 – 1755).

Hamed, T., Ernst, J. B., & Kremer, S. C. (2018). A survey and taxonomy on data and pre-processing techniques of intrusion detection systems. In *Computer and Network Security Essentials* (pp. 113–134). Springer.

He, Z., Xu, X., & Deng, S. (2003). Discovering cluster-based local outliers. *Pattern Recognition Letters*, *24*, 1641 – 1650.

Herrero, Á., & Corchado, E. (2011). *Mobile Hybrid Intrusion Detection: The MOVICAB-IDS System*. Studies in Computational Intelligence. Springer.

Hodo, E., Bellekens, X., Hamilton, A., Dubouilh, P.-L., Iorkyase, E., Tachtatzis, C., & Atkinson, R. (2016). Threat analysis of iot networks using artificial neural network intrusion detection system. In *International Symposium on Networks, Computers and Communications (ISNCC)* (pp. 1 – 6).

De la Hoz, E., de la Hoz, E., Ortiz, A., Ortega, J., & Martínez-Álvarez, A. (2014). Feature selection by multi-objective optimisation: Application to network anomaly detection by hierarchical self-organising maps. *Knowledge-Based Systems*, *71*, 322 – 338.

la Hoz, E. D., Hoz, E. D. L., Ortiz, A., Ortega, J., & Prieto, B. (2015). Pca filtering and probabilistic som for network intrusion detection. *Neurocomputing*, *164*, 71 – 81.

Ibrahim, J. M., Karami, A., & Jafari, F. (2017). A secure smart home using internet-of-things. In *Proceedings of the 9th International Conference on Information Management and Engineering* (pp. 69–74). ACM.

Jabez, J., & Muthukumar, B. (2015). Intrusion detection system (ids): Anomaly detection using outlier detection approach. *Procedia Computer Science*, *48*, 338 – 346.

Jach, A., & Kokoszka, P. (2008). Wavelet-based confidence intervals for the self-similarity parameter. *Journal of Statistical Computation and Simulation*, *78*, 1181 – 1200.

Jia, S.-Y., Jeongb, B.-K., Choia, S., & Jeong, D. H. (2016). A multi-level intrusion detection method for abnormal network behaviors. *Journal of Network and Computer Applications*, *62*, 9 – 17.

Jirapummin, C., Wattanapongsakorn, N., & Kanthamanon, P. (2002). Hybrid neural networks for intrusion detection system. In *Proc. of ITC–CSCC* (pp. 928 – 931).

Kabir, E., Hu, J., Wang, H., & Zhuo, G. (2017). A novel statistical technique for intrusion detection systems. *Future Generation Computer Systems*, .

Karami, A. (2015a). Accpndn: Adaptive congestion control protocol in named data networking by learning capacities using optimized time-lagged feedforward neural network. *Journal of Network and Computer Applications*, *56*, 1 – 18.

Karami, A. (2015b). A framework for uncertainty-aware visual analytics in big data. In *Proceedings of the 3rd International Workshop on Artificial Intelligence and Cognition* (pp. 146 – 155). CEUR-WS.

Karami, A., & Guerrero-Zapata, M. (2014). Mining and visualizing uncertain data objects and named data networking traffics by fuzzy self-organizing map. In *Second International Workshop on Artificial Intelligence and Cognition* (pp. 156 – 163). CEUR-WS.

Karami, A., & Guerrero-Zapata, M. (2015a). An anfis-based cache replacement method for mitigating cache pollution attacks in named data networking. *Computer Networks*, *80*, 51 – 65.

Karami, A., & Guerrero-Zapata, M. (2015b). A fuzzy anomaly detection system based on hybrid pso-kmeans algorithm in content-centric networks. *Neurocomputing*, *149*, 1253 – 1269.

Karami, A., & Guerrero-Zapata, M. (2015c). A hybrid multiobjective rbf-pso method for mitigating dos attacks in named data networking. *Neurocomputing*, *151*, 1262 – 1282.

Karami, A., & Johansson, R. (2014a). Choosing dbscan parameters automatically using differential evolution. *International Journal of Computer Applications*, *91*, 1 – 14.

Karami, A., & Johansson, R. (2014b). Utilization of multi attribute decision making techniques to integrate automatic and manual ranking of options. *Journal of information science and engineering*, *30*, 519–534.

Kayacik, H. G., & Zincir-Heywood, A. N. (2006). Using self-organizing maps to build an attack map for forensic analysis. In *Proceedings of the 2006 International Conference on Privacy, Security and Trust: Bridge the Gap Between PST Technologies and Business Services* (p. 33).

Khalid, C., Zyad, E., & Mohammed, B. (2015). Network intrusion detection system using l1-norm pca. In *11th International Conference on Information Assurance and Security (IAS)* (pp. 118 – 122).

Kiziloren, T., & Germen, E. (2009). Anomaly detection with self-organizing maps and effects of principal component analysis on feature vectors. In *Fifth International Conference on Natural Computation* (pp. 509 – 513). volume 2.

Kohonen, T., & Honkela, T. (2007). Kohonen network. Scholarpedia 2.1, 1568.

Lashkari, A. H., Kadir, A. F. A., Gonzalez, H., Mbah, K. F., & Ghorbani, A. A. (2017). Towards a network-based framework for android malware detection and characterization. In *In Proceeding of the 15th International Conference on Privacy, Security and Trust*.

Luo, B., & Xia, J. (2014). A novel intrusion detection system based on feature generation with visualization strategy. *Expert Systems with Applications*, *41*, 4139 – 4147.

Marr, P. (2015). Spss mini-lab: Outliers. `http://webspace.ship.edu/pgmarr/Geo441/Lectures/OPT%201%20-%20Outlier%20Detection.pdf`, online; accessed 20-May-2017.

Mishra, P., Pilli, E. S., Varadharajan, V., & Tupakula, U. (2017). Psi-netvisor: Program semantic aware intrusion detection at network and hypervisor layer in cloud. *Journal of Intelligent & Fuzzy Systems*, *32*, 2909–2921.

Mitrokotsa, A., & Douligeris, C. (2005). Detecting denial of service attacks using emergent self-organizing maps. In *Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology* (pp. 375 – 380).

Moustafa, N., & Slay, J. (2015). Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In *Military Communications and Information Systems Conference (MilCIS)* (pp. 1–6).

Moustafa, N., Slay, J., & Creech, G. (2017). Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks. *IEEE Transactions on Big Data*, .

NSL-KDD (2014). NSL-KDD dataset. `http://www.unb.ca/cic/research/datasets/nsl.html`, online; accessed 5-May-2017.

Obimbo, C., & Jones, M. (2012). Applying variable coefficient functions to self-organizing feature maps for network intrusion detection on the 1999 kdd cup dataset. *Procedia Computer Science*, *8*, 333 – 337.

Olszewski, D. (2014). Fraud detection using self-organizing map visualizing the user profiles. *Knowledge-Based systems*, *70*, 324 – 334.

Pachghare, V., Kulkarni, P., & Nikam, D. M. (2009). Intrusion detection system using self organizing maps. In *International Conference on Intelligent Agent & Multi-Agent Systems (IAMA)* (pp. 1 – 5).

Powers, S. T., & He, J. (2008). A hybrid artificial immune system and self organising map for network intrusion detection. *Information Sciences*, *178*, 3024 – 3042.

Powersa, S. T., & He, J. (2008). A hybrid artificial immune system and self organising map for network intrusion detection. *Information Sciences*, *178*, 3024 – 3042.

Raman, M. G., Somu, N., Kirthivasan, K., Liscano, R., & Sriram, V. S. (2017). An efficient intrusion detection system based on hypergraph-genetic algorithm for parameter optimization and feature selection in support vector machine. *Knowledge-Based Systems*, *134*, 1–12.

Salim, I., & Razak, T. A. (2016). A study on ids for preventing denial of service attack using outliers techniques. In *IEEE International Conference on Engineering and Technology (ICETECH)* (pp. 768 – 775).

Shakhatreh, A. Y. I., & Bakar, K. A. (2011). A review of clustering techniques based on machine learning approach in intrusion detection systems. *International Journal of Computer Science Issues (IJCSI)*, *8*.

da Silva, L. E. B., & Wunsch, D. C. (2017). An information-theoretic-cluster visualization for self-organizing maps. *IEEE Transactions on Neural Networks and Learning Systems*, *PP*, 1 – 19.

Subba, B., Biswas, S., & Karmakar, S. (2016). A neural network based system for intrusion detection and attack classification. In *Twenty Second National Conference on Communication (NCC)* (pp. 1 – 6).

Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the kdd cup99 data set. In *IEEE International Conference on Computational Intelligence for Security and Defense Applications* (pp. 53 – 58).

UNB (2018). Canadian institute for cybersecurity datasets. `http://http://www.unb.ca/cic/datasets/index.html`, online; accessed 1-May-2018.

dong Wang, C., feng Yu nd Huai-bin Wang, H., & Liu, K. (2007). Som-based anomaly intrusion detection system. In *Embedded and Ubiquitous Computing: International Conference, Taipei, Taiwan* (pp. 356 – 366). Springer Berlin Heidelberg.

Wang, H., Gu, J., & Wang, S. (2017). An effective intrusion detection framework based on svm with feature augmentation. *Knowledge-Based Systems*, *136*, 130–139.

Yi, J., Huang, D., Fu, S., He, H., & Li, T. (2016). Optimized relative transformation matrix using bacterial foraging algorithm for process fault detection. *IEEE Transactions on Industrial Electronics*, *63*, 2595 – 2605.

Yu, K. F., Yan, F., & Lin, Z. J. (2010). Research of outlier mining based adaptive intrusion detection techniques. In *Third International Conference on Knowledge Discovery and Data Mining* (pp. 552 – 555).

Zhang, W., Zhang, Y., Bai, X., Liu, J., Zeng, D., & Qiu, T. (2016). A robust fuzzy tree method with outlier detection for combustion models and optimization. *Chemometrics and Intelligent Laboratory Systems*, *158*, 130 – 137.