Monshizadeh, Mehrnoosh; Khatri, Vikramajeet; Kantola, Raimo; Yan, Zheng

A deep density based and self-determining clustering approach to label unknown traffic

Published in:
Journal of Network and Computer Applications

DOI:
10.1016/j.jnca.2022.103513

Published: 01/11/2022

Document Version
Publisher's PDF, also known as Version of record

Journal of Network and Computer Applications 207 (2022) 103513

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

A deep density based and self-determining clustering approach to label unknown traffic

Mehrnoosh Monshizadeh [a,b,*], Vikramajeet Khatri [c], Raimo Kantola [b], Zheng Yan [b,d]

[a] Nokia Bell Labs, France
[b] Department of Comnet, Aalto University, Finland
[c] Nokia Bell Labs, Finland
[d] The State Key Lab of ISN, Xidian University, China

ARTICLE INFO

Keywords:
Intrusion detection
Data mining
Machine Learning
Network security
Network traffic

ABSTRACT

Analyzing non-labeled data is a major concern in the field of intrusion detection as the attack clusters are continuously evolving which are unknown for the system. Many studies have been conducted on different techniques such as clustering to solve this issue. Consequently, in this paper the clustering techniques are applied based on the packets' similarity to categorize unknown traffic. After clustering is done by the proposed architecture, the security investigator analyzes one packet from each cluster (instead of thousands of packets) and generalize the result of analysis to all packets belonging to the cluster. The proposed architecture, namely Associated Density Based Clustering (ADBC) applies multiple unsupervised algorithms and a co-association matrix to detect attack clusters of any shape as long as they have density-connected elements. Furthermore, the architecture automatically determines the best number of clusters in order to categorize non-labeled data. The performance of proposed architecture is evaluated based on the various metrics, while its generalization capability is tested with three publicly available datasets.

1. Introduction

In order to protect networks against cyber-attacks, Intrusion Detection Systems (IDS) are designed to detect malicious traffic. The IDS monitors network traffic, compares it with predefined patterns, identifies suspicious activities and informs security investigator about deviations. The IDSs mainly rely on static signature from known attacks or the fixed pattern of the safe traffic. Even though some of current IDS techniques look promising with comprehensive features, still they lack intelligent and dynamic mechanism to detect unknown attacks (Top 10, 2022).

The signature-based methods build a model of normal traffic behavior and compare the input data with the expected normal traffic; if the input is different, a signature would be generated for the attack class to train the model with new information and for the next time detection. Yet, this approach may have high false positive rate since it compares the input traffic with an expected model (Pietro et al., 2016). Other IDSs are more intelligent and apply ML algorithms in their structures. Some of these studies combine the ML algorithms with honeypot to collect labeled malicious traffic. However, these techniques only rely on the received attacks by honeypot; in addition, there is a risk that honeypot node is identified to the attacker (Yadav et al., 2016).

Furthermore, studies that utilize ML for intrusion detection, rarely evaluate the unknown traffic analysis and even if the issue is addressed, a high detection rate is not achieved (Yousefi-Azar et al., 2017; Sun et al., 2020; Fred and Jain, 2005; Wang et al., 2020; Xue-yong et al., 2010; Dong et al., 2019; Shakya and Makwana, 2017; Turner and Joseph, 2017; Yang et al., 2020; Bedi et al., 2021). These solutions may improve the detection rate in overall and only for a specific dataset while their performance varies considerably for another dataset and per each type of attack (Muhlenbach and Lallich, 2009; Kim et al., 2004; Jongsuebsuk et al., 2013; Lin et al., 2014). On the other hand, they present limited evaluation rather than a detailed analysis (Sun et al., 2018; Yang et al., 2019; Wei et al., 2020).

On the other hand, Machine Learning (ML) techniques are introduced as a complementary to the IDS to dynamically identify the relevant data of interest and intelligently figure out the security threats. Based on the learning process, ML algorithms are divided in different categories such as supervised and unsupervised techniques (Phadke et al., 2019).

In supervised techniques, the labeled data is used to train the model; classification methods belong to this category. For IDS application,

* Corresponding author at: Department of Comnet, Aalto University, Finland.
E-mail addresses: mehrnoosh.monshizadeh@nokia-bell-labs.com, mehrnoosh.monshizadeh@aalto.fi (M. Monshizadeh), vikramajeet.khatri@nokia-bell-labs.com (V. Khatri), raimo.kantola@aalto.fi (R. Kantola), zheng.yan@aalto.fi (Z. Yan).

https://doi.org/10.1016/j.jnca.2022.103513
Received 22 July 2022; Received in revised form 24 August 2022; Accepted 6 September 2022
Available online 14 September 2022

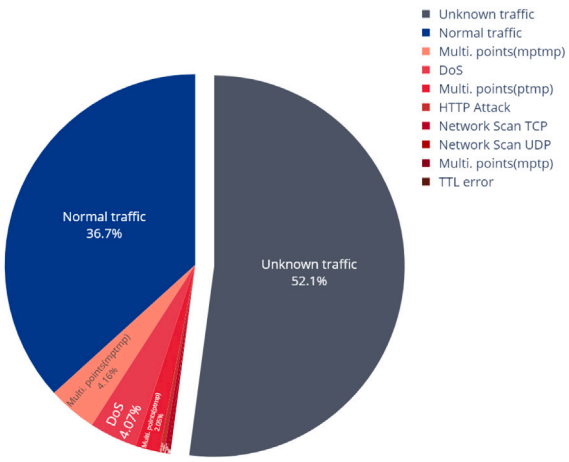

Fig. 1. MAWILab-2018 data distribution.

these algorithms can classify various types of known cyber-attacks. The classifiers are first trained with labeled network traffic including benign and attack packets whereas they can be used later for testing and to identify malicious packets based on the trained model. In unsupervised techniques, the training data is not labeled; the clustering algorithms belong to this category. In the ML-based IDS, where the algorithms have no prior information about input traffic the clustering methods will be employed to categorize unknown traffic. These algorithms divide unknown traffic in distinct clusters in such that packets with similar properties go to the same group. These clusters will be further used by a security investigator to detect malicious packets and generalize the investigation to the entire cluster (Mehmood and Md Rais, 2016; Portela et al., 2019).

It should be noted that a data set (e.g., network traffic data set) may include several features that cause performance degradation in detection process. To overcome this problem, feature selection or extraction methods such as Support Vector Machine (SVM) and Variational Auto Encoders (VAEs) are used to select a smaller number of features and reduce the dimension of the dataset. The selected or extracted features will be provided to the classification and clustering algorithms in order to help them in classifying the known attacks and clustering unknown traffic (Yousefi-Azar et al., 2017; Monshizadeh et al., 2018; Li et al., 2020).

However, in the field of machine learning, non-labeled data analysis is one of the well-known challenges and many studies have been conducted on different techniques to solve this issue. On the other hand, with the emergence of unknown attacks, current intrusion detection systems suffer from a low detection rate. In real-life scenarios as shown in Fig. 1 for MAWILab-2018 dataset, a considerable amount of incoming data does not belong to any known class and subsequently, leads to class imbalance and high false positive and negative ratios.

On the other hand, annotating large datasets is very costly and hence only a few examples can be labeled manually. In addition, for unknown traffic, dividing data into classes without having information on the nature of the traffic is challenging. Therefore, clustering methods are employed to gain some insight into the structure of the data. However, clustering techniques also have some drawbacks. In intrusion detection, clusters can appear with different sizes, shapes, data sparseness, and overlapping degrees. Even if there are many clustering algorithms, none of them is able to identify all the cluster forms and structures encountered in real-life scenarios. Therefore, a more sophisticated unsupervised approach, which automatically determines the best number of clusters would be a major goal in analyzing non-labeled data.

In general, the main challenge of clustering algorithms is that clusters can appear with different sizes, density, and degree of separation (Fred and Jain, 2005). It is therefore difficult to select an algorithm that performs well on a particular dataset, but also to adjust the different hyperparameters of that same algorithm so that it can be generalized.

Following the mentioned challenges and in order to efficiently detect unknown attacks, this paper proposes a density-based method, namely Associated Density Based Clustering (ADBC).

The method utilizes several unsupervised algorithms in conjunction with a Density-Based Spatial Clustering of Applications with Noise (DBSCAN) in order to categorize unknown traffic in various clusters. The hypothesis is that if two packets belong to the same attack type, it is more likely that they fall into the same cluster when any clustering algorithm is applied with any hyperparameters. In other words, when several clustering algorithms are applied, the more two samples fall into the same cluster, the more likely they belong to the same attack type. Specifically, given a set of packets, a new distance measure between data points is calculated based on the mentioned assumption and is represented in a co-association matrix, which is later used for DBSCAN clustering.

Overall, the contribution of this paper is to introduce an efficient method for clustering unknown traffic with the following characteristics:

- Categorizing unknown data into distinct clusters in such the packets belong to one cluster are similar to each other and dissimilar to the packets belong to other clusters. This model achieves a good homogeneity score for various datasets with different attacks, which proves that each cluster contains mainly members of a single attack class, and a high silhouette coefficient score which shows that the clusters are well defined and have minimal overlap.
- Introducing a new metric that helps to achieve high silhouette score by changing the feature space. The new similarity metric is different from Euclidean distance, where the distance between data points reflects the similarity between packets. The new similarity matrix will be used to create a co-association matrix that will be fed to DBSCAN as input.
- Evaluating proposed architecture performance robustness against three datasets that have different attacks and sizes.
- Cluster analysis generalization in a way that few packets from each generated cluster are analyzed, the malicious packets are identified, and the result of analysis is generalized to the entire cluster. This approach saves security investigator time and resources.

The rest of the paper is organized as follows. Section 2 provides motivation for the paper and reviews some studies that applied unsupervised techniques such as DBSCAN mainly for intrusion detection application and briefly addresses their short comes. Section 3 introduces the ADBC architecture as well the algorithms that are required for the experimental results. In Section 4, the implementation settings that are needed for further experiments are presented. Section 5 discusses the experimental results for ADBC architecture as well for other clustering methods that are implemented by authors for comparison purpose. In Section 6, the paper provides a comparison between ADBC and the state-of-the-art methods along with the scope of future research. Finally, the conclusion is drawn in Section 7.

2. Motivation and related work

Several studies applied unsupervised techniques to analyze unlabeled datasets and to cluster the datapoints based on their similarities. Even though, these studies applied various optimized version of DBSCAN to provide a good silhouette score, yet their achieved

performance considerably depends on the datasets. In addition, hyperparameter settings are changing based on the different datasets. Furthermore, majority of the prior art methods are applied for different application than IDS and their employed datasets are dissimilar to network traffic dataset. On the other hand, their testing conditions are distinct and their presented results lack a thorough evaluation of clusters quality and analysis. Therefore, this paper proposes an efficient density-based method to categorize unknown traffic into various clusters. The purpose of this architecture is to provide a high silhouette score regardless of applied dataset. Furthermore, the architecture aims to provide insights based on the created clusters to a cybersecurity investigator and assist in detecting unknown attacks with reduction of time and computation resources during analysis.

In order to analyze unknown data or detect anomalies, several studies have applied various clustering techniques such as DBSCAN and its variants. DBSCAN do not need to have number of clusters predefined, in addition it has the advantage of finding clusters of any shape, as long as their elements are density-connected. This advantage is important when dealing with a clustering problem of unknown data where the shape of the clusters is uncertain.

In order to divide data into the groups based on the similarity of characteristics, several studies have applied various clustering techniques such as DBSCAN or a combination of unsupervised algorithms. DBSCAN do not need to have number of clusters predefined, in addition it has the advantage of finding clusters of any shape, as long as their elements are density-connected. This advantage is important when dealing with a clustering problem of unknown data where the shape of clusters is uncertain. In order to evaluate DBSCAN performance in related articles, mainly the silhouette score is presented to show the level of clusters distinction. However, in these studies achieved silhouette score is considerably depends on the applied dataset.

Muhlenbach and Lallich (2009) proposed a graph-based clustering (GBC) method that defines the best number of clusters and detects anomalies. This study is based on a neighborhood graph and regions of influence approach. The method is evaluated based on the Dunn's, Silhouette, and the Davies–Bouldin metrics and for five datasets from the University of California Irvine (UCI) Repository, three artificial data sets (test-2c-2o, test-random, yin-yang), and Ruspini dataset. This method is limited to hard clustering and fails if there is recovery between the different clusters. On the other hand, the maximum silhouette score depends on the dataset (achieved values are between 0.584 and 0.910 depending on the dataset). Similarly, for Dunn's and Davies–Bouldin's indices, the maximum value depends on the dataset.

Turner and Joseph (2017) combined static rule extraction mechanism with machine learning in order to detect intrusion. The study performs cluster analyses applying a k-means algorithm, a hierarchical agglomerative clustering algorithm, and a density-based clustering algorithm independently and for each ruleset. The silhouette metric is used to evaluate the method performance. However, few data and a small number of clusters (simple clusters) have been used to test the method's performance. And consequently, the authors achieved a good silhouette score, since this metric depends considerably on the nature of the data.

Sun et al. (2020) applied DBSCAN algorithm to cluster unknown protocol messages into classes with different formats. The performance of the method is evaluated under the direction of supervised and unsupervised cluster quality metrics such as the Silhouette Coefficient and Dunn Index in order to decrease manual intervention. However, the maximum reached by the Silhouette Coefficient and Dunn Index is 0.67 and 0.77.

Xue-yong et al. (2010) introduced an improved DBSCAN algorithm to detect intrusion. The proposed model improves the formula for calculating distances. The presented clusters merging experimental results proves that this method reduces the false-negative rate and improves the performance of intrusion detection systems. For the experiment, the study applies a feature selection method with cross-validation on KDD Cup 1999 dataset. In order to evaluate the model, metrics such as false ratio (FR) and detection rates have been used in this paper. Though, the model applied clustering technique and discussed about hyperparameter (such as *epsilon* and *MinPts*) setting, the study did not provide any analysis on clusters and unsupervised metric such as silhouette score.

Dong et al. (2019) proposed a model combining k-means with DBSCAN for intrusion detection. The combination solves the k-means' sensitivity to initial clustering centers and noise points and reduces the influence of fixed neighborhood radius in DBSCAN. The experiments are done on the NSL-KDD dataset applying false positive ratio and supervised validation. Though, the model applied clustering technique, the study did not provide any analysis on clusters and unsupervised metric such as silhouette score.

Shakya and Makwana (2017) presented a combination of Sequential minimal Optimization Classifier (SMO), k-means clustering, and DBSCAN to improve detection rate. However, DBSCAN is only used for noise reduction and not for clustering. The study used KDD dataset, feature selection method and chose accuracy as an evaluation metric.

Sabottke et al. (2019) applied DBSCAN clustering for detecting attacks in web server logs. Three months of web server logs are collected for this study. Potentially malicious sessions lie either in the DBSCAN noise or in the territory of a malicious activity cluster. If samples that are labeled as noise in the DBSCAN clustering are intended to be analyzed as potential threats by human analysts, then it is consequently important to parametrize DBSCAN clustering in order to reduce the number of noise samples. To evaluate clustering performance, the authors calculate the silhouette score. A high silhouette score indicates that the clusters are compact (each point is similar to the other points in its cluster) and well separated (each point is unlike the points in other clusters). The authors manually examined whether those clusters corresponded to keywords associated with attacks (i.e., "login", "admin", and database insertion keywords). These keywords were chosen based on feedback from the human experiment. Finally, they manually analyze the clusters to identify normal and attack behaviors. The clustering achieves a silhouette score of 0.99. However, the obtained clusters are composed of different types of attacks (mixed clusters) which makes the analysis difficult for security investigator. Furthermore, with this method the result of packet (per cluster) analysis cannot be generalized to the entire cluster.

Dockhorn et al. (2015) presented an algorithm based on the combination of two hierarchical versions of DBSCAN (HDBSCAN) for automatically determining locally optimal parameter settings, which is achieved by fixing one parameter and iterating through possible values of the second parameter. The study compares the measures of edge-correlation and silhouette coefficient. The silhouette coefficient and edge correlation reported few noise points and scored not as high as the density-based silhouette coefficient (whose formula is not mentioned). The authors compared the use of silhouette coefficient and edge correlation as two such measures. However, both measures prefer convex-shaped clusters and cannot adapt to all cluster shapes produced by DBSCAN. Therefore, they proposed a density-based interpretation of the silhouette coefficient, which rates the density of a cluster as the minimal value and sets it in relation to the minimal ε distance to the next cluster. In contrast to the original silhouette coefficient, this optimization criterion can adapt to clusters of arbitrary shapes. The comparison of internal validation measures revealed that the density-based silhouette coefficient performed best in most experiments. However, in the search for convex-shaped clusters, silhouette coefficient and edge correlation perform better than the proposed density-based silhouette coefficient.

Fred and Jain (2005) proposed a method based on the concept of evidence accumulation clustering. They combine various clustering partitions of a given dataset to define a partition that is better than the original partitions. This method either applies different clustering algorithms or the same clustering algorithm with different parameters. However, the model is applied on image datasets and not for intrusion detection application. The current study is motivated by this concept.

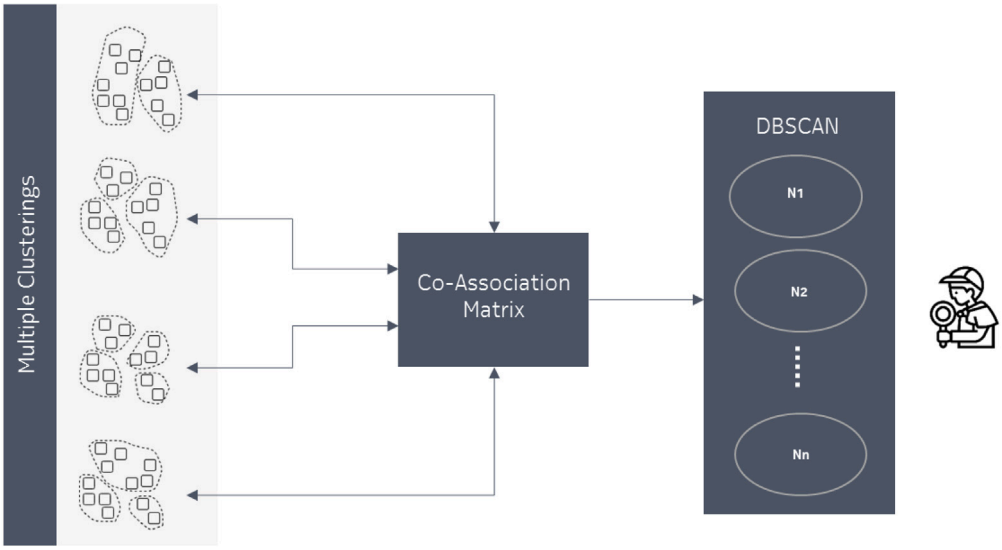

Fig. 2. Architecture of Associated Density Based Clustering (ADBC).

3. Architecture

The proposed architecture is a complementary part of the Hybrid Anomaly Detection Model (HADM) platform, which has been explained and previously published in three papers (Monshizadeh et al., 2018, 2019, 2021). The HADM comprises various machine learning (ML) algorithms to filter network traffic and identity malicious activities on the network. The HADM applies classification algorithms to detect predefined known attacks and clustering algorithms to identify unknown attacks. Consequently, current paper focuses on applying clustering techniques for unknown traffic analysis. As shown in Fig. 2, the architecture comprises several algorithms and a co-association matrix.

As it was discussed in Section 2, in order to analyze non labeled datasets in an unsupervised way, several studies have employed clustering algorithms such as DBSCAN or a combination of DBSCAN with other linear algorithms namely SVM or deep learning algorithms such as VAEs. Therefore, in order to prove the proposed architecture superiority to prior art, current paper, compares performance of ADBC against prior art algorithms. For this purpose, the prior art methods are implemented by authors in the same testing environment and by employing similar datasets that are applied to ADBC. Consequently, in this section the prior art algorithms beside the applied algorithms in ADBC are described. Furthermore, the evaluation metrics are presented.

3.1. Applied algorithms

The applied algorithms, their internal architecture and parameters are explained below.

VAE. This is an unsupervised Latent-variable-based deep generative model. VAE comprises two neural networks: an encoder network and a decoder network. VAE architecture can be seen in Fig. 3.

The encoder is a neural network that inputs a data point x and outputs a latent representation z. This latent variable z belongs to a latent space of lower dimension than the input space. The encoder has weights and biases ϕ. The distribution of the latent variable z and the encoder are denoted as $q(z|x;\phi)$.

The decoder is a neural network that receives the latent variable z as input and reconstructs $\hat{x}$ from the probability distribution $p(x|z;\theta)$ with θ expressing the weights and biases of the decoder.

The loss function is a negative loglikelihood with a regularizer:

$$D_{KL}(q(z|x;\phi)) \parallel (p(z;\theta)) - E_{z\sim q\phi}[\log p(x|z;\theta)] \tag{1}$$

$p(z)$ is the expected distribution (the prior) of z which is specified as a standard normal distribution with mean 0 and variance 1.

An observation x is assumed to be distributed according to $p(x|z;\theta^*)$, where the decoder takes as input z and outputs $p(x|z;\theta)$. The choice of this distribution depends on the type of data. In this paper, a multivariate Gaussian distribution is applied. In order to estimate θ to get the closest possible $p(x|\theta)$ to the true data distribution, the decoder can be fit by maximizing the marginal likelihood as seen in (2):

$$p(x;\theta) = \int p(x|z;\theta)p(z)dz \tag{2}$$

However, this likelihood cannot be evaluated or approximated as it is intractable. Even trying to use $p(z|x;\theta)$ will not solve this problem because $p(z|x;\theta)$ is intractable too.

The variational autoencoder model solves this problem by using variational inference which uses majorization-minimization principles to solve this optimization problem. The approach is to approximate $p(z|x;\theta)$ using an encoder network and to use this approximation to estimate a lower bound on the marginal log-likelihood. As a result, the model will learn its parameters by maximizing this lower bound (the Evidence Lower Bound).

Hence, $q(z|x;\phi)$ is considered as the approximating distribution of $p(z|x;\theta)$ where $q(z|x)$ is a multivariate Gaussian distribution. It is parametrized with the encoder that takes as input x and outputs $q(z|x;\phi)$.

The marginal log-likelihood of an observation x and for any variational distribution $q(z|x;\phi)$ over the latent variables z can be expressed as follows:

$$log p(x;\theta) = L(x;\phi,\theta) + D_{KL}(q(z|x;\phi) \parallel p(z|x;\theta)) \tag{3}$$

where $L(x;\phi,\theta)$ represents the Evidence Lower Bound (ELBO) as seen in (4):

$$L(x;\phi,\theta) = E_{z\sim q\phi}[\log p(x|z;\theta) - \log q(x|x;\phi)] \tag{4}$$

As the Kullback–Leibler (KL) divergence is non-negative: $log\ p(x;\theta) \geq L(x;\phi,\theta)$ with equality only when $q(z|x;\phi) = p(z|x;\theta)$. Therefore, the objective function maximized in variational inference is:

$$\begin{aligned} L(x;\phi,\theta) &= E_{z\sim q\phi}[\log p(x|z;\theta) - \log q(z|x;\phi)] \\ &= -D_{KL}(q(z|x;\phi) \parallel p(z;\theta)) + E_{z\sim q\phi}[\log p(x|z;\theta)] \end{aligned} \tag{5}$$

As it is shown in (5), the ELBO has two terms. The first is a regularization term called the KL divergence and it ensures that the encoder stays close to the prior. The second is the reconstruction term.

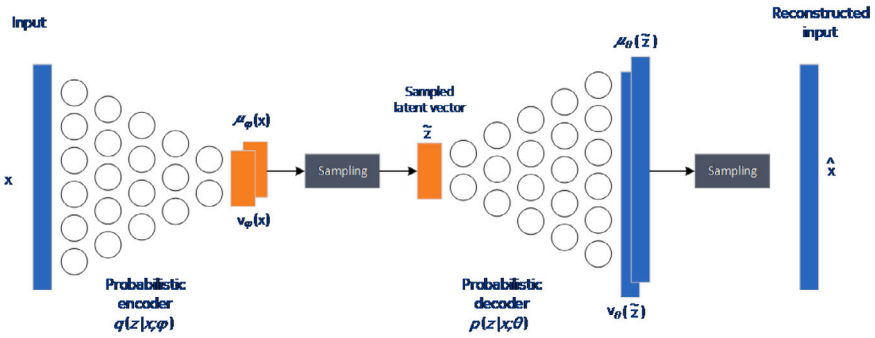

Fig. 3. Variational Autoencoder architecture.

Even if an analytical expression of the ELBO is missed, having an approximation of it using Monte Carlo estimate is possible (Kingma and Welling, 2014).

Lambda-VAE. The architecture of the lambda VAE which is a variation of a VAE was introduced by the authors of Wang et al. (2019). It modifies the KL divergence of the original VAE which is responsible for encouraging all latent embeddings to cluster around the origin. In fact, this behavior is not desirable in IDS application because it can increase class overlapping in the latent space.

Hence, the idea behind lambda VAE is to change the prior $p(z)$ used in the KL divergence $KL(q\theta\ (z|xi)||p(z))$ depending on the label so that each class is placed on a separate dimension in the latent space.

For example, if the input sample has a label of 2 and the dimension of the latent vector z (z has the same definition as in the VAE) is 5, the mean of the encoded latent vector of this sample will be forced to $[0,0,\lambda,0,0]$. In this way, all samples with a label of 2 will be clustered around the direction [0,0,1,0,0] of the latent space. The vector $[0,0,\lambda,0,0]$ is an example of a lambda-hot encoding vector.

This behavior is enforced by replacing the prior $p(z)$ in the KL divergence with the lambda-hot encoding vector. So, instead of including $KL(q\theta\ (z|xi)||p(z))$ in the loss of the VAE, it is replaced with $KL(q\theta\ (z|xi)||v(x))$ where the $v(x)$ is the lambda-hot encoding vector of the input sample x.

This method was proven to enforce clusters in the latent space for the examples presented by the authors of Wang et al. (2019). However, this was not the case for network traffic. Furthermore, the λ is a hyperparameter that needs to be set in a way that pushes each cluster far from the others in the latent space in order to avoid overlapping between clusters. For the current paper experiments, lambda is set to 5 after testing other values (Wang et al., 2019).

SVMonline. Incremental SVM calculates the loss and retrains linear SVM in every batch using stochastic gradient descent. It assigns SVM weights to each feature and selects those with the highest absolute value as the best discriminative features. Although SVMonline relies on the linear dependency of features and labels as in F-Score, it is more robust than F-Score, since it splits the dataset into small batches and calculates the average of model coefficients that further increases the robustness (Chen, 2003).

k-means. In this algorithm n data items are divided into K clusters while new data entry would be assigned to the cluster with the highest similarity; therefore, it has lower similarity to other clusters. The similarity is defined based on the distance function, which is typically a metric. k-means algorithm has three steps: assigning the new data to the closest cluster, re-estimating the mean, and finally iteration or normalization of data (Fred and Jain, 2005; Monshizadeh and Yan, 2014).

For observations $x_1, x_2, \ldots, x_n$ where each observation is a d-dimensional real vector and for Euclidean distance function, k-means clustering aims to partition the n observations into $k(\leq n)$ sets $S = \{S_1, S_2, \ldots, S_k\}$ so as to minimize the within-cluster sum of squares.

$$arg_s\ min = \sum_{i=1}^{k} \sum_{x \in S_i} \| x - \mu_i \|^2 = arg_s\ min \sum_{i=1}^{k} |S_i| Var\ S_i \tag{6}$$

where μ_i is the mean of points in S_i. This is equivalent to minimizing the pairwise squared deviations of points in the same cluster.

$$arg_s\ min = \sum_{i=1}^{k} \frac{1}{2|S_i|} \sum_{x,y \in S_i} \| x - y \|^2 \tag{7}$$

Co-association matrix. This technique is similar to a voting mechanism that is used to combine the clustering results, leading to a new measure of similarity between samples or data points (Fred and Jain, 2005).

The idea behind this voting is that data points belonging to the same category (having the same nature) are very likely to be assigned to the same cluster in different data partitions or data clustering.

Taking the co-occurrences of pairs of samples or packets in the same cluster as votes for their association, the N data partitions (clusters) of n samples are mapped into a $n \times n$ co-association matrix:

$$C(i,j) = \frac{n_{ij}}{N} \tag{8}$$

where

- C is the co-association matrix.
- N is the number of partitions created by the N clusterings.
- n_{ij} is the number of times that the packet pair (i,j) is assigned to the same cluster among the N partitions.

In order to create this co-association matrix, each clustering is represented by a $n \times n$ matrix (n is the total number of samples) where the (i,j) position is either 1 if observations i and j belong to the same cluster and 0 otherwise. The average of all these matrices constitutes the co-association matrix.

DBSCAN. It is a density-based clustering algorithm as its name suggests. It was designed by Ester et al. (1996) to group connected regions of high density into the same cluster, in a clustering problem with noise where clusters have arbitrary shapes. The outcome of the clustering depends on two hyperparameters: the maximum distance allowed between two points within the same cluster referred to as ε and the *MinPts*, which defines the minimum number of data points required to form a distinct and dense cluster.

Let a region of the space with a radius of ε centered at a point p of the dataset D be a dense region if it contains at least *MinPts* points. For each point p, an ε -neighborhood N_ϵ can be defined as follows:

$$N_\epsilon(p) = \{q \in D | dist(p,q) \leq \epsilon\} \tag{9}$$

The ε -neighborhood equals the set of points inside a hypersphere of radius ε centered at point p. A point is called core-point if its ε-neighborhood N_ϵ contains at least *MinPts* points.

$$|N_\epsilon(p)| \geq MinPts \tag{10}$$

DBSCAN looks for clusters by checking the neighborhood of each object in the dataset. If the neighborhood of an object p contains more points than *MinPts*, a new cluster with p as the core-point is created. It then iteratively collects the objects directly density-reachable from these core points, which may involve merging multiple clusters. The process ends when no new objects can be added to any cluster (Khan et al., 2014).

3.2. Evaluation metrics

To assess ADBC performance, the applied evaluation metrics are briefly explained in the following subsections.

(1) **Homogeneity**: A clustering result satisfies homogeneity if all of its clusters contain only data points that are members of a single class. On the other word, for IDS application homogeneity shows how well all packets belong to an attack type are assigned to the same cluster. In an ideal situation, the class distribution within each cluster should be skewed to a single class, meaning zero entropy. To determine how close a given clustering is to this ideal, the conditional entropy of the class distribution is examined. The score is determined between 0 and 1, with high values indicating a good homogeneity outcome. Homogeneity is calculated using the following formulas (Rosenberg and Hirschberg, 2007; Ozdemir et al., 2015).

$$h = \begin{cases} 1 & \text{if H(C,K)} = 0 \\ 1 - \frac{H(C|K)}{H(C,K)} & \text{else} \end{cases} \tag{11}$$

where,

$$H(C|K) = -\sum_{k=1}^{|K|}\sum_{c=1}^{|C|}\frac{A_{ck}}{N}\log\frac{A_{ck}}{\sum_{c=1}^{|C|}A_{ck}} \tag{12}$$

$$H(C,K) = -\sum_{k=1}^{|K|}\sum_{c=1}^{|C|}\frac{A_{ck}}{N}\log\frac{A_{ck}}{N} \tag{13}$$

where,

- N is the number of data points
- $C=\{c_i|i=1,\ldots,n\}$ is a set of classes
- $K=\{k_i|i=1,\ldots,m\}$ is a set of clusters
- A represents the contingency table produced by the clustering algorithm, such that $A=a_{ij}$ with a_{ij} the number of data points that are members of label c_i and elements of cluster k_j.
- $H(C|K)$ the conditional entropy of the class distribution

The $H(C|K)$ is maximal and equals $H(C)$ when the clustering provides no new information (the class distribution within each cluster is equal to the overall class distribution). And $H(C|K)$ is 0 when each cluster contains only members of a single class that means a perfectly homogeneous clustering. Therefore, when all samples in a cluster k have the same label c, the homogeneity equals 1.

(2) **Silhouette Score**: This metric shows how well clusters are apart from each other and clearly distinguished. It ranges between −1 to 1. Values near 0 indicate overlapping clusters. Negative values generally indicate that a sample has been assigned to the wrong cluster, as a different cluster is more similar (Sklearn metrics, 2021; Das, 2021). A silhouette score of a data point i is defined as follows:

$$Silhouette\ score(i) = \frac{b_i - a_i}{max(b_i, a_i)} \tag{14}$$

where,

- b_i represents the smallest mean distance of i to all points in any other cluster
- a_i is the average dissimilarity of i to all other data points of the same cluster.

If $b_i > a_i$, then a point is well separated from its neighboring cluster whereas it is closer to all points from the cluster it belongs to.

4. Implementation phases

The implementation of ADBC involves cleaning and pre-processing datasets, applying sampling techniques and feature extraction methods, and utilizing clustering algorithms to categorize unknown traffic.

4.1. Datasets

In order to evaluate ADBC model performance, three publicly available datasets are used. These datasets include a variety of network traffic attacks and meet the real traffic criteria to some extent. The mentioned datasets were selected after a survey among 14 publicly available datasets (Monshizadeh et al., 2021).

The MAWILab-2018 dataset is captured on a link between USA and Japan, every day and since 2001 For the current paper, the traffic from 28th August 2018 is used (Fontugne et al., 2010). Furthermore, in order to check model resilience and robustness and to have a diversity of attacks, all the DoS attacks contained in ISCX-2017 dataset (Sharafaldin et al., 2018) were extracted and injected into MAWILab-2018 (DoS attack class). In this dataset, there is an unknown class from the beginning which is labeled as such.

The ISCX-2017 dataset consists of 51G network traffic metadata that is labeled including 80 features and the full packet payload. The network traffic is provided on protocols, such as HTTP, HTTPS, FTP, SSH, and email. The dataset includes the most common attacks based on the 2016 McAfee report, such as Web-based, Brute force, Denial of Service (DoS), Distributed Denial of Service (DDoS), Infiltration, Heart-bleed, Bot and Scan covered in this dataset (Monshizadeh et al., 2021). In this dataset, some packets cannot be correlated to any of the labels provided by the dataset owners. Therefore, they are considered as unknown traffic.

The ISCX-2012 dataset was captured in 2012 over one week in an emulated environment. Dataset includes normal and malicious traffic (Sharafaldin et al., 2018; Ring et al., 2019; Shiravi et al., 2012). For current paper experiments, the attacks are grouped into three categories: Local to Remote (L2R), Remote to Local (R2L), and Local to Local (L2L). In this dataset, some packets are considered unknown as they cannot be correlated to any of the labels provided by the dataset owners.

While the network traffic payload may have different characteristics for every dataset, this study only analyzes the header of the network traffic datasets that consist of similar attributes and protocols. Therefore, mixing datasets has not been an issue as the data points were also close in the feature space during the experiments. The distribution of packets in the selected subsets of the datasets is shown in Table 1. The packets are considered in normal (−2), unknown (0) and attacks (1, ...,) classes. In the field of IDS, unknown traffic refers to any packet that is not labeled and not classified to any pre-defined category.

The packets in subset of datasets are randomly selected but the proportion of each class is preserved as original dataset.

4.2. Data preprocessing

Data cleaning, converting the columns to the right types, handling missing values, splitting IP addresses into four fields, vectorizing categorical variables, normalizing the dataset, and changing the labels of attack classes in order to differentiate different attack classes are the processes carried out in this phase. For the normalization, statistical and scaling normalization are used (Wang et al., 2009).

Table 1
Distribution of packets in the selected subsets of the datasets.

Dataset	Class type	Class No.	Packets	Total
MAWILab-2018	Normal	−2	9173	43 631
	Unknown	0	6135	
	Attack Class 1 (DoS)	2	4329	
	Attack Class 2 (Multi. points mptmp)	3	4625	
	Attack Class 3 (Multi. points mptp)	4	3636	
	Attack Class 4 (Multi. points ptmp)	5	7216	
	Attack Class 5 (HTTP attack)	6	1435	
	Attack Class 6 (Network scan TCP)	7	2704	
	Attack Class 7 (Network scan UDP)	8	4368	
	Attack Class 8 (TTL error)	9	10	
ISCX-2017	Normal	−2	9512	54 430
	Unknown	0	8138	
	Attack Class 1 (DoS)	2	7727	
	Attack Class 2 (Bot)	3	1248	
	Attack Class 3 (PortScan)	4	9653	
	Attack Class 4 (Infiltration)	5	5884	
	Attack Class 5 (FTP-Patator)	6	3855	
	Attack Class 6 (Heartbleed)	7	8413	
ISCX-2012	Normal	−2	10 811	43 731
	Unknown	0	13 604	
	Attack Class 1 (L2R)	2	8201	
	Attack Class 2 (R2L)	3	6640	
	Attack Class 3 (L2L)	4	4475	

Table 2
Features in datasets.

No.	Features	No.	Features
1	Ethernet size	31	UDP length
2	Ethernet destination (divided into 6 features)	32	ICMP type
8	Ethernet source (divided into 6 features)	33	ICMP code
14	IP header length	34	Duration of connection
15	IP type of service	35	Connection starting time
16	IP length	36	IP fragmentation flag
17	IP time to live	37	IP fragmentation overlap
18	IP protocol	38	TCP ACK flag
19	IP source (divided into 4 features)	39	TCP retransmission
23	IP destination (divided into 4 features)	40	TCP push flag
27	TCP source port	41	TCP SYN flag
28	TCP destination port	42	TCP FIN flag
29	UDP source port	43	TCP urgent flag
30	UDP destination port		

In order to improve the performance of the algorithms, nominal attributes are transformed into numeric attributes. In addition, the IP address and hexadecimal Medium Access Control (MAC) address of the applied datasets are transformed into separate numeric attributes. Each numeric attribute is normalized using batch mean and standard deviation unless there is an already defined range (e.g., IP address range) (Monshizadeh et al., 2021). Datasets contain 43 features as shown in Table 2.

5. Experimental results

All the experiments are carried out on a server with Intel® Xeon® 16 x E5-2623 CPU @3.0 GHz (4 cores in each processor), 128 GB RAM and 1.6 TB HDD. The scripts were developed in Python in a Linux environment (Ubuntu 20.04.1 LTS) and utilized Scikit-learn and Tensorflow2 library (Scikit-learn, 2021).

In order to prove the proposed architecture superiority to prior art, this section evaluates and compares performance of ADBC against prior art algorithms. For this purpose, the following prior art methods are implemented by authors in the same testing environment and by employing similar datasets that are applied to ADBC.

- DBSCAN and SVMonline (with known and unknown traffic)
- DBSCAN and lambda VAE (with known and unknown traffic)
- DBSCAN and vanilla VAE (with known and unknown traffic)
- ADBC (with known and unknown traffic)

The known traffic is used in all experiments to verify the model is able to cluster traffic properly. The known traffic is classified in multiple classes as shown in Table 1, whereas the clustering process categorizes unknown traffic based on similarity among data points. Furthermore, this architecture not only clusters unknown traffic but provides insights to the cybersecurity expert as well. In other words, the composition of the cluster and the proximity between the unknown and known data samples can help the expert decide how much time and investigation is needed for each new attack or sample. In brief, the goal is to direct the expert's attention to new attacks and facilitate the analysis of attacks that are already known through insights.

5.1. DBSCAN and SVM_o

In this experiment SVM_o feature selection is used with DBSCAN in order to cluster unknown traffic. The known traffic is included in the experiments, but the labels are only used during the evaluation as the applied method is unsupervised. The best selected features for each dataset and ordered by importance are as follows:

- MAWILab-2018: IP source 4, IP fragment offset, IP source 3, TCP source port, UDP destination port, and IP source 2.
- ISCX-2017: Connection starting time, IP source 3, IP destination 2, Ethernet source 2, IP destination 4, and IP destination 3.
- ISCX-2012: Connection starting time, Ethernet destination 3, Ethernet source 2, Ethernet destination 1, TCP destination port and IP time to live.

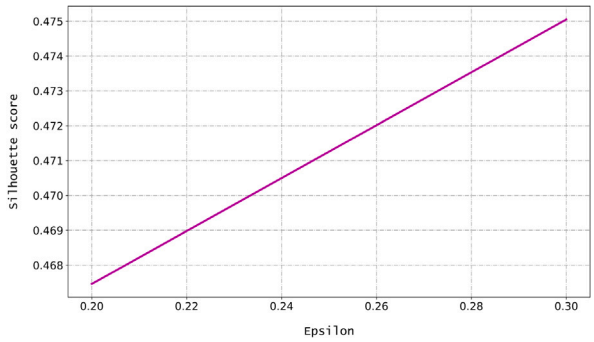

Fig. 4. DBSCAN_SVMo for MAWILab-2018.

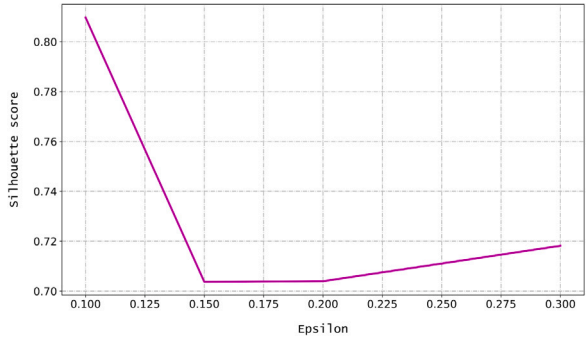

Fig. 5. DBSCAN_SVMo for ISCX-2017.

Figs. 4–6 shows the performance evaluation of DBSCAN-SVM$_o$ for each dataset. Though, this method is examined for two values of *MinPts* hyperparameter where the minimum number of packets assigned in each cluster are not less than 5 and 10, only the results for *MinPts=10* is presented in this section.

From Figs. 4–6, it is obvious that maximum silhouette score achieved by this method depends on the dataset (0.475 for MAWILab-2018, 0.81 for ISCX-2017 and 0.84 for ISCX-2012 dataset). Furthermore, the epsilon hyperparameter varies for each dataset in order to achieve the highest silhouette score (epsilon = 0.3 for MAWILab-2018, 0.1 for ISCX-2017 and 0.20 for ISCX-2012). Hence DBSCAN needs to be tuned based on each dataset. These results reject the method robustness.

Furthermore, Figs. 7–12, represent the data points using the six best features selected by SVM$_o$. At each time, the data points are projected into a 3D feature space for visualization purposes. As it is shown in these figures, even some clusters are more visible, still due to the high degree of overlapping (due to low silhouette score), distinguishing each cluster is a challenging task. This indicates the data space created by the selected features in this method are not suitable for clustering. In other words, these figures show that the desired result cannot be obtained directly using a clustering algorithm because this is not a natural clustering problem. Therefore, the data space must be changed in order to transform this problem into a clustering one. Notice that this observation is valid for the three datasets.

5.2. DBSCAN and vanilla VAE

This experiment applies vanilla VAE feature extraction with DBSCAN to cluster unknown traffic. Notice that the known traffic is used without its labels. The unknown traffic (label 0) should not be considered as a class because it cannot be a single cluster. In fact, it is composed of many classes. However, it is represented in the figures using the same color to distinguish it from the rest of the traffic. The

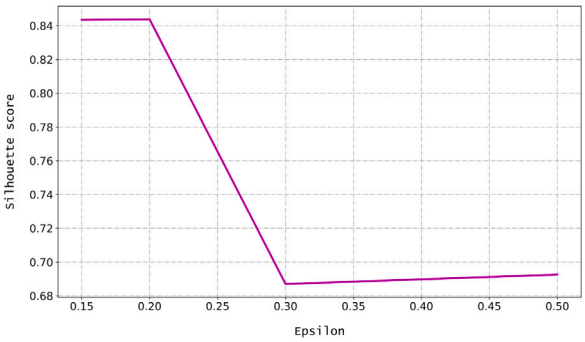

Fig. 6. DBSCAN_SVMo for ISCX-2012.

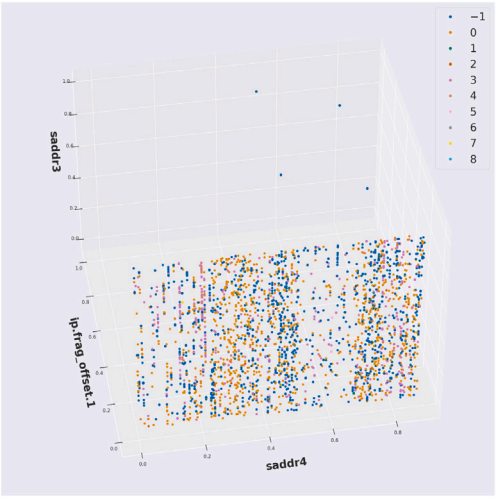

Fig. 7. DBSCAN_SVMo with first feature set for MAWILab-2018.

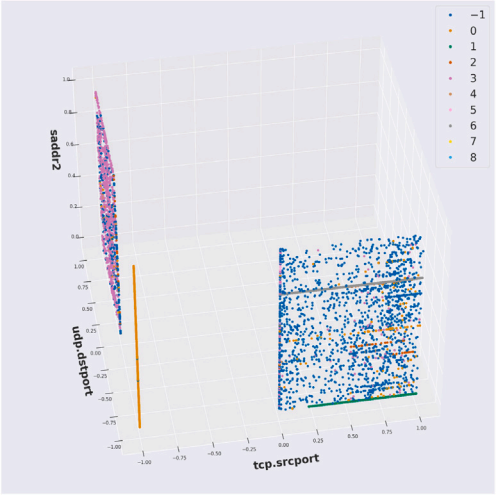

Fig. 8. DBSCAN_SVMo with second feature set for MAWILab-2018.

data is projected at each time into a 3D space of 3 extracted features (The VAE extracted 6 encoded features overall for each dataset: *z0, z1,z2,z3, z4* and *z5*) for visualization purposes.

Figs. 13–15 show the performance evaluation of DBSCAN-VanillaVAE for each dataset. Though, this method is examined for two values of *MinPts* hyperparameter where the minimum number of

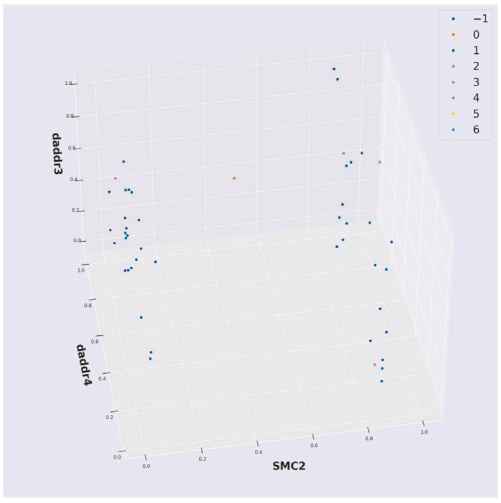

Fig. 9. DBSCAN_SVMo with first feature set for ISCX-2017.

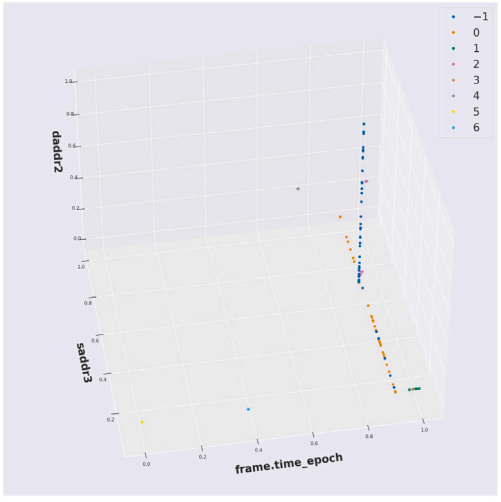

Fig. 10. DBSCAN_SVMo with second feature set for ISCX-2017.

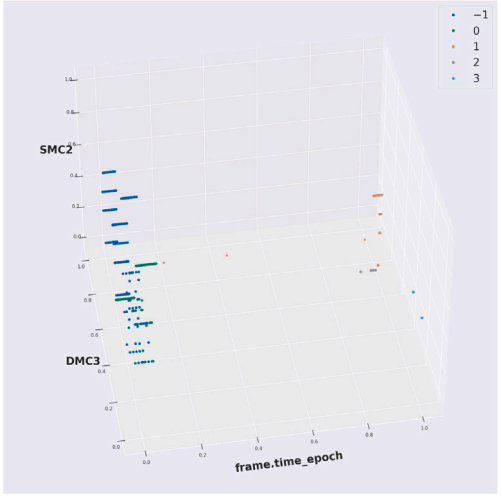

Fig. 11. DBSCAN_SVMo with first feature set for ISCX-2012.

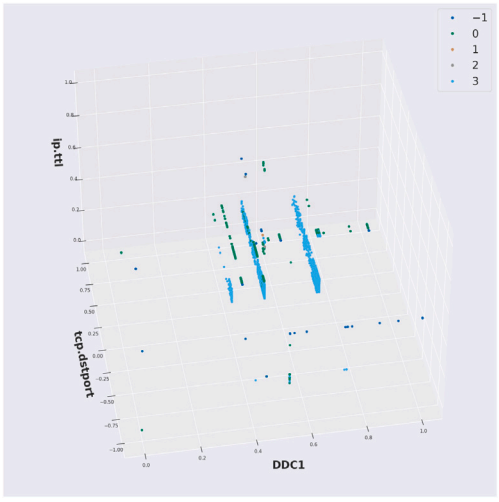

Fig. 12. DBSCAN_SVMo with second feature set for ISCX-2012.

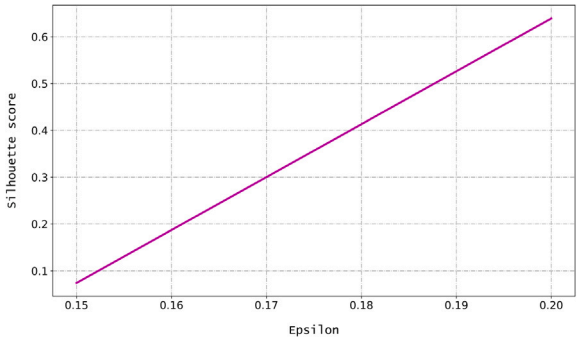

Fig. 13. DBSCAN_Vanilla VAE for MAWILab-2018.

packets assigned in each cluster are not less than 5 and 10, only the results for *MinPts=10* is presented in this section.

From Figs. 13–15, it is obvious that maximum silhouette score achieved by this method depends on the dataset (0.67 for MAWILab-2018, 0.43 for ISCX-2017 and 0.47 for ISCX-2012 dataset). Furthermore, the epsilon hyperparameter varies for each dataset in order to achieve the highest silhouette score (epsilon = 0.2 for MAWILab-2018, 0.3 for ISCX-2017 and 0.20 for ISCX-2012). Hence DBSCAN needs to be tuned based on each dataset. Similar to previous method, the silhouette score and epsilon value depends on the dataset meaning the model is not robust.

Furthermore, as illustrated in Figs. 16–21, though the Vanilla VAE makes the classes look more like clusters, yet the clusters are still close to each other and overlap in this new feature space which will make clustering a hard task.

5.3. DBSCAN and Lambda VAE

This experiment applies Lambda VAE feature extraction with DBSCAN to cluster unknown traffic. Notice that known traffic is used in these experiments without its labels. The Lambda VAE was used here in order to encourage the clustering of the data in the latent space because the original VAE does not enforce any clustering in the latent space. In fact, the VAE loss function encourages the latent embeddings to be around the origin which is not very useful since it creates overlapping clusters.

Figs. 22–24 show the performance evaluation of DBSCAN-LambdaVAE for each dataset. Though, this method is examined for two values of *MinPts* hyperparameter where the minimum number of

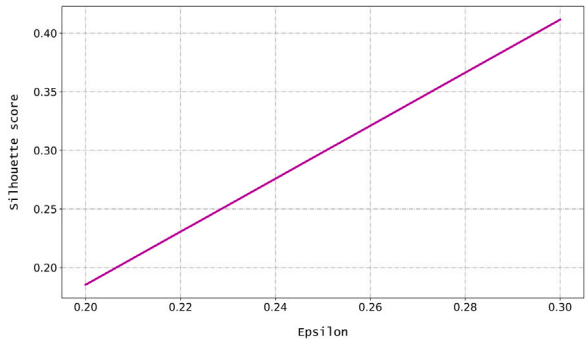

Fig. 14. DBSCAN_Vanilla VAE for ISCX-2017.

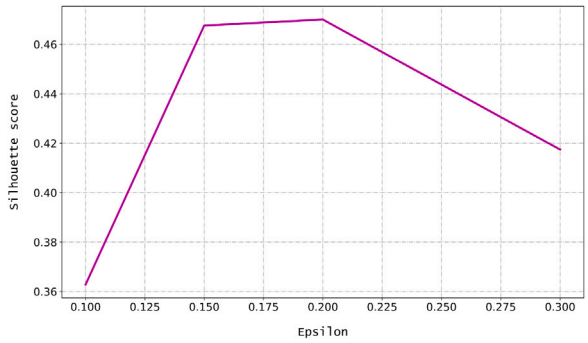

Fig. 15. DBSCAN_Vanilla VAE for ISCX-2012.

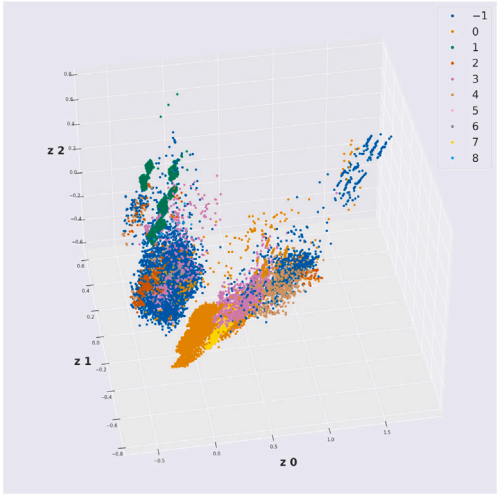

Fig. 16. DBSCAN_Vanilla VAE with first feature set for MAWILab-2018.

packets assigned in each cluster are not less than 5 and 10, only the results for *MinPts = 10* is presented in this section.

From Figs. 22–24, it is obvious that maximum silhouette score achieved by this method depends on the dataset (0.38 for MAWILab-2018, 0.47 for ISCX-2017 and 0.49 for ISCX-2012 dataset). Furthermore, the epsilon hyperparameter varies for each dataset in order to achieve the highest silhouette score (epsilon = 0.2 for MAWILab-2018, 0.3 for ISCX-2017 and 0.3 for ISCX-2012). Hence DBSCAN needs to be tuned based on each dataset. Similar to previous method, the silhouette score and epsilon value depends on the dataset meaning the model is not robust.

In this experiment, the Lambda VAE is supposed to encourage pushing each class far from others in the latent space in order solve

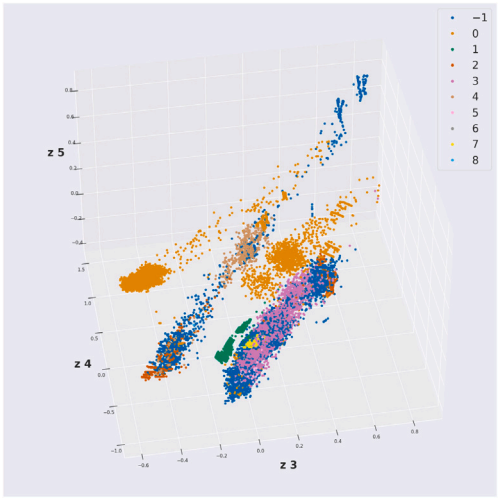

Fig. 17. DBSCAN_Vanilla VAE with second feature set for MAWILab-2018.

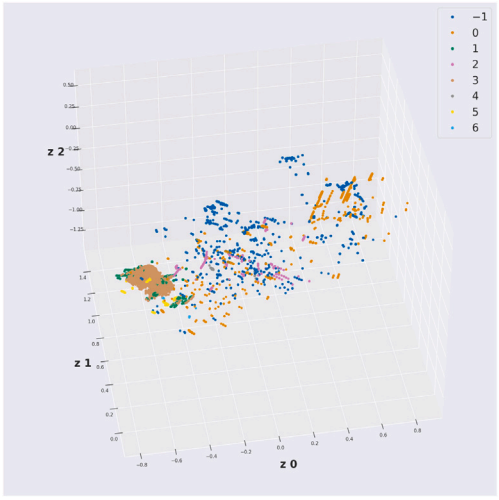

Fig. 18. DBSCAN_Vanilla VAE with first feature set for ISCX-2017.

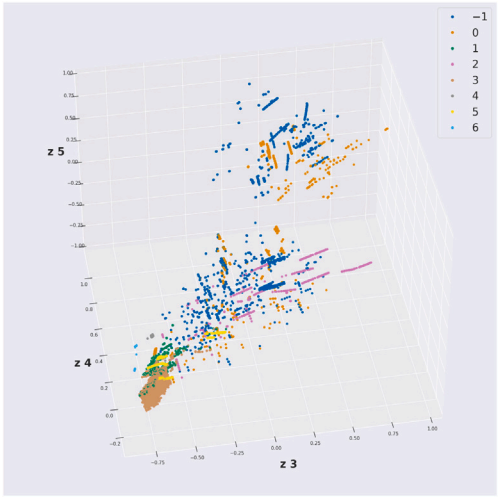

Fig. 19. DBSCAN_Vanilla VAE with second feature set for ISCX-2017.

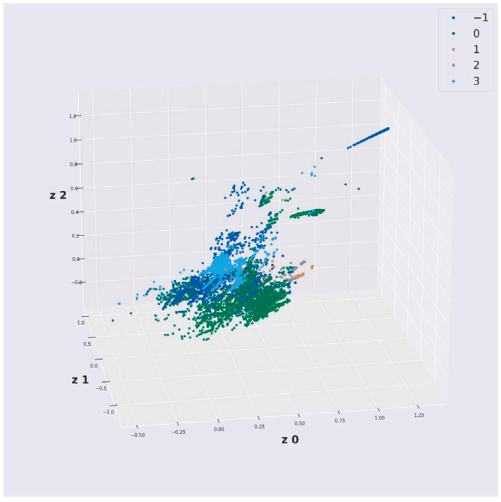

Fig. 20. DBSCAN_Vanilla VAE with first feature set for ISCX-2012.

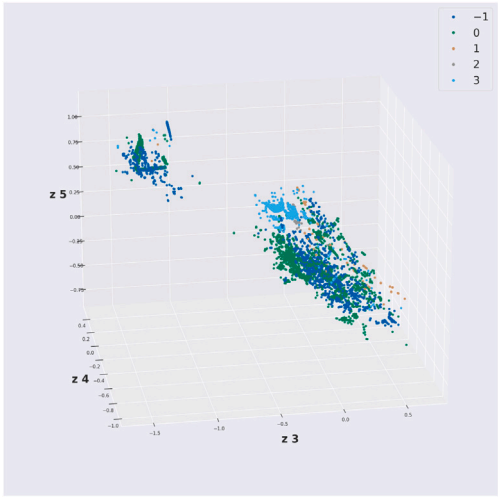

Fig. 21. DBSCAN_Vanilla VAE with second feature set for ISCX-2012.

the class overlapping problem. While the lambda VAE slightly reduces the overlapping, it does not solve the problem completely as shown in Figs. 25–30 for all datasets. Therefore, the clustering cannot be performed directly on feature space created by the extracted features using lambda VAE.

5.4. ADBC

This method aims to change the data space by creating a more robust distance metric that replaces the Euclidean distance. Therefore, the distance between the data points in the new space will be computed according to a clustering-based distance instead of the usual Euclidean distance.

In order to create this distance, the first step is to apply different clustering algorithms to the input data. For instance, this paper applies the same clustering algorithm k-means, with different numbers of clusters. In other words, it apples k-means with several values for hyperparameters to create different clusterings.

In the second step, the previous clusterings are used in order to create the Co-Association Matrix as explained in Section 3. The Co-Association Matrix allows to directly deduce the distance matrix which will be used as an input for DBSCAN.

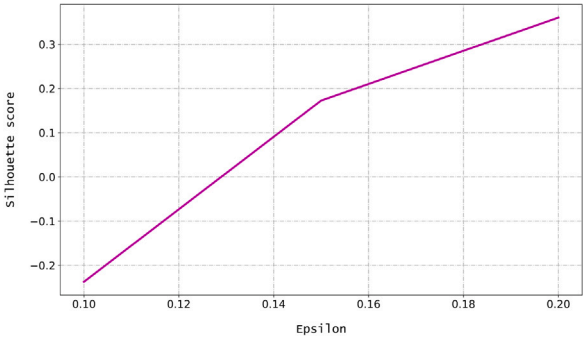

Fig. 22. DBSCAN_Lambda VAE for MAWILab-2018.

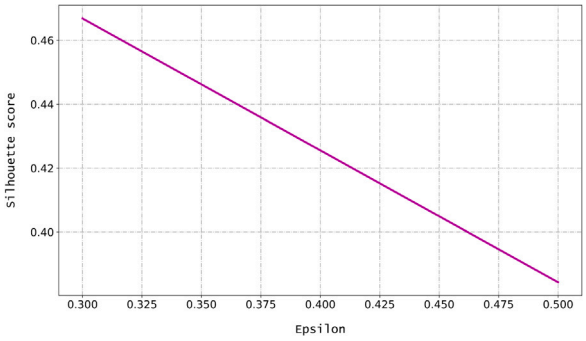

Fig. 23. DBSCAN_Lambda VAE for ISCX-2017.

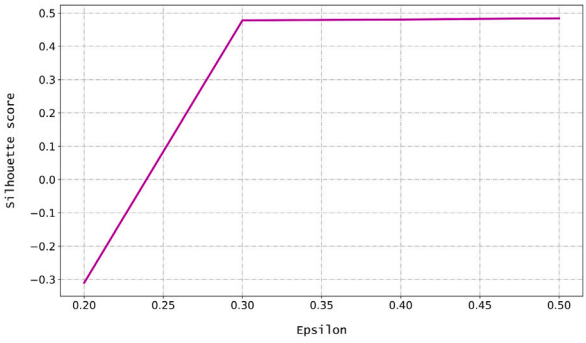

Fig. 24. DBSCAN_Lambda VAE for ISCX-2012.

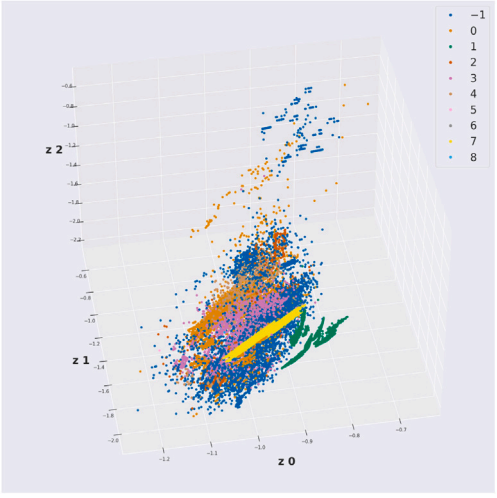

Fig. 25. DBSCAN_Lambda VAE with first feature set for MAWILab-2018.

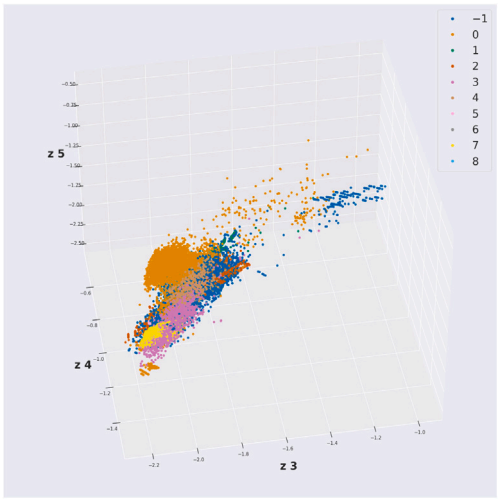

Fig. 26. DBSCAN_Lambda VAE with second feature set for MAWILab-2018.

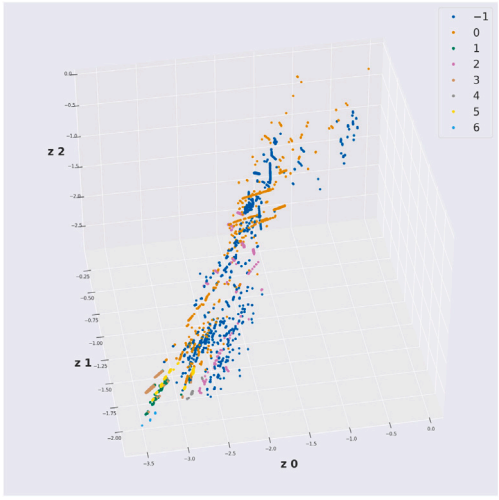

Fig. 27. DBSCAN_Lambda VAE with first feature set for ISCX-2017.

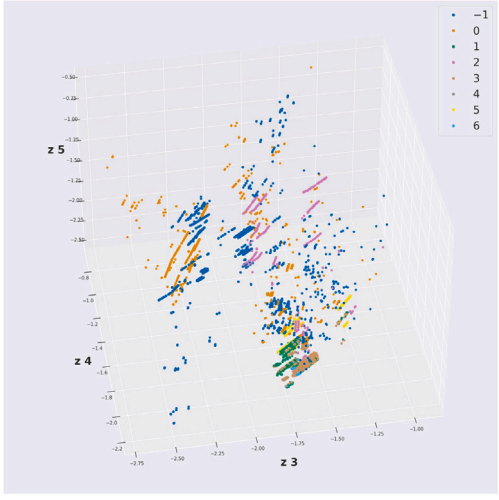

Fig. 28. DBSCAN_Lambda VAE with second feature set for ISCX-2017.

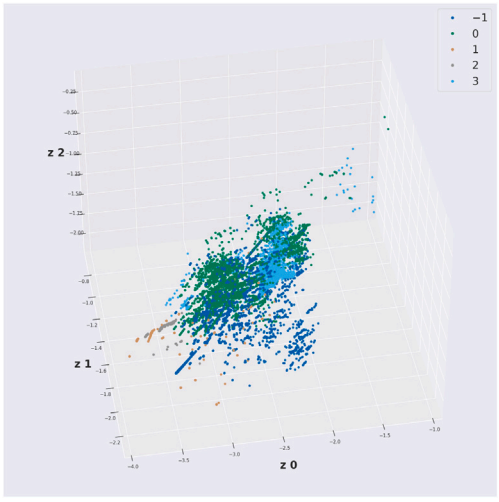

Fig. 29. DBSCAN_Lambda VAE with first feature set for ISCX-2012.

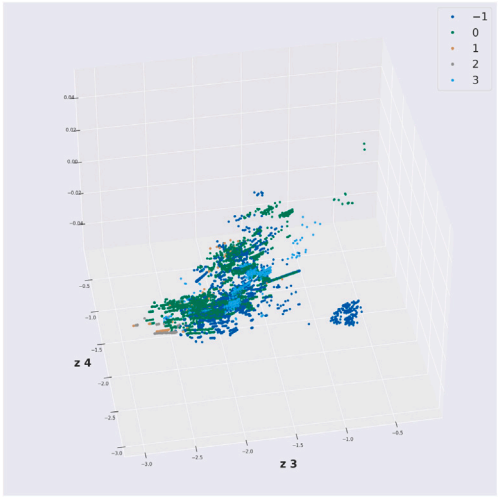

Fig. 30. DBSCAN_Lambda VAE with second feature set for ISCX-2012.

The following explains the steps of ADBC architecture. In general, the ADBC architecture has four main phases:

1. Creating a Co-Association Matrix with applying N different clustering methods to a subset of the input data in order to produce N clustering sets. This step will help us avoid the drawbacks of a single clustering technique applied to an intrusion detection dataset with clusters having very different densities. In fact, the application of a single clustering algorithm is never sufficient nor stable because the result varies a lot with minor changes in the hyperparameters or the input data. In this step, the different clustering algorithms or even the same algorithm with different hyperparameters or initializations are applied.
2. Applying DBSCAN on the distance matrix (deduced from the Co-Association Matrix) representing the new distance between data points in order to detect the attack clusters. For this purpose, a co-association matrix that is a combination of the obtained N clustering sets is created. A distance matrix can be easily derived from this co-association matrix. The distance represented in this matrix is different from the Euclidean distance that rarely reflects the similarity between packets in real scenarios. In fact, this distance is based on the following idea: the more often two packets fall into the same cluster when several clustering

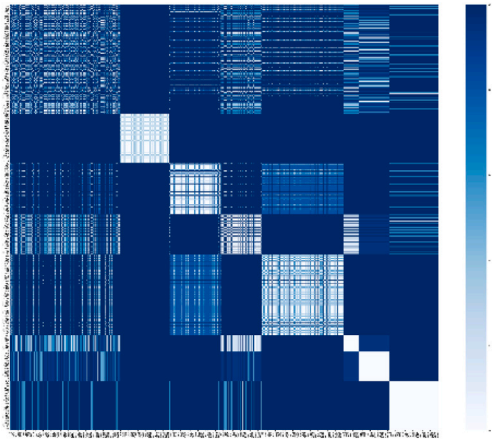

Fig. 31. Distance matrix of MAWILab-2018.

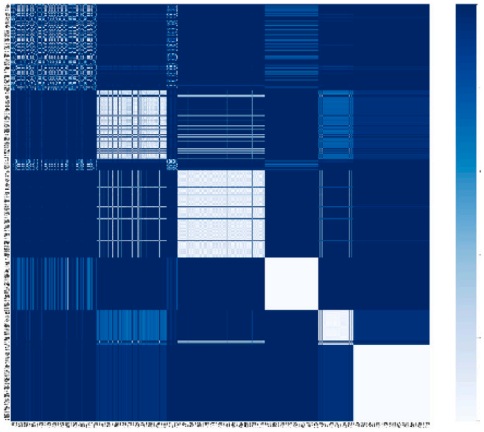

Fig. 32. Distance matrix of ISCX-2017.

algorithms are applied, the more similar they are and the more likely they are to belong to the same attack type. Therefore, the combination process that is used to obtain the co-association matrix is based on a voting mechanism.

3. As the resulting distance matrix represents the input data points in a new feature space where the distance between the points is a new measure of similarity, DBSCAN can be applied directly to this distance matrix in order to detect the attack classes.
4. The created clusters will be generalized to the whole dataset and analyzed by a security investigator (human or an algorithm) in order to identify malicious clusters.

Figs. 31–33 show the distance matrices which are issued from the co-association matrices for the three datasets. These distance matrices are grouped by clusters and reveal that the proposed architecture can detect the known attack classes without using the labels. The similar distance matrices can be obtained with variations in the DBSCAN parameters, which shows the robustness of the final clustering. This is due to the voting mechanism that was used to create the distances it means the more often two data points appear in the same clusters, the smaller the distance between them.

In these figures, data points are grouped into attack classes. In the distance matrices, these classes are somehow visible. In fact, they can be detected using white areas (where the distance between data points is low).

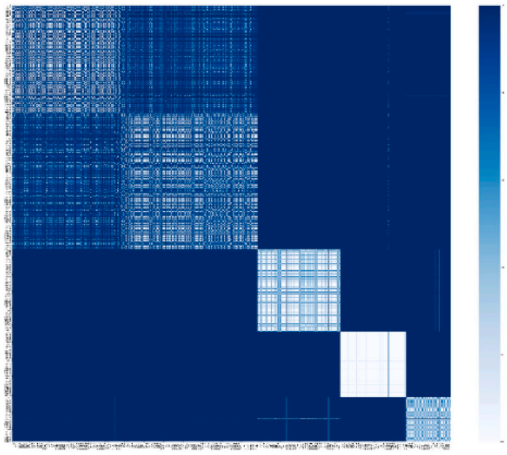

Fig. 33. Distance matrix of ISCX-2012.

5.5. Performance evaluation

In order to evaluate ADBC performance, a comprehensive examination has been done based on the architecture robustness and for different implementation factors such as hyperparameter setting and diverse datasets. Figs. 34–39 depict the performance of the obtained clustering architecture for each dataset. Though most studies in this field consider only silhouette score for performance evaluation, this paper presents the homogeneity score in addition to silhouette score to evaluate the ADBC performance. If the homogeneity score is high, each cluster will be mainly composed of the same attack class. However, the overall number of clusters will be high which is, in general, a drawback for the security investigator as this means that more investigation time will be needed. If the homogeneity score is less important, there will be fewer clusters to investigate however there is no guarantee that they are composed mainly of the same attack. Therefore, in order to select proper evaluation metrics, finding a good compromise between the homogeneity score and the silhouette score is important.

This paper combines both supervised and unsupervised scores to have a more complete idea about the performance of the clustering and the coherence of the insights that will be given based on the known traffic. Consequently, this paper focuses on the silhouette score because intuitively, a high silhouette score indicates that the clusters are compact (each point is similar to the other points in its cluster) and well separated (each point is unlike the points in other clusters) which gives an idea about the clusters for both unknown and known traffic. Moreover, this study considers the homogeneity score which ensures that the clusters contain mainly the same data points at each time (this score does not include the unknown traffic as it is a supervised score).

As it is illustrated in Figs. 34–39 for the *MinPts=10* for the same hyperparameter epsilon(eps = 0.05), the architecture provides the highest silhouette score 0.99 for MAWILab-2018, 0.99 for ISCX-2017 and 0.98 for ISCX-2012. Similarly, for *MinPts=5* for epsilon = 0.05 the highest silhouette score is achieved for three datasets. In addition, the highest homogeneity score is achieved for all datasets and with same hyperparameter settings. These results conclude that the architecture does not need to be tuned often. Therefore, it can be claimed that one of the main achievements of this architecture is the robustness against different datasets.

5.6. Cluster analysis

This paper provides insights based on the created clusters to a cybersecurity investigator to help in detecting unknown attacks. The cluster composition analysis is based on the type of packets in each

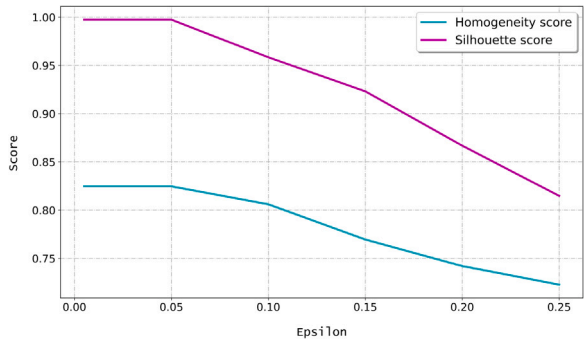

Fig. 34. Performance scores for MAWILab-2018 for MinPts = 5.

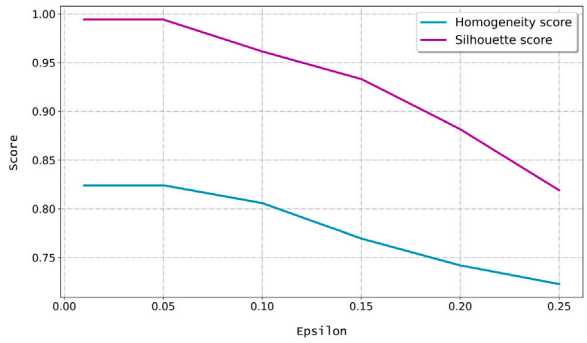

Fig. 35. Performance scores for MAWILab-2018 for MinPts = 10.

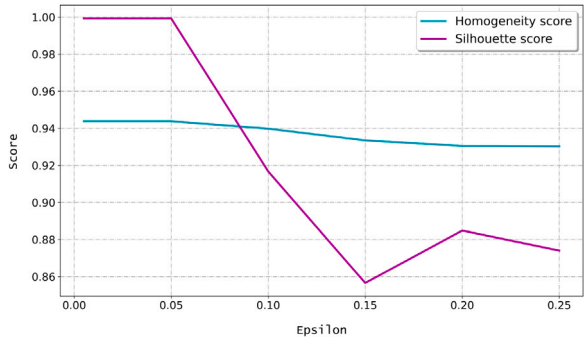

Fig. 36. Performance scores for ISCX-2017 for MinPts = 5.

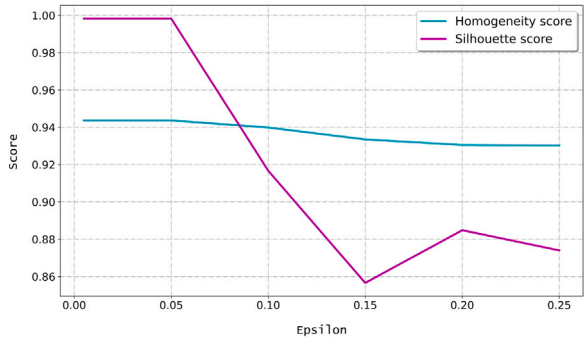

Fig. 37. Performance scores for ISCX-2017 for MinPts = 10.

cluster. Therefore, instead of analyzing thousands of packets, only few samples will be investigated, and result would be generalized to the entire cluster. This approach helps the security investigator with reducing the time and computation resources.

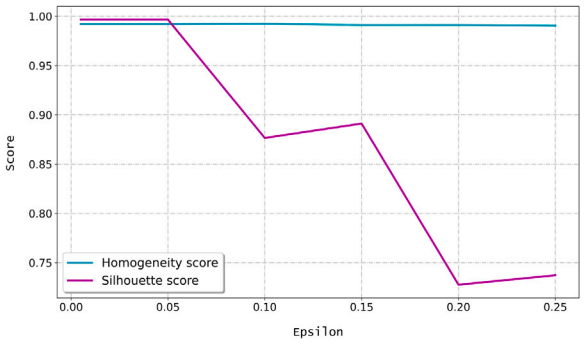

Fig. 38. Performance scores for ISCX-2012 for MinPts = 5.

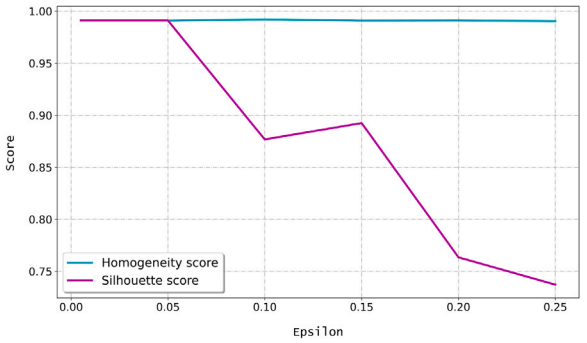

Fig. 39. Performance scores for ISCX-2012 for MinPts = 10.

- Class -2: Current cluster contains both outliers and normal traffic. To solve this issue, these two categories can be separated (as it is possible in DBSCAN to consider a separate class −1 for noise). Furthermore, some techniques such as user profiling can be applied to separate different type of normal traffic that are distributed among the clusters.
- Class 0: Though the output of the model still has an unknown cluster, the packets that are categorized as unknown are much less than original ones. In addition, the approach gives possibility to security investigator to analyze one unknown packet and generalize the analysis to other packets in the cluster.
- Class 3, 4 and 5: These clusters represent multipoint attack categories and they are labeled based on the services rather than attack types. Therefore, there is overlapping between some of the clusters for the mentioned attacks.

An example of cluster composition for a subset of MAWILab-2018 and hyperparameters: *eps* = 0.1 and *MinPts* = 5 is depicted in Table 3 after applying ADBC on the mentioned dataset.

This table depicts the number of packets in each cluster distributed according to various attack categories. This clustering achieved a silhouette score of 0.958 with 26 clusters and produces 32 noise packets.

For this example, a hyperparameter tuning is used in order to balance the total number of clusters, the size of each cluster and the performance metrics. The obtained metrics for this example are:

- Homogeneity score 0.8061
- Silhouette score 0.9584

The unknown traffic contained in these clusters will be manually analyzed by the security investigator to identify normal and attack behaviors. However, the provided clusters will point the investigator's attention to new attacks, hence facilitate the unknown traffic analysis. Therefore, it is important to provide the investigator with complete information about the composition of the clusters in order to speed up the investigation.

Table 3
Cluster analysis for MAWILab-2018.

		Class No.									
		−2	0	2	3	4	5	6	7	8	9
Cluster	1	38	20		4	8					
	2	33			2897					1	
	3	639	6		9	882					
	4		3566								
	5	2212	301								
	6	486	8			18			2704		
	7	590	11			299		1432			
	8	11				128					4368
	9	943	28		7	2168		3			
	10	198	2		75		4888			4	
	11	1044	18		811		1925			2	
	12			1071							
	13	533	11		175	4					
	14			3258							
	15		1097								
	16	1641	104		45	49					
	17		432								
	18	159	21				400				
	19	97	1		564						1
	20		482								
	21	80	22		37		3			2	
	22	345	5								
	23	178									
	24	1				80					
	25	23									
	26	22			1						

For instance, if the threshold is considered 50 packets in the Table 3 then for cluster 16, there are 1641 packets classified as safe and 104 packets classified as unknown. The classes 3 (45 packets) and 4 (49 packets) will be discarded since packets in these classes are under the threshold. Hence, the investigator only analyzes two classes.

In the analysis process, clusters that contain less than the packet threshold, may be discarded. For the rest of the clusters and for decreasing computation resources, the packet numbers in each cluster will be converted to percentage of total number of packets. Furthermore, the malicious clusters and consequently, the attack packets will be identified based on some characteristics (e.g., the time stamp in relation to location of generated packets and so on) by security investigator.

The detected attack packets will be fed into the architecture for training purposes. Hence, the architecture will be trained periodically with new packets after a time threshold (e.g., monthly). The threshold time for training process will be defined based on the computation requirements.

Thus, it must be determined whether the algorithms have previously undergone training by newly detected attack packets. If no training has been previously done, then new training will be needed for clustering algorithms. Likewise, if the algorithms have already been trained, but the training took place outside a predefined time window or after a predefined amount of data, then the algorithms undergo retraining to ensure it can handle data properly. The time window and the amount of data may be selected by a user based on the application.

6. Discussion and future direction

As it was described in Section 2, several studies have applied unsupervised techniques to analyze datasets and cluster the datapoints based on their similarities. As it is shown in Table 4, these studies apply optimized version of DBSCAN or a combination of DBSCAN with various clustering techniques in order to categorize different type of datasets. Though, some of these methods provide good silhouette score, yet the achieved performance considerably depends on the datasets. In addition, hyperparameter settings are changing based on the different datasets. It must be noted that some of the prior art methods are applied for different application than IDS, their employed datasets are dissimilar to network traffic dataset, their testing conditions are distinct and even the presented results lack a thorough analysis of clusters quality. Therefore, in order to have a valid comparison between ADBC and other methods, similar dataset and testing environment are required, therefore authors implemented some of the prior art methods such as DBSCAN, DBSCAN-SVM$_o$, DBSCAN-VAEs in similar testing environment and on the same datasets. The achieved results proved the ADBC outperforms prior art methods in term of clustering performance and robustness.

In Fig. 40, the first plot shows the classes in the input space, where there are no natural clusters. Therefore, a clustering algorithm like DBSCAN cannot be applied directly to this input space. Hence, the DBSCAN is applied on different feature spaces as described in the previous sections. In the second and third experiments, DBSCAN was applied on a latent space of a VAE and a lambda VAE where the classes look more like clusters. However, these clusters have very distinct densities, and they are overlapping. Therefore, DBSCAN is applied on a co-association matrix in the last experiment in order to solve all the mentioned problems. This method helps to resolve the problem of varying densities of clusters and the overlapping problem.

Furthermore, a cluster composition analysis is presented based on the number of packets in each cluster and according to various attack categories. This analysis is intended to give insights to the cybersecurity investigator. Accordingly, it will be sufficient for security investigator to analyze a few packets belonging to each cluster and generalize the result to all packets belonging to the cluster rather than analyzing every packet. However, in order to determine the type of activity that each cluster corresponds, each cluster centroid or mean must be checked and inspected manually. On the other hand, the point to multipoint attacks provided in the applied dataset are labeled based on services rather than attack categories. Therefore, the clustering results provided in Table 3 demonstrate overlapping between some of the clusters for the mentioned attack categories. In order to avoid such overlapping (in the evaluation phase) another dataset can be applied with distinct distribution of attacks per protocol.

Moreover, the complexity and scale of learning algorithms are still questionable and out of scope of this paper and will be investigated by authors in the future work. In addition, the authors aim to automate the cluster analysis part and reduce the human interaction with ML algorithms.

Table 4
Prior art methods comparison.

Model	Dataset	Silhouette score	Arguments
• GBC (Muhlenbach and Lallich, 2009)	• auto-mpg • Breast Cancer Wisconsin • Ecoli • Iris • Ruspini • Test 2 Clusters • Test with random values • Yin and Yang	• 0.53 (ku[a] = 2), 0.41 (ku = 5) • 0.49 (ku = 2) • 0.72 (ku = 3), 0.30 (ku = 8) • 0.80 (ku = 2), 0.70 (ku = 3) • 0.91 (ku = 4) • 0.82 (ku = 4) • −0.59 (ku = 4) • 0.14 (ku = 2)	• Datasets consist of randomly selected points and do not have similar pattern as network traffic dataset. • This method is limited to hard clustering and fails if there is recovery between the different clusters. • The silhouette score greatly depends on the dataset where the achieved values are between −0.593 and 0.910. • Hyperparameters setting are changing based on datasets.
• DBSCAN • Kmeans • Agglomerative (Turner and Joseph, 2017)	• Snort rule 2.9.9.0	• 0.81 • 0.84 • 0.84	• Datasets are snort rules for UDP, TCP and ICMP. • Experiments are on few samples and concentrate on snort rules. • Results are provided for one dataset without any analysis on hyperparameter settings.
• Optimized DBSCAN (Sun et al., 2020)	• set1-upc • set2-camus	• 0.77 (eps = 4.2) • 0.67 (eps = 4.8)	• Datasets are internet messages. • Datasets are small size (36MB and 518KB). • Hyperparameter epsilon changes for datasets to achieve highest silhouette score.
• IIDBG (Xue-yong et al., 2010)	• KDD Cup 1999 Data	• NA	• Model is applied for IDS application with network traffic dataset. • The study provides supervised metrics such as detection rate and false negative ratio for performance evaluation. • The study applied clustering technique and discussed about hyperparameter such as epsilon and MinPts setting. • The study did not provide any analysis on clusters and unsupervised metric such as silhouette score.
• DB-Kmeans (Dong et al., 2019)	• NSL-KDD	• NA	• Model is applied for IDS application and employed network traffic dataset. • The study provides supervised metrics such as detection rates and false negative ratio for performance evaluation. • The study applied clustering technique but did not provide analysis on clusters and unsupervised metric such as silhouette score.
• Combination of DBSCAN, K-Means++ and SMO algorithms (Shakya and Makwana, 2017)	• KDD	• NA	• Study is for IDS application with network traffic dataset. • The study provides supervised metric such as accuracy for performance evaluation. • DBSCAN is used for noise reduction and not for clustering.
• DBSCAN (Sabottke et al., 2019)	• Web server logs	• 0.99 (eps = 0.1)	• Robustness against different dataset is not evaluated. • The obtained clusters are composed of different types of attacks (mixed clusters) that makes analysis difficult. • The result of packet per cluster analysis cannot be generalized to the entire cluster.
• HDBSCAN (Dockhorn et al., 2015)	• Aggregation • Moon • Blobs-1000D • Spiral • R15 • D31 • Flame	• 0.89 (eps = 1.42) • 0.68 (eps = 0.22) • 1.00 (eps = 47.27) • 1.00 (eps = 3.66) • 0.91 (eps = 0.42) • 0.87 (eps = 1.02) • 0.60 (eps = 2.66)	• V-measures are provided for performance evaluation. • Model is proper for convex shaped clusters rather than density-based clusters. • Hyperparameter epsilon changes for different datasets to achieve high silhouette score.
• Evidence Accumulation Clustering (Fred and Jain, 2005)	• Random data in a five-dimensional hypercube • Half-rings • Three concentric rings • Cigar • Iris	• NA	• Method is applied for image analysis. • Metrics such as number of clusters in clustering ensemble is used for the performance evaluation.

[a]ku is the number of clusters used for the test.

7. Conclusion

In the field of machine learning, non-labeled data analysis is one of the well-known challenges. In real-life scenarios, considerable amount of incoming data does not belong to any known category; and for unknown traffic, dividing data into the classes without having information on the nature of the traffic is challenging. Hence, clustering methods are introduced to gain some insight into the structure of the data. However, clustering techniques also have some drawbacks such as overlapping as clusters can appear with different shapes, sizes, and data density; in addition, it is difficult to generalize an algorithm for different dataset with the same hyperparameters tuning. Therefore, to solve the mentioned challenges, a novel and combined unsupervised approach is proposed in this paper. The architecture utilizes several clustering algorithms in conjunction with co-association matrix and a DBSCAN in order to categorize unknown traffic into various clusters. The model Introduces a new metric that helps to achieve high silhouette score by changing the feature space. The new similarity metric is different from Euclidean distance, where the distance between data points reflects the similarity between packets.

The illustrated results in this paper prove that architecture provides very high silhouette score (almost 0.99) that means clusters (attack classes) are distinct and with minimum overlapping. Therefore, this method helped to resolve the problem of varying densities of clusters and the overlapping problem. Similar analysis has been done for other

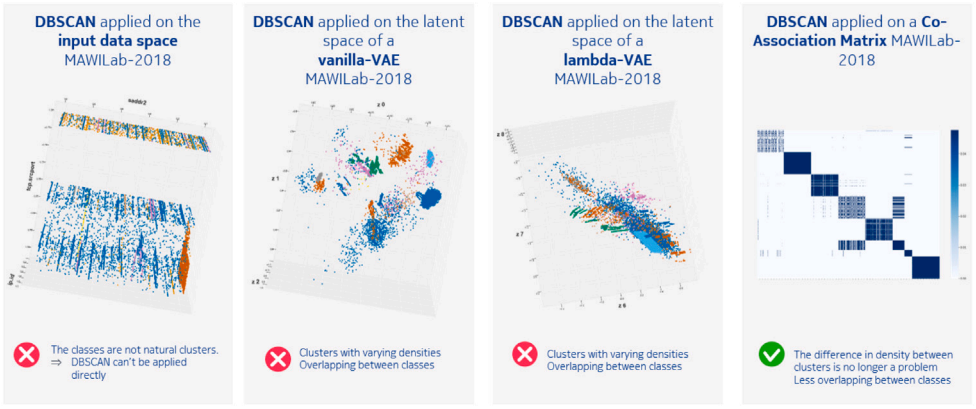

Fig. 40. Performance comparison among different techniques.

datasets with different size and divers attacks in order to verify architecture robustness and scalability. Therefore, it can be claimed that one of the main achievements of this architecture is the robustness against different datasets where with the same hyperparameter (epsilon), the architecture provides the highest silhouette score.

Furthermore, study provides cluster analysis generalization in a way that few packets from each generated cluster are analyzed, the malicious packets are identified, and the result of analysis is generalized to the entire cluster. This approach saves security investigator time and resources.

CRediT authorship contribution statement

Mehrnoosh Monshizadeh: Conceptualization, Methodology, Validation, Resources, Writing – review & editing, Writing – original draft, Software, Visualization. **Vikramajeet Khatri:** Data curation, Validation, Writing – review & editing, Software, Visualization. **Raimo Kantola:** Supervision, Writing – review & editing. **Zheng Yan:** Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Mehrnoosh Monshizadeh reports financial support and equipment, drugs, or supplies were provided by Nokia Bell Labs. Mehrnoosh Monshizadeh reports a relationship with Nokia Bell Labs that includes: employment and equity or stocks. Mehrnoosh Monshizadeh has patent SECURITY IN COMMUNICATION NETWORKS pending to Nokia Networks (FI 20225102).

Data availability

The authors do not have permission to share data.

References

Bedi, P., Gupta, N., Jindal, V., 2021. I-SiamIDS: an improved Siam-IDS for handling class imbalance in network-based intrusion detection systems. Appl. Intell. 51 (2), 1133–1151. http://dx.doi.org/10.1007/s10489-020-01886-y.

Chen, X.-W., 2003. Gene selection for cancer classification using bootstrapped genetic algorithms and support vector machines. In: Computational Systems Bioinformatics. CSB2003. Proceedings of the 2003 IEEE Bioinformatics Conference. CSB2003. pp. 504–505. http://dx.doi.org/10.1109/CSB.2003.1227389.

Das, A., 2021. Unsupervised learning techniques using python - K means and silhouette score for clustering. URL https://towardsdatascience.com/unsupervised-learning-techniques-using-python-k-means-and-silhouette-score-for-clustering-d6dd1f30b660 [cited 2021-09-17].

Dockhorn, A., Braune, C., Kruse, R., 2015. An alternating optimization approach based on hierarchical adaptations of DBSCAN. 2015 IEEE Symposium Series on Computational Intelligence 749–755. http://dx.doi.org/10.1109/SSCI.2015.113.

Dong, G., Jin, Y., Wang, S., Li, W., Tao, Z., Guo, S., 2019. DB-Kmeans:An intrusion detection algorithm based on DBSCAN and K-means. In: 2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS). pp. 1–4. http://dx.doi.org/10.23919/APNOMS.2019.8892910.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. pp. 226–231, URL https://dl.acm.org/doi/10.5555/3001460.3001507.

Fontugne, R., Borgnat, P., Abry, P., Fukuda, K., 2010. Mawilab: combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking. In: Proceedings of the 6th International Conference (Co-NEXT '10). URL https://doi.org/10.1145/1921168.1921179.

Fred, A.L.N., Jain, A.K., 2005. Combining multiple clusterings using evidence accumulation. IEEE Trans. Pattern Anal. Mach. Intell. 27, 835–850. http://dx.doi.org/10.1109/TPAMI.2005.113.

Jongsuebsuk, P., Wattanapongsakorn, N., Charnsripinyo, C., 2013. Network intrusion detection with fuzzy genetic algorithm for unknown attacks. In: The International Conference on Information Networking 2013 (ICOIN). pp. 1–5. http://dx.doi.org/10.1109/ICOIN.2013.6496342.

Khan, K., ur Rehman, S., Aziz, K., Fong, S.J., Sarasvady, S., Vishwa, A., 2014. DBSCAN: Past, present and future. In: The Fifth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT 2014). pp. 232–238. http://dx.doi.org/10.1109/ICADIWT.2014.6814687.

Kim, M., Na, H., Chae, K., Bang, H., Na, J., 2004. A combined data mining approach for ddos attack detection. In: Kahng, H.-K., Goto, S. (Eds.), Information Networking. Networking Technologies for Broadband and Mobile Networks. Springer Berlin Heidelberg, pp. 943–950. http://dx.doi.org/10.1007/978-3-540-25978-7_95.

Kingma, D.P., Welling, M., 2014. Auto-encoding variational Bayes. arXiv:1312.6114.

Li, X., Chen, W., Zhang, Q., Wu, L., 2020. Building auto-encoder intrusion detection system based on random forest feature selection. Comput. Secur. 95, 101851. http://dx.doi.org/10.1016/j.cose.2020.101851, URL https://www.sciencedirect.com/science/article/pii/S0167404820301231.

Lin, S.-C., Chen, P.S., Chang, C.-C., 2014. A novel method of mining network flow to detect P2P botnets. Peer-to-Peer Netw. Appl. 7 (4), 645–654. http://dx.doi.org/10.1007/s12083-012-0195-x.

Mehmood, T., Md Rais, H.B., 2016. Machine learning algorithms in context of intrusion detection. In: 2016 3rd International Conference on Computer and Information Sciences (ICCOINS). pp. 369–373. http://dx.doi.org/10.1109/ICCOINS.2016.7783243.

Monshizadeh, M., Khatri, V., Atli, B.G., Kantola, R., 2018. An intelligent defense and filtration platform for network traffic. In: Wired/Wireless Internet Communications (WWIC). pp. 107–118. http://dx.doi.org/10.1007/978-3-030-02931-9_9.

Monshizadeh, M., Khatri, V., Atli, B.G., Kantola, R., Yan, Z., 2019. Performance evaluation of a combined anomaly detection platform. IEEE Access 7, 100964–100978. http://dx.doi.org/10.1109/ACCESS.2019.2930832.

Monshizadeh, M., Khatri, V., Gamdou, M., Kantola, R., Yan, Z., 2021. Improving data generalization with variational autoencoders for network traffic anomaly detection. IEEE Access 9, 56893–56907. http://dx.doi.org/10.1109/ACCESS.2021.3072126.

Monshizadeh, M., Yan, Z., 2014. Security related data mining. In: 2014 IEEE International Conference on Computer and Information Technology. pp. 775–782. http://dx.doi.org/10.1109/CIT.2014.130.

Muhlenbach, F., Lallich, S., 2009. A new clustering algorithm based on regions of influence with self-detection of the best number of clusters. In: 2009 Ninth IEEE International Conference on Data Mining. pp. 884–889. http://dx.doi.org/10.1109/ICDM.2009.133.

Ozdemir, A., Bolaños, M., Bernat, E., Aviyente, S., 2015. Hierarchical spectral consensus clustering for group analysis of functional brain networks. IEEE Trans. Biomed. Eng. 62 (9), 2158–2169. http://dx.doi.org/10.1109/TBME.2015.2415733.

Phadke, A., Kulkarni, M., Bhawalkar, P., Bhattad, R., 2019. A review of machine learning methodologies for network intrusion detection. In: 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC). pp. 272–275. http://dx.doi.org/10.1109/ICCMC.2019.8819748.

Pietro, A.D., philippe Vasseur, J., Cruz, M.J., 2016. Signature creation for unknown attacks. U.S. Patent 20160028750(A1).

Portela, F.G., Almenares Mendoza, F., Benavides, L.C., 2019. Evaluation of the performance of supervised and unsupervised machine learning techniques for intrusion detection. In: 2019 IEEE International Conference on Applied Science and Advanced Technology (ICASAT). pp. 1–8. http://dx.doi.org/10.1109/iCASAT48251.2019.9069538.

Ring, M., Wunderlich, S., Scheuring, D., Landes, D., Hotho, A., 2019. A survey of network-based intrusion detection data sets. Comput. Secur. 86, 147–167. http://dx.doi.org/10.1016/j.cose.2019.06.005, URL https://www.sciencedirect.com/science/article/pii/S016740481930118X.

Rosenberg, A., Hirschberg, J., 2007. V-Measure: A conditional entropy-based external cluster evaluation measure. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL). pp. 410–420", URL https://aclanthology.org/D07-1043.

Sabottke, C.F., Chen, D., Layman, L., Dumitras, T., 2019. How to trick the borg: threat models against manual and automated techniques for detecting network attacks. Comput. Secur. 81, 25–40. http://dx.doi.org/10.1016/j.cose.2018.07.022, URL https://www.sciencedirect.com/science/article/pii/S0167404818311283.

2021. Scikit-learn: machine learning in python. URL https://scikit-learn.org/stable/ [cited 2021-10-11].

Shakya, V., Makwana, R.R.S., 2017. Feature selection based intrusion detection system using the combination of DBSCAN, K-Mean++ and SMO algorithms. In: 2017 International Conference on Trends in Electronics and Informatics (ICEI). pp. 928–932. http://dx.doi.org/10.1109/ICOEI.2017.8300843.

Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A., 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: Proceedings of the 4th International Conference on Information Systems Security and Privacy - ICISSP. http://dx.doi.org/10.5220/0006639801080116.

Shiravi, A., Shiravi, H., Tavallaee, M., Ghorbani, A.A., 2012. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. Comput. Secur. 31, 357–374. http://dx.doi.org/10.1016/j.cose.2011.12.012, URL https://www.sciencedirect.com/science/article/pii/S0167404811001672.

2021. Sklearn metrics silhouette score. URL https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html [cited 2021-06-09].

Sun, J., Wang, X., Xiong, N., Shao, J., 2018. Learning sparse representation with variational auto-encoder for anomaly detection. IEEE Access 6, 33353–33361. http://dx.doi.org/10.1109/ACCESS.2018.2848210.

Sun, F., Wang, S., Zhang, C., Zhang, H., 2020. Clustering of unknown protocol messages based on format comparison. Comput. Netw. 179, 107296. http://dx.doi.org/10.1016/j.comnet.2020.107296, URL https://www.sciencedirect.com/science/article/abs/pii/S138912862030445X.

2022. Top 10 BEST intrusion detection systems (IDS) [2022 rankings]. Software testing help. URL https://www.softwaretestinghelp.com/intrusion-detection-systems/ [cited 2022-06-20]..

Turner, C., Joseph, A., 2017. A statistical and cluster analysis exploratory study of snort rules. Procedia Comput. Sci. 114, 106–115. http://dx.doi.org/10.1016/j.procs.2017.09.023, URL https://www.sciencedirect.com/science/article/pii/S1877050917318173, Complex Adaptive Systems Conference with Theme: Engineering Cyber Physical Systems, CAS October 30 – November 1, 2017, Chicago, Illinois, USA.

Wang, A., Blair, N., Belkhale, S., 2019. Encouraging categorical meaning in the latent space of a VAE. URL https://www.nathanblair.me/pdfs/Encouraging_categorical_meaning_in_the_latent_space_of_a_VAE.pdf.

Wang, X., Du, Y., Lin, S., Cui, P., Shen, Y., Yang, Y., 2020. adVAE: A self-adversarial variational autoencoder with Gaussian anomaly prior knowledge for anomaly detection. Knowl.-Based Syst. 190, 105187. http://dx.doi.org/10.1016/j.knosys.2019.105187.

Wang, W., Zhang, X., Gombault, S., Knapskog, S.J., 2009. Attribute normalization in network intrusion detection. In: 2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks. pp. 448–453. http://dx.doi.org/10.1109/I-SPAN.2009.49.

Wei, Y., Chow, K.-P., Yiu, S.-M., 2020. Insider threat detection using multi-autoencoder filtering and unsupervised learning. In: Peterson, G., Shenoi, S. (Eds.), Advances in Digital Forensics XVI. Springer International Publishing, pp. 273–290. http://dx.doi.org/10.1007/978-3-030-56223-6_15.

Xue-yong, L., Guo-hong, G., Jia-xia, S., 2010. A new intrusion detection method based on improved DBSCAN. In: 2010 WASE International Conference on Information Engineering, Vol. 2. pp. 117–120. http://dx.doi.org/10.1109/ICIE.2010.123.

Yadav, N., Scheib, E., Agasthy, R., 2016. Network behavior data collection and analytics for anomaly detection. U.S. Patent 20160359695(A1).

Yang, Y., Zheng, K., Wu, C., Yang, Y., 2019. Improving the classification effectiveness of intrusion detection by using improved conditional variational AutoEncoder and deep neural network. Sensors 19 (11), http://dx.doi.org/10.3390/s19112528, URL https://www.mdpi.com/1424-8220/19/11/2528.

Yang, Y., Zheng, K., Wu, B., Yang, Y., Wang, X., 2020. Network intrusion detection based on supervised adversarial variational auto-encoder with regularization. IEEE Access 8, 42169–42184. http://dx.doi.org/10.1109/ACCESS.2020.2977007.

Yousefi-Azar, M., Varadharajan, V., Hamey, L., Tupakula, U., 2017. Autoencoder-based feature learning for cyber security applications. In: 2017 International Joint Conference on Neural Networks (IJCNN). pp. 3854–3861. http://dx.doi.org/10.1109/IJCNN.2017.7966342.

Mehrnoosh Monshizadeh is finalizing her PhD at Electrical School of Aalto University, Finland. She is working at Nokia Bell Labs as security research specialist. Her research interests include cloud security, mobile network security, IoT security and data analytics.

Vikramajeet Khatri has M.Sc. degree in information technology from Tampere University of Technology, Finland. He is working as security specialist at Nokia Bell Labs. His research interests include intrusion detection, malware detection, IoT security and cloud security.

Raimo Kantola has a D.Tech degree in computer science from Helsinki University of Technology, Finland. He is a professor in networking technology at department of Comnet, Aalto University, Finland. His research interests include SDN, customer edge switching, trust in networks and cloud security.

Zheng Yan received the D.Sc. degree in technology from the Helsinki University of Technology, Espoo, Finland, in 2007. She is currently a Professor in the School of Cyber Engineering, Xidian University, Xi'an, China and a Visiting Professor and Finnish Academy Research Fellow at the Aalto University, Helsinki, Finland. Her research interests are in trust, security, privacy, and security-related data analytics. Dr. Yan is an area editor or an associate editor of IEEE Network, Internet of Things Journal, Information Fusion, Information Sciences, IEEE ACCESS, and Journal of Network and Computer Applications, etc. She served as a General Chair or Program Chair for numerous international conferences, including IEEE TrustCom 2015 and IFIP Networking 2021. She is a Founder Steering Committee Co-Chair of IEEE Blockchain conference. She received many awards, including Distinguished Inventor Award of Nokia, the Best Journal Paper Award issued by IEEE Communication Society Technical Committee on Big Data and the Outstanding Associate Editor of 2017/2018 for IEEE Access.