EasyChair Preprint

№ 13570

# UARC:Unsupervised Anomalous Traffic Detection with Improved U-Shaped Autoencoder and RetNet Based Multi-Clustering

Yunyang Xie, Kai Chen, Shenghui Li, Bingqian Li and Ning Zhang



June 6, 2024

# UARC:Unsupervised Anomalous Traffic Detection with Improved U-shaped Autoencoder and RetNet based Multi-Clustering

Yunyang Xie[1], Kai Chen[2] *, Shenghui Li[3], Bingqian Li[4], and Ning Zhang[5]

Hubei Key Laboratory of Distributed System Security, Hubei Engin eering Research Center on Big Data, Security School of Cyber Science and Engineering, Huazhong University Technology,Wuhan,430074, China
{xieyunyang,kchen,lishenghui,libq2022,zn_hust}@hust.edu.cn

**Abstract.** With the ongoing advancement of deep learning, modern network intrusion detection systems increasingly favor utilizing deep learning networks to improve their ability to learn traffic characteristics. To address the challenge of obtaining a substantial amount of labeled training data, many intrusion detection systems now focus on unsupervised anomaly detection methods. Despite this shift, researchers still face the daunting task of distinguishing a significant volume of anomalous traffic and dealing with data imbalance. To address these real-world challenges, we introduce UARC, a system capable of achieving unsupervised anomaly traffic detection through multi-clustering. UARC utilizes an enhanced U-shaped autoencoder and a feature fusion method incorporating Masked Retnet to effectively extract spatiotemporal features from network traffic. It combines these techniques with the HDBSCAN algorithm for multi-clustering of traffic, providing a form of reverse guidance for network learning. Experimental results on multiple datasets demonstrate that UARC can cluster various types of traffic with an impressive accuracy rate of up to 97.96%, while achieving a 99.70% AUC value for anomaly detection.

**Keywords:** Network intrusion detection · Unsupervised learning · Multi-Clustering · Auto-encoder · RetNet.

## 1 Introduction

As computer network technology continues to advance, the network traffic in our surroundings is experiencing exponential growth. Since the onset of COVID-19, scenarios involving remote work and communication have become unavoidable for people. According to a 2020 survey conducted by the Canadian Internet Registration Authority (CIRA), approximately two-thirds of IT professionals found themselves compelled to work from home due to COVID-19, resulting in a substantial surge in network traffic[37]. This surge has engendered increasingly severe network security predicaments. In the year 2021 alone, there were

* Corresponding author

over 66 instances of zero-day vulnerabilities being exploited in network attacks, nearly twice the count observed in 2020[35]. In the year 2022, losses attributable to network security issues soared to an alarming $4.35 million and exhibited a sustained upward trajectory[33]. The scope of network attacks is consistently broadening, marked by the continual emergence of various novel attack methodologies. This dynamic landscape has elevated intrusion detection technology to a pivotal research focus within the realm of network security.

Due to the demonstrated effectiveness of deep neural networks in learning concealed data features, the adoption of deep learning techniques has unequivocally become a prevailing trend in contemporary intrusion detection[26]. Nonetheless, the significant challenge in training neural networks lies in the requisite large-scale annotated data corpus[18]. Consequently, a substantial body of research has concentrated on unsupervised intrusion detection methodologies.

However, the majority of these endeavors primarily dichotomize samples into benign and anomalous traffic patterns based solely on reconstruction loss metrics[10]. This approach, nevertheless, presents several inherent predicaments: (i)Network attacks are typically mixed, and selecting specific attacks from different types of traffic for analysis is a challenging and costly task[3]; (ii)network attacks require multiple flows to complete, meaning that attack behaviors are highly coupled with temporal sequences.

Consider a DDOS attack as an illustration—it incessantly dispatches an extensive volume of data to the target, aiming to incapacitate its services. When scrutinizing its individual flow, it appears nearly indistinguishable from benign traffic[11]. Temporal features play a pivotal role in discriminating between various types of DDOS attack traffic.

Network intrusion detection also contends with challenges stemming from the issue of imbalanced data[16]. Current intrusion detection datasets commonly suffer from substantial class imbalance issues, with only a limited representation of high-risk attack traffic. This inherent imbalance poses a challenge for models to effectively capture and learn the distinctive features associated with high-risk attacks. Addressing such challenges often resorts to post-attack resampling strategies[8], but this approach is neither efficient nor devoid of risks.

In this paper, we introduce a feature fusion method that relies on an improved U-shaped network and the RetNet network. We employ clustering results to guide the feature extraction network, thereby facilitating improved separability in the network. Furthermore, we devise techniques to enrich the features, enhancing the discriminative power of coarse-grained information within the data, such as port numbers[22]. We evaluated our model on the CIC-IDS2017[24], CIC-IDS2018[29], and UNSW-NB15[23] datasets. The contributions of this paper are outlined as follows:

- We propose an improved U-shaped auto-encoder that effectively confines feature extraction to a Euclidean space, yielding well-separated features for the data.

- We firstly pioneer the application of RetNet in the field of network traffic intrusion detection, achieving superior results compared to Transformer models.
- We took into account all attack categories within the dataset without merging or altering them. The model ultimately yielded robust clustering results, demonstrating its capability to identify highly uncommon class attacks.

The paper is organized as follows: Section 2 presents an overview of related work; Section 3 provides detailed insights into the data, processing methods, and the architectural details of the model; Section 4 presents experimental results along with comparative analyses; and finally, Section 5 serves as the conclusion.

## 2 Related Works

### 2.1 Unsupervised anomaly detection

Unsupervised anomaly detection refers to a method used to identify anomalies or outliers within data without the reliance on labeled data[1]. This approach has gained prominence in the field of intrusion detection due to its ability to detect novel attack patterns. Unsupervised methods can be broadly categorized into two main classes: reconstruction-based methods and clustering-based methods.

**Reconstruction-based methods** PCA(Principal Component Analysis) is one of the most popular detection methods. However, PCA's feature transformation is confined to linear spaces and cannot capture nonlinear relationships among features, rendering it progressively less suited for increasingly intricate anomaly detection tasks[14].

As deep learning advances, reconstruction-based methods gain prominence in unsupervised anomaly detection. This approach involves employing autoencoder network architectures for feature extraction and identifying anomalies through disparities in reconstruction loss. In recent years, numerous methods, including AVAE introduced by An.J[5], which combines the probability distribution of variable variation with the variational auto-encoder, using reconstruction probability as an anomaly indicator.

Aytekin introduced CAE-l2[7], which incorporates a normalized layer with l2 constraints into CAE to enforce hypersphere constraints on the data. Andresini[6] employed multiple auto-encoders to learn both positive and negative samples of multi-channel data, integrating them with convolutional neural networks for anomaly detection. Shan Ali[4] fused the MKL(Multiple Kernel Learning) framework with multiple diverse deep auto-encoders to learn distinct feature combinations for DDoS attack detection .

These methodologies heavily rely on auto-encoders' data reconstruction capability. Nevertheless, deeper reconstruction networks may entail data compression, potentially causing data distortion and the subsequent loss of critical information. Furthermore, reconstruction methods are limited to distinguishing between normal and abnormal data, posing challenges in discerning various attack

types. This deficiency becomes apparent in the face of the escalating complexity of network attacks.

**Clustering-based methods** Cluster-based methods emphasize the spatial distribution characteristics of data and subsequently aggregate the data accordingly.

Ling Lai[17] introduced an improvement to the K-Means algorithm by utilizing sample path length as an anomaly score . However, this algorithm faces challenges when dealing with non-convex data clusters.

LinHua Gao[13] employed PCA for data dimensionality reduction and similarity-based partitioning, incorporating them into the spectral clustering algorithm for anomaly detection. However, the results of spectral clustering are highly dependent on the quality of the similarity matrix.

Huanhuan Zhang[39] applied the Fuzzy C-Means clustering algorithm for sample group partitioning, but it is highly sensitive to initialization, and different random centroid selections can lead to significantly different results.

Clustering-based methods typically focus on utilizing statistical features of data but overlook the time-series features of network traffic, which can result in suboptimal detection accuracy.

### 2.2 Time series anomaly detection

In recent years, researchers have increasingly recognized the significance of time-series information within network traffic data. Long Short-Term Memory networks (LSTM), recognized as a superior recurrent network in comparison to Recurrent Neural Network (RNN), excel in memory capabilities and are more adept at representing extensive sequential information[19, 15].

Ashish[32] introduced the Transformer model based on the attention mechanism for handling contextual information in sequential data. The model adopts an encoder-decoder architecture, leveraging a parallel computing design to enhance computational speed. The incorporation of multiple self-attention heads augments the network's prowess in feature extraction.

Wang Wei[34] introduced a self-supervised anomaly detection model based on the Transformer architecture, enhancing its capacity for extracting temporal features using a set of adaptable transformations, yielding promising results across multiple datasets.

Despite the impressive performance demonstrated by the Transformer, its parallel structure is accompanied by a reduction in performance when applied to recursive reasoning tasks.This issue was considered unsolvable until 2023 when Sun[31] introduced the RetNet model, which seamlessly combines parallelized training with efficient handling of recursive inference tasks. This breakthrough provides RetNet with outstanding capabilities in sequence feature extraction and processing.

### 2.3 Imbalanced cybersecurity data

The CIC-IDS2017 & 2018 datasets have long been staples in the field of network intrusion detection research. Nevertheless, the datasets' inherent challenges, characterized by severe class imbalance has been persistent concerns. A commonly adopted approach to mitigate these challenges involves collapsing the anomaly classes and subsampling to create a balanced dataset[24].

YuHua Yin[38] crafted an IDS system employing Birch and K-means clustering alongside an MLP classifier. Regrettably, within its test set, the proportion of normal data to abnormal data is closely balanced. Additionally, they consolidate various types of DDOS traffic and web attack traffic into a single category.

Alam.S[2] employed a residual CAE network augmented with L2 constraints to detect anomalous network traffic. Although their research employed the PVAMU-DDoS2020 dataset, they exclusively selected DDOS traffic and benign traffic for the composition of their training and testing sets.

While these approaches enhance the quality of results, they can introduce a disparity between experimental outcomes and test results in a real network environment. Thus, we opt to maintain the original categories of abnormal samples in our study and intentionally introduce challenges related to imbalance and extremely rare classes to validate the authentic effectiveness of our model.

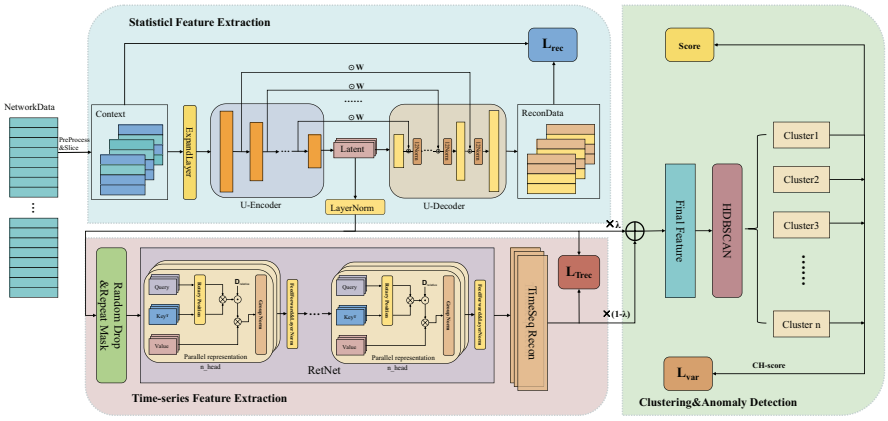



**Fig. 1.** The overview of UARC.

## 3 Proposed method

In this section, we delve into the specific intricacies of the UARC-based model. This model is crafted to tackle the challenges posed by vast amounts of unlabeled traffic and exceedingly rare novel attacks. By employing an enhanced U-shaped AutoEncoder network, our model endeavors to learn the projection representation of traffic data in a linearly separable space, thereby optimizing the utilization of Euclidean distance for clustering different types of traffic. Additionally, the model incorporates a RetNet featuring a masked context reconstruction module, facilitating the learning of time-based features in traffic.

The cross-fusion of these features contributes to an augmented clustering performance. Finally, we introduce an anomaly score designed to evaluate the anomaly level of traffic clusters.

Fig. 1 delineates the architecture of UARC, comprising four integral components: the data preprocessing module, statistical feature extraction module, time series feature extraction module, and clustering detection module. Notably, the input data representing network traffic is conventionally presented as features extracted from flows rather than raw packet data, stored in a .csv format. Data undergoes processing within a statistical feature extraction network to uncover the implicit relationships between diverse features. To more authentically simulate and extract time series information, we partition the continuous data into Contexts of size N. Prior to the extraction of time series features, we introduce random "dropouts" and "duplications" to mimic real-world network environments. In the context of time series feature extraction, we opt for the RetNet network architecture as a substitute for the Transformer. Experimental results substantiate that this choice indeed enhances the model's performance and efficiency.

Subsequently, we employ HDBSCAN for clustering, particularly when the number of target clusters is unknown. This serves as a guiding mechanism for the network to acquire improved clustering representations. Ultimately, the model generates an anomaly score, providing insight into the extent of anomaly within the clustering results.

### 3.1 Data processing

Our model functions exclusively within an unsupervised context, as delineated in Fig. 1. The learning process, depicted in the figure, initiates with inputs that include both a training set and a test set. The training set exclusively comprises normal network traffic, whereas the test set encompasses a diverse array of network attack traffics, with a minute proportion representing novel attack types. Following preprocessing by the preprocessing module, these data serve as inputs for the subsequent modules. The Data Preprocessing Module encompasses three key components: feature scaling, soft one-hot encoding, and feature enrichment.

Feature scaling involves normalizing input features to the [0,1] range through a max-min scaling strategy.

The soft one-hot encoding component adopts a more nuanced approach to one-hot encode input features. We have observed that when other input features are normalized to the [0,1] range, the abundance of zeros in the one-hot encoding can impede the network's ability to effectively learn the represented features. Hence, a minute value $\varepsilon$ is introduced to alter the one-hot encoding values, as depicted in Eq.1, where $K$ signifies the total number of categories for the encoding target.

$$x = \begin{cases} 1-\varepsilon & if\ x \quad = 1 \\ \frac{\varepsilon}{K-1} & if\ x \quad = 0 \end{cases} \tag{1}$$

In the context of network traffic analysis, features like protocol type and port number often serve as vital discriminative factors for identifying malicious attacks. Hence, through the feature enrichment component, we carefully select these features and subject them to a series of transformations to derive new feature representations. This set of transformations may involve a combination of multiple distinguishable nonlinear functions. To elaborate further, we define this process as the transformation of the original feature $x_i$ through a set of transformations $T_i$ into $\{x_1, x_2, ..., x_n\}$. The functions composing $T_i$ can encompass operations such as $|sin(x)|$, $x^2\sqrt{x}$, $log(x+1)$, and so on.

### 3.2 Improved U-shaped autoencoder

We represent the auto-encoder with input $I$ as $D(E(I))$, where $E(I)$ serves as the input to the decoder. For each layer of the encoder, we employ $f_i$ to denote the combination of its linear layer and the GeLU activation function, as outlined in Eq.(2).

$$f_i(x) = GeLU(xW_f + b_f) \tag{2}$$

Consequently, the representation of the n-layer encoder is articulated in Eq.(3).

$$E(I) = f_n(f_{n-1}(...f_1(I))) \tag{3}$$

We maintain a record of the output from each layer within the encoder, denoted as $L_E\{l_n, l_{n-1}, ..., l_1\}$, for the purpose of data reconstruction. In this context, $l_n$ represents the output of the encoder $E(I)$, which functions as the input for the decoder. As for the other $l_i$ layers within the decoder, they are subject to multiplication by A collection of trainable weight matrices $W_u$ and added to the output of the preceding layer. Subsequently, after undergoing L2 normalization, these values are employed as input for the subsequent layer. The incorporation of L2 normalization serves to introduce Euclidean space constraints onto the data.

We use $g_i$ to represent the combination of the linear layer and activation function within the decoder, as exemplified in Eq.(4).

$$g_i(x) = GeLU(\frac{x}{\sqrt{x^T x}}W_g + b_g)) \tag{4}$$

The linear layers in the decoder decrease in the reverse order as compared to the encoder. The interplay between the shapes and sizes of the input data for $f_i$ and $g_i$ should align with the conditions outlined in Eq.(5).

$$Input\_shape : I_{f_i} = I_{g_{n-i+1}} \quad i \in [1, n] \tag{5}$$

We compute the MSE loss between the input $I$ and the reconstructed output $I^R$ as the loss function for the entire auto-encoder network, with the goal to minimizing the value of Eq.(6). $\mathcal{K}$ represents the index set of the input $I$.

$$\mathcal{L}_{rec} = \frac{\sum_{k \in \mathcal{K}} (I_k - I_k^R)^2}{|\mathcal{K}|} \tag{6}$$

To ensure the stability of the forward input distribution for the subsequent modules, we introduce layer normalization to the output $l_n$ of the hidden layer. The resultant dimensionality reduction representation is represented as $L_{sta}$. Consequently, the output of the entire auto-encoder is illustrated in Eq.(7),where $\gamma$ and $\beta$ are derived through network computations.

$$L_{sta} = \frac{l_n - \overline{l_n}}{\sqrt{\widehat{l_n} + \varepsilon}}\gamma + \beta \tag{7}$$

### 3.3 Retnet based context reconstruction

To better capture the time-series features of network traffic, we have introduced a masking module to RetNet, thereby augmenting the "complexity" associated with reconstructing traffic patterns. $\mathcal{N}$ represents the set of input data, which we partition into $m = \lfloor \frac{|\mathcal{N}|}{c} \rfloor$ temporal sequence blocks of size $c$, denoted as $\mathcal{C}\{c_1, c_2, ..., c_m\}$. For each data point $c_i\{x_1, x_2, ...., x_j\} \in \mathcal{C}$ Eq.(8), we introduce random transformations: with a 10% probability, we set it to zero, simulating packet loss events in network transmission; with a 10% probability, we replace it with another data point from $c_i$, mimicking replay events; and with a 80% probability, we retain the original data.

$$x_j = \begin{cases} 0 & 10\%\ of\ the\ time \\ x_k & 10\%\ of\ the\ time \\ x_j & 80\%\ of\ the\ time \end{cases} \qquad x_j, x_k \in c_i\ ,\ j \neq k \tag{8}$$

The preprocessed data $L^M$ is employed as the input for the RetNet network, which focuses on learning temporal features by reconstructing the replaced segments of data. RetNet maintains a constant latent dimension. In our approach, we employ the ParallelRetention structure within RetNet as the training network, and the final reconstructed output is derived through the RecurrentRetention, aiming to maximize the reconstruction loss for anomalous traffic.

ParallelRetention consists of sub-modules, including MSR (Multi-Scale Retention) and FFN (Feed-Forward Network), which are stacked in parallel,as outlined in Eq.(9). The core objective of ParallelRetention is to expedite feature learning through parallel computations.

$$\begin{aligned} H_l &= MSR(LayerNorm(L_l^M)) + L_l^M \\ L_{l+1}^M &= FFN(LayerNorm(H_l)) + H_l \end{aligned} \tag{9}$$

The multi-head attention mechanism of MSR is implemented by multiple retention heads. For each retention head, we apply trainable transformations denoted as $W$ to the input $\mathcal{C}$ , resulting to generate $Q, W, V$. To capture the relative contextual relationships within the Context, we employ rotary position embedding on $Q$ and $W$, as demonstrated in Eq.(10) and Eq.(11).

$$Q_n = \mathcal{C}W_Q, K_n = \mathcal{C}W_K, V = \mathcal{C}W_V \tag{10}$$

$$Q = Q_n e^{in\theta}, K = K_n (e^{in\theta})^{\dagger} \tag{11}$$

We compute the inner product of $Q$ and $K^T$, which is subsequently adjusted by the scaling matrix $D \in \mathbb{R}^{|\mathcal{C}|\times|\mathcal{C}|}$ (Eq.(12)). This scaling operation is instrumental in causal blocking and constraining the network's acquisition of relative positional information, thereby augmenting the learning weights for data points that are in closer temporal proximity. This encourages the network to place greater emphasis on temporally adjacent traffic.

$$D = \begin{cases} \gamma^{n-m}, & n \geq m \\ 0, & n < m \end{cases} \tag{12}$$

Therefore, we can formulate the complete structure of the retention head, as expressed in Eq.(13).

$$RetentionHead(\mathcal{C}) = GroupNorm((QK^T \odot D)V) \tag{13}$$

During the reconstruction of the test dataset, we utilize RecurrentRetention instead of ParallelRetention, as depicted in Fig. 2. Notably, $W_Q$, $W_K$, and $W_V$ retain the same definitions as outlined in Eq.(10) and Eq.(11). It is noteworthy that due to the prior learning of $W_Q$, $W_K$, and $W_V$ in ParallelRetention, recursive reconstruction can be efficiently accomplished within $O(n)$ time. During the

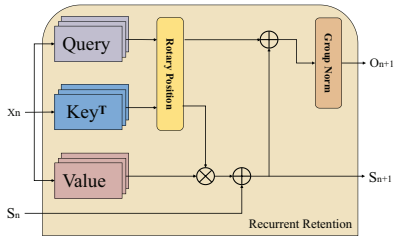



**Fig. 2.** The structure of RecurrentRetention.

training of the RetNet network, we employ cosine similarity as a guiding metric for the network's learning process. Herein, we denote the input data for the masked reconstruction network as $L$, the reconstructed data as $L^R$, leading to the formulation of the loss function as detailed in Eq.(14).

$$\mathcal{L}_{Trec} = 1 - \frac{L \cdot L^R}{||L||\ ||L^R||} \tag{14}$$

We reintegrate statistical features with temporal features to further enhance the fusion's efficacy. The ultimate reduced-dimensional representation produced by the comprehensive feature fusion network is presented in Eq.(15). Notably, $\lambda$ serves as a weighting factor, allowing for the adjustment of the reintegration's relative influence.

$$L_R = \lambda \times L_{sta} + (1 - \lambda) \times L^R \tag{15}$$

The loss function for the complete feature fusion network is represented by Eq.(15). Notably, the scaler serves as a scaling factor to fine-tune the weighting

of the RetNet network's influence within the broader network learning process.

$$\mathcal{L}_{FE} = \mathcal{L}_{rec} + scaler \times \mathcal{L}_{Trec} \tag{16}$$

### 3.4 Multi-Cluster Network

The distribution density of network traffic data exhibits non-uniformity. Hence, we employ the HDBSCAN algorithm. The model utilizes the mutual reachability distance in lieu of the direct distance between two samples(Eq.(17)), thereby bolstering robustness to uniformly distributed samples.

$$\begin{aligned} core_k(x) &= d(x, N^k(x)) \\ d_{mreach-k}(a,b) &= max\{core_k(a), core_k(b), d(a,b)\} \end{aligned} \tag{17}$$

HDBSCAN [21] algorithm does not require prior knowledge of the number of clusters. Following L2 normalization, the squared Euclidean distance of the extracted features becomes equivalent to the cosine distance. Hence,for the noise points identified after clustering, marked as -1, we calculate their cosine similarity with each cluster center and compare it with a threshold. Data within the threshold will be merged into the cluster with the highest similarity without recalculating the cluster center. Points outside the threshold form new clusters. The threshold is determined by the average inter-cluster distance.

During the training process, we use the CH-score as the loss function for the clustering phase, guiding the network to learn better clustering shapes. In Eq.(18), we denote the total data points as $\mathcal{N}$, the total number of clusters obtained as $\mathcal{S}$, $c_e$ represents the samples of cluster centers, $n_q$ represents the total number of samples within cluster q, and $c_q$ represents the sample set of cluster q.

$$\mathcal{L}_{cluster} = \frac{\sum_{q=1}^{k} n_q (c_q - c_e)(c_q - c_e)^T}{\sum_{q=1}^{k} \sum_{x \in c_q} (x - c_q)(x - c_q)^T} \cdot \frac{(\mathcal{N} - \mathcal{S})}{(\mathcal{S} - 1)} \tag{18}$$

The entire network's loss function is defined as:

$$\mathscr{L} = \mathcal{L}_{FE} + \mathcal{L}_{cluster} + \frac{1}{\mathcal{S}} \tag{19}$$

where, $\frac{1}{\mathcal{S}}$ is a penalty term to prevent the network from learning too few clusters.

During the detection phase, we calculate the average reconstruction loss within each cluster obtained after clustering, along with the cluster's standard deviation, to compute the anomaly score(Eq.(21)). Due to the potential disparity in scale between the cluster's standard deviation and the average reconstruction loss, we need to apply an amplification factor to the average reconstruction loss. We sort the resulting clusters by their anomaly scores in ascending order, and any cluster with a score higher than that obtained from normal traffic will be considered as an anomaly.

$$Score = \sqrt{\frac{1}{|\mathcal{S}|} \sum_{x_i \in c_q} (x_i - \overline{x})^2} + \beta \cdot \frac{\sum_{x_i \in c_q} \mathcal{L}_{FE}}{|\mathcal{S}|} \tag{21}$$

## 4 Experiments

This section will provide an overview of the experimental setup and results analysis. Experimental setup includes datasets used and comparison methods. To demonstrate the effectiveness of our model, we conducted tests and cross-dataset evaluations on three datasets, comparing them with common unsupervised anomaly detection methods. Additionally, we will perform ablation experiments to validate our contributions. Finally, sensitivity tests on hyperparameters of the spatiotemporal feature fusion network will be conducted to observe their impact on network learning.

### 4.1 Datasets

We used the commonly used datasets: CIC-IDS2017 [24], CIC-IDS2018 [29] and UNSW-NB15 [23]. These datasets are derived from real traffic, providing statistical features and timestamps. Additionally, they contain a very small proportion of attack types. To simulate complex network traffic scenarios, we continuously selected a portion of traffic from different dates within each dataset and combined them to create the test sets, with 30% of normal traffic used for the training sets. Below, we will show the proportions and quantities of each class of traffic in these representative datasets.

**CIC-IDS2017 & CIC-IDS2018:**The CIC-IDS2017&2018 datasets were created and released by the Canadian Institute for Cybersecurity. They are constructed by capturing real network traffic to reflect various network traffic patterns and attacks encountered in real-world networks. These datasets contain data from different categories of network traffic, including normal traffic and various types of network attacks such as DoS (Denial of Service) attacks, DDoS (Distributed Denial of Service) attacks, malware, and scans, among others. The datasets provide a wide range of statistical features, including features based on communication protocols, packet sizes, duration, source and destination IP addresses, and more. The detailed information about the proportions of different types of traffic in their representative datasets is shown in Table 1.

**UNSW-NB15:**The UNSW-NB15 dataset, formulated by Moustafa and Slay (2015) at the Network Security Lab of the Australian Centre for Cyber Security, employing the IXIA PerfectStorm tool, encompasses 9 distinct attack categories. It boasts a comprehensive array of 49 features associated with each traffic record. Detailed statistics concerning the distribution of various traffic types within this representative dataset are elucidated in Table 1.

### 4.2 Comparison Methods

We will concurrently employ both traditional methodologies and deep learning techniques to conduct a comparative analysis within the domain of anomaly detection. Furthermore, we will juxtapose these approaches with the methodologies commonly utilized in multi-class classification tasks for comprehensive assessment.

**Table 1.** Structure representative dataset.

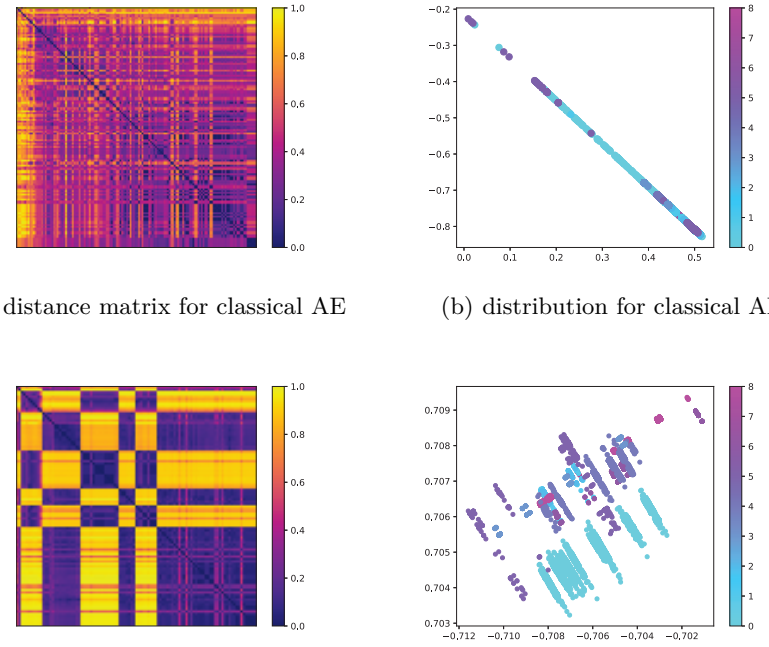
| CIC-IDS2017 | | | CIC-IDS2018 | | | UNSW-NB15 | | |
|---|---|---|---|---|---|---|---|---|
| **Class** | **Instances** | **Percentage** | **Class** | **Instances** | **Percentage** | **Class** | **Instances** | **Percentage** |
| Benign | 15,000 | 73.14 | Benign | 15,000 | 89.37 | Benign | 13,800 | 78.41 |
| DoS Hulk | 2,000 | 9.75 | FTP-BruteForce | 2,000 | 14.89 | Fuzzers | 1,487 | 8.45 |
| DoS GoldenEye | 2,000 | 9.75 | SSH-Bruteforce | 2,000 | 14.89 | Exploits | 1,311 | 7.45 |
| DoS slowloris | 2,000 | 9.75 | DoS attacks-GoldenEye | 2,000 | 14.89 | Reconnaissance | 477 | 2.71 |
| DoS Slowhttptest | 2,000 | 9.75 | DDOS attack-HOIC | 1,000 | 7.45 | DoS | 226 | 1.28 |
| PortScan | 1,500 | 7.31 | Brute Force -Web | 611 | 4.55 | Generic | 189 | 1.07 |
| FTP-Patator | 1,000 | 4.88 | DoS attacks-Slowloris | 500 | 3.72 | Shellcode | 64 | 0.36 |
| SSH-Patator | 1,000 | 4.88 | Brute Force -XSS | 230 | 1.71 | Worms | 25 | 0.14 |
| Heartbleed | 10 | 0.05 | SQL Injection | 87 | 0.65 | Backdoors | 21 | 0.12 |
| | Toatl | 20,510 | | Toatl | 13,428 | | Toatl | 17,600 |

- **IF:**The Isolation Forest algorithm defines anomalies as data points that are easily isolated, meaning they are distant from densely populated clusters and exhibit sparse distributions. It performs anomaly detection by recursively partitioning the dataset until all sample points are isolated, thus identifying outliers with shorter paths [20].
- **DAGMM:**DAGMM seamlessly integrates the dimensionality reduction and density estimation processes, facilitating an end-to-end joint training approach [40].
- **DEEP-SVDD:**This method leverages neural networks for the extraction of data features and confines the normal samples within a hypersphere. Anomalous samples are distanced from this hypersphere, residing outside of its boundaries[28].
- **CAE-l2:**CAE-l2 involves replacing the intermediate layer of the autoencoder with an L2 normalization layer, which enhances the compatibility of feature extraction with the Euclidean distance metric. As a result, it leads to improved clustering accuracy when applying k-means clustering for graphical representation [7].
- **GOAD:** GOAD projects data onto distinct regions through geometric transformations and maps these transformed data into a new sample space. Under the concept of single-class classification, each geometric transformation subspace is mapped into a sphere [9].
- **THOC:**THOC employs an extended recurrent neural network with skip connections and integrates it hierarchically with a clustering network to capture temporal dynamic features across multiple scales [30].
- **Whipser:**Whipser utilizes the ordered information represented by frequency domain features to achieve bounded information loss, ensuring high detection accuracy, while also constraining the feature dimension, thus achieving high detection throughput [12].
- **CRMC:** CRMC utilizes a comparative learning approach based on residual autoencoders to extract statistical features and employs GRU to extract time series features. It also develops a clustering tree structure based on the DBSCAN algorithm, which determines abnormal data based on the tree height [27].

Among the baseline methods mentioned above, Isolation Forest represents a traditional machine learning algorithm, whereas the other methods are rooted

in deep learning techniques. Notably, CAE-l2 and Multiple-clustering support multi-clustering testing. To facilitate a more robust comparison of model effectiveness, a same data preprocessing pipeline was applied across all experiments. Notably, the input dimensions for both the CIC-IDS2017 and CIC-IDS2018 datasets amount to 90 dimensions, while the UNSW-NB15 dataset comprises 55 dimensions. DEEP-SVVD employs data compression into a spherical space, where the compression space is configured to be 20 dimensions. Conversely, the GOAD algorithm leverages a one-dimensional convolutional neural network for data transformation, with a mapping space dimensionality of 40. In contrast, the other methods operate within a 10-dimensional compression space. Each method was executed 10 times on each dataset, spanning 500 epochs per run, and the final experimental results were computed as the arithmetic mean of these trials.

### 4.3 Experiments results



(a) distance matrix for classical AE

(b) distribution for classical AE

(c) distance matrix for improved U-AE

(d) distribution for improved U-AE

**Fig. 3.** Experimental results on clustering separability.

**Multi-clustering** To demonstrate the effectiveness of the feature fusion network, we computed the Euclidean distance matrix of extracted features and the distribution of clustered samples (Fig. 3). In Fig. 3(a) and Fig. 3(b), the classical autoencoder architecture was employed, whereas in Fig. 3(c) and Fig. 3(d), we utilized the improved U-shaped autoencoder structure that we introduced. It is

evident that in Fig. 3(c), UARC has introduced conspicuous patterns of proximity and sparsity among the samples, signifying strong separation capabilities. This assertion is reinforced in Fig. 3(d) as well.

In the experiments for clustering accuracy, we extended the commonly used dimensionality reduction algorithms in combination with the HDBSCAN algorithm on top of the baseline methods that support multi-clustering. Furthermore, to highlight the superiority of the RetNet network in our model compared to the Transformer, we incorporated the comparison with Transformer in this section.

Considering the uncertainty in the number of clusters generated by the HDBSCAN algorithm and the fact that we cannot pre-determine the number of traffic types in a real network, we introduce a metric(Eq.(20)) to simultaneously represent traffic detection rate and accuracy in clustering traffic of the same type. We use $Ss_1, s_2, ..., s_i$ to represent the predicted cluster set, where $e_i$ represents the most prevalent traffic type within each cluster, $p_{e_i}$ denotes its prevalence, $n_{e_i}$ signifies the number of instances in that cluster, and $N_E$ is the total count of traffic of type E. $\delta(e_i, E) = n_{e_i}$ if $e_i = E$ (otherwise, it is 0).

$$RecallAcc_E = \sum_{s_i \in S} \frac{p_{e_i} \times \delta(e_i, E)}{N_E} \tag{20}$$

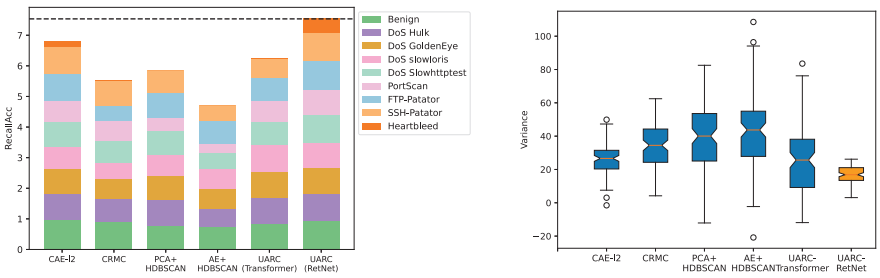
However, since we know the number of traffic types in the dataset, we further perform secondary clustering on the cluster centers of $S$ using the AgglomerativeClustering algorithm. We evaluate Eq.(21) using the standard unsupervised clustering ACC [36]. In this equation, $l_i$ represents the true cluster labels, $p_i$ stands for the predicted cluster labels, $Map()$ denotes the best mapping function to arrange predicted labels for the optimal alignment with true labels, and $\theta(x, y) = 1$ if $x = y$ (otherwise, it's 0).

$$Acc = \max_{Map} \frac{\sum_{i=1}^{|I|} \theta(l_i, Map(p_i))}{|I|} \tag{21}$$

Fig. 4(a) shows the RecallAcc performance of UARC compared to other models facing different types of traffic, while Fig. 4(b) illustrates the variance distribution of sub-classes in the clustering results of different models. Table 2 provides the comparison results for standard ACC, with bold sections representing the best results.It is evident that UARC demonstrates more accurate clustering accuracy on each class of traffic group and has smaller inter-class variance. This demonstrates that UARC can more accurately differentiate between different types of traffic in a more compact manner. Additionally, we have replaced some structures in UARC for comparison, which also proves the effectiveness of the proposed improvements.

The performance decrease on the standard ACC is attributed to the constraint on the number of clusters. We believe that loosening the requirement on the number of clusters appropriately can help improve the accuracy of each subclass.

Our model showcases exceptional performance in the unsupervised clustering of unknown anomalous traffic. Even the non-top results are closely aligned with




(a) The multi-clustering result of RecallAcc. (b) Inter-class variance of each model

**Fig. 4.** Clustering results presentation of each model

the top results. These anomalous traffic types often exhibit more pronounced differences compared to benign traffic, such as the repetitive packet lengths in DDoS attacks. UARC excels in handling extremely rare classes, providing evidence of the effectiveness and superiority of RetNet in the realm of network traffic anomaly detection. It's worth highlighting that UARC using Transformer exhibits noticeable differences in results compared to the UARC employing RetNet. This discrepancy can be attributed to challenges related to the depth of temporal networks. In order to improve UARC's detection efficiency, we adopted a smaller hidden layer size (64 dimensions) and a shallower network depth (4 layers) in the temporal feature extraction network. In such a scenario, where feature extraction capabilities are relatively weaker, RetNet, with its robust inference capabilities, clearly demonstrates its strengths.

**Table 2.** The result of standard ACC with baseline methods.

| Method | CIC-IDS2017 | CIC-IDS2018 | UNSW-NB15 |
|---|---|---|---|
| CAE-l2 | 0.6537 | 0.6672 | 0.5987 |
| CRMC | 0.6254 | 0.6437 | 0.3749 |
| PCA+HDBSCAN | 0.4563 | 0.4798 | 0.2375 |
| AE+HDBSCAN | 0.5712 | 0.5968 | 0.3357 |
| UARC(Transformer) | 0.6489 | 0.6823 | 0.5991 |
| UARC(RetNet) | **0.7102** | **0.7266** | **0.6096** |

The experimental results indicate that researchers can use our model to cluster unknown traffic sets and quickly identify specific traffic types through simple analysis of the clustering results. Moreover, our model can also identify a small number of attack traffic instances within traffic sets and cluster them separately, facilitating rapid detection of new attack samples for research. In cross-dataset experiments, we trained our model on the IDS2018 dataset and applied it to the IDS2017 dataset. The results show that its effectiveness decreases only slightly,

underscoring the usability and versatility of our model in real-world scenarios.The optimal mapping can be determined using the Kuhn-Munkres algorithm [25].

**Abnormal detection** Table.3 presents a comparative analysis of the performance between our model and other methods in the context of anomaly detection, incorporating metrics such as AUC and F1 scores. The most notable results are highlighted in bold. In our experimental setup, we categorize data from normal traffic clusters with a detection rate below 90% as anomalies, ensuring a rigorous approach to anomaly detection. Despite this stringent criterion, our model consistently outperforms the comparison methods across various metrics. Notably, when compared to the IDS2017 and IDS2018 datasets, the test results on the UNSW-NB15 dataset exhibit a noticeable decrease in performance. This can be attributed to two key factors: (i) UNSW-NB15 contains a relatively smaller proportion of anomalous data, making it an "almost normal" dataset; (ii) the UNSW-NB15 dataset offers fewer statistical features, and although we have enriched its feature set, it still falls short in terms of expressive power compared to the IDS2017 and 2018 datasets.

**Table 3.** The anomaly detection result with baseline methods.

| | CIC-IDS2017 | | | CIC-IDS2018 | | | 2018cross2017 | | | UNSW-NB15 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Method | AUC | Recall | F1 | AUC | Recall | F1 | AUC | Recall | F1 | AUC | Recall | F1 |
| IF | 0.5339 | 0.8241 | 0.7889 | 0.5104 | 0.9017 | 0.7439 | 0.4339 | 0.8241 | 0.6889 | 0.5127 | 0.8933 | 0.7307 |
| CRMC | 0.8776 | 0.9094 | 0.2987 | 0.8294 | 0.8392 | 0.3833 | 0.6430 | 0.9562 | 0.2064 | 0.8135 | 0.7468 | 0.7310 |
| THOC | 0.8032 | 0.8707 | 0.8856 | 0.8987 | 0.8611 | 0.9058 | 0.7521 | 0.7336 | 0.7195 | 0.7400 | 0.7907 | 0.7834 |
| Whipser | 0.6694 | 0.9157 | 0.2561 | 0.7852 | 0.7364 | 0.3506 | 0.5002 | 0.5409 | 0.1564 | 0.6074 | 0.8737 | 0.2999 |
| CAE-l2 | 0.9716 | 0.9367 | 0.9045 | 0.9170 | 0.9697 | 0.9590 | 0.8846 | 0.7273 | 0.7123 | 0.8846 | 0.8507 | 0.8511 |
| DEEP-SVDD | 0.6803 | 0.7016 | 0.6930 | 0.7297 | 0.5556 | 0.6890 | 0.6108 | 0.6212 | 0.6151 | 0.7170 | 0.6850 | 0.6743 |
| DAGMM | 0.9298 | 0.9445 | 0.9371 | 0.9296 | 0.9451 | 0.9373 | 0.9300 | 0.9453 | 0.9376 | 0.9301 | 0.9442 | 0.9371 |
| GOAD | 0.9216 | **1.0000** | 0.9020 | 0.9712 | 0.8238 | 0.7967 | 0.8125 | 0.8364 | 0.8596 | 0.8876 | 0.9013 | 0.8901 |
| UARC(Transformer) | 0.8547 | 0.8431 | 0.8559 | 0.8589 | 0.8546 | 0.8182 | 0.8547 | 0.8431 | 0.8759 | 0.8621 | 0.8570 | 0.8971 |
| UARC(RetNet) | **0.9944** | 0.9970 | **0.9970** | **0.9902** | **1.0000** | **0.9933** | **0.9462** | **1.0000** | **0.9892** | **0.9451** | **0.9502** | **0.9430** |

We also delved into the performance of various methods when confronted with imbalanced datasets. We conducted experiments by selecting benign samples from the representative CIC-IDS2018 dataset and combining them with anomalous samples in proportions ranging from 1%, 10%, 20%, to 100%. For each well-trained baseline method, we ran experiments on the test set five times and calculated the average results. Fig. 5 illustrates that, in general, most methods exhibit superior performance under balanced data conditions compared to imbalanced ones. Notably, DEEP-SVDD and DAGMM demonstrate favorable experimental results only when the benign and anomalous sample quantities are approximately equal.

Our experiments have affirmed that our method consistently maintains exceptional performance when dealing with imbalanced data. This can be attributed to the guidance provided by the clustering network to the feature learning network, enabling even benign traffic to exhibit variations due to diverse communication scenarios. In essence, UARC utilizes clustering to segment the imbalanced sample population into balanced groups comprising numerous small

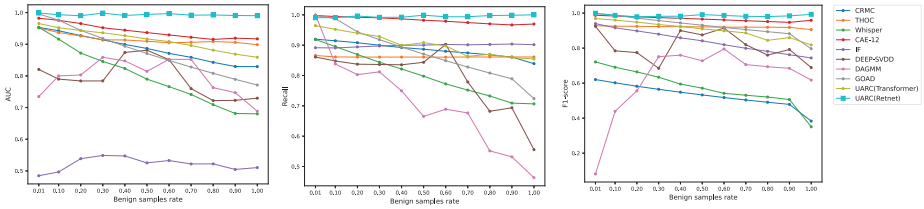
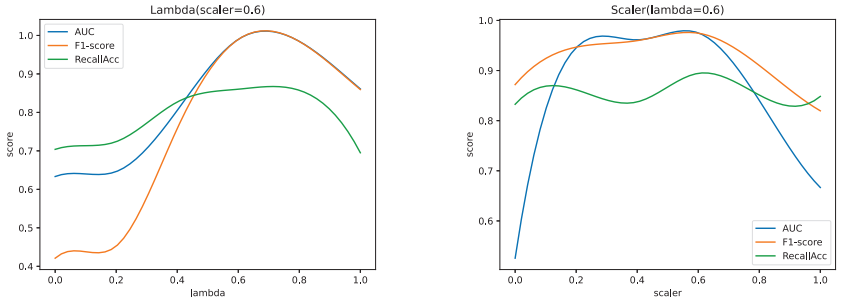



**Fig. 5.** Performance of different methods on imbalanced datasets.

clusters. Despite the considerable macro-level imbalance between benign and anomalous samples, at a finer-grained communication partitioning level, they achieve balance.

**Parameter sensitivity test** In Eq.(15) and Eq.(16), we have discussed the factors that influence the final dimensionality reduction and the weight of network learning: $\lambda$ and *scaler*. It is evident that when $\lambda = 1$, the model's dimensionality reduction reduces to just the output of the U-shaped auto-encoder, whereas when $\lambda = 0$, the dimensionality reduction comprises solely the output of RetNet. Regarding *scaler*, when it equals 0, RetNet no longer guides network learning. We conducted parameter tests within the range of $[0, 1]$, with the fixed value of the other parameter determined by optimizing for the best results(Fig. 6). Our evaluation metrics encompass AUC, F1-score, and RecallAcc. It is notable that, when $\lambda$ and scaler assume extreme values, the model fails to demonstrate significant performance improvements. In our experiments, we segment contin-




(a) The influence of $\lambda$. (b) The influence of *scaler*.

**Fig. 6.** The influence of parameters $\lambda$ and *scaler* on UARC.

uous traffic into contexts for processing. Recognizing that the choice of context size can impact the extent of contextual information learned by the model, we conducted experiments to evaluate the influence of context size on model performance, as illustrated in Fig. 7. We utilized the CIC-IDS2018 representative

dataset and examined a range of context sizes from 10 to 100. The experimental findings reveal that larger context sizes result in a performance decline. Our analysis of the clusters responsible for increased misclassifications with larger context sizes unveiled a primary reason: larger contexts lead to benign and anomalous samples being included in each other's contexts, prompting the temporal model to acquire inappropriate temporal relationships. Hence, practical usage should prioritize the maintenance of a smaller context size, typically within the range of 10-30.

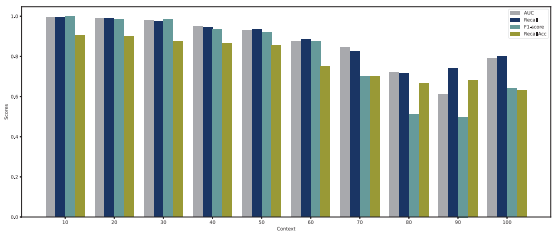


**Fig. 7.** Performance of different context sizes on UARC.

## 5 Conclusion and future work

Intrusion detection plays a crucial role in ensuring network security. Unsupervised intrusion detection methods offer a solution to the challenges posed by high annotation costs and imbalanced data. In this study, we implemented a deep learning model that integrates temporal and spatial features through clustering to enhance feature extraction and cluster separation. Experimental tests demonstrate our model's proficiency in clustering different types of attack traffic and identifying rare attack types. Furthermore, our work highlights the potential application prospects of RetNet and robust inference capabilities in the field of intrusion detection.

However, our model did not provide interpretable reasons for these results. Robust interpretability provides credible support for intrusion detection systems. As a prospect for future research, we intend to further enhance the interpretability of our model.

## References

1. Agarwal, S.: Data mining: Data mining concepts and techniques. In: 2013 international conference on machine intelligence and research advancement. pp. 203–207. IEEE (2013)
2. Alam, S., Alam, Y., Cui, S., Akujuobi, C.M.: Unsupervised network intrusion detection using convolutional neural networks. In: 2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC). pp. 0712–0717. IEEE (2023)
3. AlEroud, A., Karabatis, G.: Using contextual information to identify cyber-attacks. Information fusion for cyber-security analytics pp. 1–16 (2017)
4. Ali, S., Li, Y.: Learning multilevel auto-encoders for ddos attack detection in smart grid network. IEEE Access **7**, 108647–108659 (2019). https://doi.org/10.1109/ACCESS.2019.2933304

5. An, J., Cho, S.: Variational autoencoder based anomaly detection using reconstruction probability. Special lecture on IE **2**(1), 1–18 (2015)
6. Andresini, G., Appice, A., Mauro, N.D., Loglisci, C., Malerba, D.: Multi-channel deep feature learning for intrusion detection. IEEE Access **8**, 53346–53359 (2020). https://doi.org/10.1109/ACCESS.2020.2980937
7. Aytekin, C., Ni, X., Cricri, F., Aksu, E.: Clustering and unsupervised anomaly detection with l 2 normalized deep auto-encoder representations. In: 2018 International Joint Conference on Neural Networks (IJCNN). pp. 1–6. IEEE (2018)
8. Bagui, S., Li, K.: Resampling imbalanced data for network intrusion detection datasets. Journal of Big Data **8**(1), 1–41 (2021)
9. Bergman, L., Hoshen, Y.: Classification-based anomaly detection for general data. arXiv preprint arXiv:2005.02359 (2020)
10. Cotroneo, D., Paudice, A., Pecchia, A.: Empirical analysis and validation of security alerts filtering techniques. IEEE Transactions on Dependable and Secure Computing **16**(5), 856–870 (2017)
11. Devi, R.S., Bharathi, R., Kumar, P.K.: Investigation on efficient machine learning algorithm for ddos attack detection. In: 2023 International Conference on Computer, Electrical & Communication Engineering (ICCECE). pp. 1–5 (2023). https://doi.org/10.1109/ICCECE51049.2023.10085248
12. Fu, C., Li, Q., Shen, M., Xu, K.: Realtime robust malicious traffic detection via frequency domain analysis. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. pp. 3431–3446 (2021)
13. Gao, L., Chen, H.: Abnormal detection of blast furnace condition using pca similarity and spectral clustering. In: 2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA). pp. 2198–2203 (2018). https://doi.org/10.1109/ICIEA.2018.8398075
14. Goldstein, M., Uchida, S.: A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. PloS one **11**(4), e0152173 (2016)
15. Graves, A., Graves, A.: Long short-term memory. Supervised sequence labelling with recurrent neural networks pp. 37–45 (2012)
16. Gupta, N., Jindal, V., Bedi, P.: Cse-ids: Using cost-sensitive deep learning and ensemble algorithms to handle class imbalance in network-based intrusion detection systems. Computers & Security **112**, 102499 (2022)
17. Lai, L.: Abnormal data detection method of web database based on improved k-means algorithm. In: 2022 Global Reliability and Prognostics and Health Management (PHM-Yantai). pp. 1–7 (2022). https://doi.org/10.1109/PHM-Yantai55411.2022.9942021
18. Lai, Y., Ping, G., Wu, Y., Lu, C., Ye, X.: Opensmax: Unknown domain generation algorithm detection. In: ECAI 2020, pp. 1850–1857. IOS Press (2020)
19. Lipton, Z.C., Berkowitz, J., Elkan, C.: A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:1506.00019 (2015)
20. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: 2008 eighth ieee international conference on data mining. pp. 413–422. IEEE (2008)
21. McInnes, L., Healy, J., Astels, S.: hdbscan: Hierarchical density based clustering. J. Open Source Softw. **2**(11), 205 (2017)
22. Milosevic, M.S., Ciric, V.M.: Extreme minority class detection in imbalanced data for network intrusion. Computers & Security **123**, 102940 (2022)
23. Moustafa, N., Slay, J.: Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In: 2015 military communications and information systems conference (MilCIS). pp. 1–6. IEEE (2015)

24. Panigrahi, R., Borah, S.: A detailed analysis of cicids2017 dataset for designing intrusion detection systems. International Journal of Engineering & Technology **7**(3.24), 479–482 (2018)
25. Papadimitriou, C.H., Steiglitz, K.: Combinatorial optimization: algorithms and complexity. Courier Corporation (1998)
26. Phung, D., Webb, G.I., Sammut, C.: Encyclopedia of Machine Learning and Data Science. Springer US (2020)
27. Ping, G., Feng, S., Li, Y., Ye, X.: Unsupervised anomalous traffic detection based on cascading representation and multiple-clustering. In: 2022 IEEE 8th International Conference on Computer and Communications (ICCC). pp. 2303–2307. IEEE (2022)
28. Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S.A., Binder, A., Müller, E., Kloft, M.: Deep one-class classification. In: International conference on machine learning. pp. 4393–4402. PMLR (2018)
29. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp **1**, 108–116 (2018)
30. Shen, L., Li, Z., Kwok, J.: Timeseries anomaly detection using temporal hierarchical one-class network. Advances in Neural Information Processing Systems **33**, 13016–13026 (2020)
31. Sun, Y., Dong, L., Huang, S., Ma, S., Xia, Y., Xue, J., Wang, J., Wei, F.: Retentive network: A successor to transformer for large language models. arXiv preprint arXiv:2307.08621 (2023)
32. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
33. Vitorino, J., Praça, I., Maia, E.: Sok: Realistic adversarial attacks and defenses for intelligent network intrusion detection. Computers & Security p. 103433 (2023)
34. Wang, W., Jian, S., Tan, Y., Wu, Q., Huang, C.: Robust unsupervised network intrusion detection with self-supervised masked context reconstruction. Computers & Security **128**, 103131 (2023)
35. Yang, J., Li, H., Shao, S., Zou, F., Wu, Y.: Fs-ids: A framework for intrusion detection based on few-shot learning. Computers & Security **122**, 102899 (2022)
36. Yang, Y., Xu, D., Nie, F., Yan, S., Zhuang, Y.: Image clustering using local discriminant models and global integration. IEEE Transactions on Image Processing **19**(10), 2761–2773 (2010)
37. Yang, Z., Liu, X., Li, T., Wu, D., Wang, J., Zhao, Y., Han, H.: A systematic literature review of methods and datasets for anomaly-based network intrusion detection. Computers & Security **116**, 102675 (2022)
38. Yin, Y., Jang-Jaccard, J., Sabrina, F., Kwak, J.: Improving multilayer-perceptron(mlp)-based network anomaly detection with birch clustering on cicids-2017 dataset. In: 2023 26th International Conference on Computer Supported Cooperative Work in Design (CSCWD). pp. 423–431 (2023). https://doi.org/10.1109/CSCWD57460.2023.10152640
39. Zhang, H., Zhang, X., Xie, J., Wang, Y.: Group abnormal behavior detection based on fuzzy clustering. In: 2020 3rd International Conference on Unmanned Systems (ICUS). pp. 245–250 (2020). https://doi.org/10.1109/ICUS50048.2020.9274820
40. Zong, B., Song, Q., Min, M.R., Cheng, W., Lumezanu, C., Cho, D., Chen, H.: Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: International conference on learning representations (2018)