Behavior Pattern Mining from Traffic and Its Application to Network Anomaly Detection

Yongwei Meng, Tao Qin, Shancang Li, and Pinghui Wang

1FT_he MOE Key Lab for Intelligent Networks and Network Security, Xi’an Jiaotong University, Xi’an, China
2FT_he Department of Computer Science and Creative Technologies, University of the West of England, Bristol, UK

Correspondence should be addressed to Yongwei Meng; ywmeng@stu.xjtu.edu.cn

Received 26 January 2022; Accepted 13 June 2022; Published 29 June 2022

1.Introduction

User behavior profiling, along with anomaly detection in network traffic, plays an important role in network management, which helps keep the network under control. User normal behavior patterns often present as stable and routine for long periods, but abnormal behaviors cause unexpected changes to the normal patterns. Therefore, behavior pattern changes can be used for anomaly detection. Features extracted from raw traffic packets are usually used to capture the dynamic changes in behavior patterns, such as total number of packets or flows in a specific time window, and then, machine learning methods are combined to mine abnormal changes [1–5]. These methods are effective for detecting obvious changes caused by abnormal behaviors. However, the attack technologies are becoming more and more intelligent, and anomalies only cause slight changes in the traffic patterns. Meanwhile, the traffic volume continues to increase, and characterizing user behavior and accurately identifying the anomalies from massive network traffic are still challenging tasks for network security monitoring.

Hindawi Security and Communication Networks
Volume 2022, Article ID 9139321, 17 pages
https://doi.org/10.1155/2022/9139321
On the other hand, how to identify the anomalies from the massive traffic patterns is a typical needle in a haystack (NIHA) problem. Matrix decomposition is an effective tool for anomaly detection [17, 18], which can divide the massive patterns into two parts: one is the major pattern in the original patterns and the other is corresponding to the abnormal changes, which is suitable for anomaly detection in the network today. Thus, we can use a matrix decomposition technique to distinguish normal and abnormal traffic.

To this end, we proposed an anomaly detection scheme in this study, which jointly employs the attribute matrix and the adjacency matrix to characterize user behavior patterns and network environment characteristics and uses matrix decomposition to identify anomalies. We extract seven statistical features from network traffic to construct an attribute matrix to capture the user behavior characteristics in the time window $\Delta T$. We also employ the similarity of the specific IP addresses to construct an adjacency matrix to capture the behavior characteristics related to the network environment. We jointly employ the attribute matrix and the adjacency matrix to construct a model to describe user behavior characteristics accurately. Then, we employ CUR matrix decomposition [19] to mine the major behavior patterns from the joint model and obtain a residual matrix, which can be used to identify anomalies.

We employ two kinds of traffic datasets to verify the effectiveness and accuracy of our method. The first is the public MAWI dataset [20], which is collected from the WIDE backbone network, a trans-Pacific transit link between Japan and the USA. This dataset is labeled and used to evaluate the performance of our proposed method. The second dataset is collected from the northwest center of CERNET. The users in the monitoring network include students, faculty members, and contract personnel from service-providing companies. The behavior patterns contained are sufficiently complex and can be employed to measure the practicability of our method. The experimental results based on the two datasets show that the proposed method achieves an anomaly detection accuracy rate higher than 90% without any prior knowledge. Furthermore, our method can also trace anomalies for efficient network management.

Our contributions to this study can be summarized as follows:

1. **We Propose an Anomaly Detection Scheme Including Both the Features Extracted from Traffic Volumes and Network Environment:** We employ seven traffic features to capture the dynamic changes in the traffic volumes and employ similarity between IP addresses to construct an adjacency matrix to characterize the user behavior related to the network environment.

2. **We Formulate the Anomaly Detection Problem as a Matrix Decomposition Problem:** We formulate the anomaly detection problem as a matrix decomposition problem and employ CUR matrix decomposition to perform anomaly detection. The experimental results verify that the developed method is a simple and effective means to monitor the security of an enterprise network.

3. **We Verify the Effectiveness and Accuracy of Our Methods on Two Different Datasets:** The first is the public dataset MAWI, and the second is the dataset collected from the northwest center of CERNET. The MAWI dataset is used to validate the accuracy of the proposed scheme, while the CERNET dataset is employed to verify the practicability.

The remainder of this study is organized as follows. Section 2 presents the related work; Section 3 outlines the motivations and design goals, after which the feature definitions and framework description are presented in Section 4. In Section 5, we put forward a detailed description of the anomaly detection model. The experimental results and analysis are presented in Sections 6 and 7, after which the conclusion follows in Section 8.

## 2. Related Work

The goal of anomaly detection is to find the rare occurrences that do not conform to the patterns of the majority in the datasets [21, 22], which has been widely applied in many fields, including security, finance, health care, and social network [23–27]. A variety of techniques have been proposed for identifying anomalies, which can be presented from two aspects: supervised and unsupervised anomaly detection methods [28]. The works related to our work are summarized as follows.

As for supervised anomaly detection techniques, they usually require a labeled dataset for model training. A support vector machine (SVM) can classify the samples as normal and anomalies by maximizing the classification margin to detect anomalies. Kong et al. [29] designed an abnormal traffic identification system (ATIS) based on SVM. Gu et al. [30] proposed an intrusion detection framework based on an SVM ensemble classifier with increasing feature selection. The naive Bayesian is another simple and effective tool to detect anomalies, and many algorithms have been proposed based on Bayes’ theorem. Swarnkar et al. [31] proposed a naive Bayesian class classifier based on packet payload analysis to detect HTTP attacks. Han et al. [32] developed a naive Bayesian model for network intrusion detection based on principal component analysis (PCA). Nie et al. [33] designed a Bayesian network to model the causal relationships between network entries. Neural networks (NNs) are also widely used for anomaly detection as they can increase the accuracy of anomaly detection systems. Hodo et al. [34] employed packet traces to train an artificial neural network to detect DDoS attacks. Kwon et al. [35] used a convolutional neural network (CNN) to detect anomalies, which can select traffic features automatically from the raw dataset. A recurrent neural network (RNN) was proposed in [36] to learn temporal behaviors in large-scale network traffic data. Those methods are effective in identifying anomalies with accurately labeled datasets; however, high-quality labeled datasets are very difficult to construct in the network today.
Unsupervised anomaly detection techniques are widely used recently, as they do not require labeled dataset for model training. K-means is one basic approach to the unsupervised anomaly detection [37]. The authors in [38] used K-means to cluster the network connections into normal and anomalous communities. However, it is difficult to select the suitable k, as it depends on the applications and environments. Recently, Chen et al. [39] proposed a convolutional autoencoder (CAE)-based anomaly detection model. Said Elsayed et al. [40] proposed a hyper-approach based on long short-term memory (LSTM) autoencoder and one-class support vector machine (OC-SVM) to detect anomalies. Although the methods hold high accuracy, it is difficult to obtain a clear explanation of the results; furthermore, it is hard to trace the anomalies and apply control policy. Principal component analysis (PCA) is another unsupervised method used for anomaly detection, which can capture the normal and abnormal behaviors of the data by projecting the data instances to the principal components [41]. Wang and Battiti [42] proposed an intrusion detection method combined PCA with SVD, which can identify intrusions based on the error between the original data vector and its reconstruction data vector. However, it is not efficient for interpretation as the principal components are a linear combination of all original variables [17]. To interpret these results, the work in [43] introduced a new method, named sparse principal component analysis (SPCA), to produce modified principal components with sparse loading. Although this method can improve the interpretation, a linear relationship still exists between the principal components and the original variable. However, the variables usually do not hold the linear relationship. The sample-based matrix decomposition methods are proposed to deal with those problems, which select rows or columns from the original matrix to form the low-rank matrices. Kumar et al. [44] proposed CUR matrix decomposition to interpret the decomposition process. However, the decomposition process will occupy a large amount of memory. Sun et al. [45] proposed a new method named compact matrix decomposition (CMD), which can avoid repeated selection, in turn, reduces the computational complexity. However, this method needs to seek a non-orthogonal base by sampling the columns and/or rows of the original matrix, which will produce over-complete bases. Tong et al. [46] proposed a Colibri method to deal with these challenges. This method can iteratively find a nonredundant base and accordingly save space and time cost. However, it fails to improve the accuracy compared with the CUR and CMD matrix decomposition.

Inspired by the related works, we propose an anomaly detection method based on matrix decomposition. By combining the advantages of CUR and network characteristics, the developed methods can not only detect known and unknown anomalies, but also trace the source of the anomalies.

3. Basic Assumption and Design Goals

3.1. Basic Assumption and Its Verification. To capture the characteristics of network environment, we assume that the users who hold IP addresses with the same prefix have similar behavior patterns, and we verify this assumption from the following three aspects.

Firstly, we analyze the general principle of the IP address arrangement. Generally, IP addresses have no relationship with user behavior patterns; however, to manage conveniently, network administrators usually assign IP addresses with the same prefix to the users in one specific area. The IP address arrangement process can be summarized as follows: (1) the Internet Assigned Numbers Authority (IANA) assigns IP address pools to five different regional Internet registry (RIR) organizations in the world. (2) The regional organization assigns the IP addresses to different Internet service providers (ISPs). (3) The IP address blocks are assigned to different countries by the ISP. (4) Network administrators assign IP address blocks to different areas when they construct their local area network (LAN). Based on the above analysis, we can find that IP addresses with the same prefix are often assigned to the same area. Furthermore, the users in the same area often have similar behavior patterns as they have similar network requirements. Thus, we can conclude that users who hold IP addresses with the same prefix may hold similar behavior patterns.

Secondly, there are also some researchers working on traffic pattern profiling found that the users who hold IP addresses with the same prefix have similar behavior patterns. Jiang found that traffic behavior with the same prefix often keeps stable over time, which can be used for anomaly detection [47]. Xu found that the hosts with the same network prefixes have similar behavior among different Internet applications [48, 49]. Jiang found that the behavior similarity captured by the aggregated flows with the same network prefixes can be used to construct an abnormal identification mechanism [50]. Those works further verify the assumption.

Thirdly, we analyze the behavior patterns of the IP addresses with the same p prefixes in the MAWI and CERNET datasets. We randomly select three IP blocks, and the results are shown in Figure 1, where a-c are the results of the MAWI dataset, d-f are that of the CERNET dataset, a and d are the results of $P = 8$, b and e are the results of $P = 16$, and c and f are the results of $P = 24$. From the figure, we can find that users who hold IP addresses with the same prefix have similar behavior patterns, especially the behavior patterns in the CERNET dataset.

3.2. Design Goals. Based on the above analysis results, we mainly focus on developing a new anomaly detection method, which is effective at mining the anomalies in the network today, and the design goals are listed as follows:

1. Improve the Management Efficiency: to control the anomalies, tracing the anomalies is important and necessary. The IP address should be retained during the detection process. In our developed method, we regard each specific IP address as an index of the column to construct the attribute matrix, which can achieve the goal of abnormal IP addresses tracing easily.
(2) **Improve the Detection Accuracy:** to develop an accurate abnormal detection model with the consideration of network environment, we construct an adjacency matrix to describe the network environment. The adjacency matrix is made up of IP address of similar degree, which is calculated by the binary similarity of the IP addresses.

(3) **Improve the Practicality:** the designed method should be deployed on most enterprise networks without new hardware components, and the features used should be easily extracted. Furthermore, the method should be sensitive to special abnormal behaviors that can cause slight changes so as to detect the new anomalies.

4. **Feature Definition and Framework Description**

4.1. **Volume Feature Definition.** Firstly, to apply some measurement methods to traffic adopted encryption technology, more and more anomaly detection methods extract
features from the packet headers. The information in the packet headers is shown in Table 1.

Secondly, we analyze the characteristics of typical attacks, and the results are shown in Table 2. From the table, we can find that different attacks may lead to obvious changes to the statistics of the attributes in the packet headers, in turn, the extracted features will be changed, and those changes can be used to detect anomalies.

Finally, we can find that different network attack behaviors will lead to changes in the different attributes based on the analysis above. Their definitions are shown in Table 3. We employ the port scan attack as an example to verify the efficiency of the features extracted. Port scan attack is usually employed by hackers to find vulnerable hosts and ports, and one simple example of port scan is shown in Figure 2. The attack host will send connection requests to the ports of one same destination host, and if the destination port is open and the host provides the corresponding service, the hackers can receive responses. This kind of behavior pattern is typical and representative of the smart attacks, and this kind of pattern of can be easily captured by OD, NDDA, and NDDP defined above.

4.2. Network Environment Feature Definition. We employ the adjacency matrix composed of similarity between the IP addresses of the monitoring network to capture the characteristics of the network environment. An example of the similarity calculation between two IP addresses is presented in Table 4. The distance $d$ is defined as the number of 1 in $m$, which can be calculated by applying XOR to the binary of two given IP addresses. Then, we select the first $i$ as label from left to right of the operation results and set all bits behind the label to 0, and other bits are set to 1. The similarity $s$ is defined as the distance $d$ divided by 32, and the maximum value of the similarity is 1, which means that the two IP addresses are identical, while the minimum value is 0, which denotes that the two IP addresses are completely different. If the similarity is bigger than a specific threshold, the value of the two IP addresses in the adjacency matrix is set 1 and the construction method is shown as follows:

$$A(i, j) = \begin{cases} 1, & s \geq \text{thresh}, \\ 0, & \text{otherwise}, \end{cases}$$

where $i$ and $j$ represent any IP address and thresh denotes the threshold predefined.

We employ a campus network to illustrate the features extracted to characterize the network environment. One simple topology of a typical campus network is shown in Figure 3. The users in the same area often have similar behavior characteristics as they have similar roles, such as the users in the office area often engage in teaching-related activities as they are teachers, while the users in the dormitory area often engage in entertainment-related activities after classes as they are students.

4.3. Anomaly Detection Framework Developed. The detailed description of the scheme is illustrated in Figure 4, which consists of four steps. The main symbols used in the study are summarized in Table 5.

Step 1. Traffic Matrix Construction. We employ IP addresses and their features to design an attribute matrix to describe the traffic patterns in time window $\Delta T$. Let $V = \{s_1 p_1, s_1 p_2, \ldots, s_n p_n\}$ denote a set of $n$ IP addresses, and $F = \{f_1, f_2, \ldots, f_d\}$ denote the $d$-dimensional features per IP address. At the same time, we also construct an adjacency matrix using IP address similarity as a constraint to capture the pattern relationship between IP addresses. We combine the attribute matrix and adjacency matrix to design an anomaly detection model.

Step 2. Matrix Decomposition. We employ CUR matrix decomposition to select several representative features and IP addresses from the attribute matrix $X$ to reconstruct an attribute matrix $\bar{X}$ and attempt to keep the reconstruction matrix as similar as possible to the attribute matrix $X$.

Step 3. Residual Calculation. The difference between the attribute matrix $X$ and the reconstruction matrix $\bar{X}$ is referred to as the residual matrix, which is defined as $\bar{R} = X - \bar{X}$.

Step 4. Anomaly Detection. The residual matrix $\bar{R}$ presents the pattern changes in each feature of the IP addresses. We calculate the sum of each column of the residual matrix, which represents the total pattern changes in each IP address, and then, they are ranked in descending order. IP addresses with larger values are regarded as anomalies.

5. Anomaly Detection Model

5.1. Model for Attribute Matrix Decomposition. The seven defined traffic features can construct an attribute matrix $X$, as the following formula shows, where the columns are the specific IP addresses, the rows are the seven features, and the element is the statistical values of specific features in time window $\Delta T$.

$$X = \begin{bmatrix} X(1, 1) & X(1, 2) & \cdots & X(1, n) \\ X(2, 1) & X(2, 2) & \cdots & X(2, n) \\ \cdots & \cdots & \cdots \end{bmatrix}. \quad (2)$$

We employ CUR matrix decomposition to obtain user normal behavior patterns from the original attribute matrix $X$, which can select several rows and columns from $X$ to

| Number | Attributes | Data type | Meanings |
|--------|------------|-----------|----------|
| 1      | srcIP      | String    | Source IP address |
| 2      | destIP     | String    | Destination IP address |
| 3      | srcPort    | Int       | Source port |
| 4      | destPort   | Int       | Destination Port |
| 5      | pktNum     | Long      | Number of packets |
| 6      | pktSize    | Long      | Packet size |
| 7      | protocol   | Int       | Protocol type |

Table 1: Information in packet headers.
Table 2: Analysis of typical attacks.

| Attack types | Basic description | Related attributes | Proposed features |
|--------------|-------------------|--------------------|-------------------|
| DoS/DDoS     | Many hosts or a host send a large number of network attack packets to a destination host | srcIP, pktNum, pktSize | OD, NDDA, NDR, NP, LT |
| Port scan    | An attack host sends massive scan packets to several ports of a destination host | destPort | OD, NDDP, NDDA |
| Network scan | An attack host sends a lot of scan packets to multiple destination hosts | srcIP | NDDA, NP, SNF |
| Worm         | An attack host will scan vulnerability ports of a destination host | destIP, destPort | OD, NDDP |
| Flash crowd  | Many hosts send a great number of connection requests to a destination host in a short time | srcIP, pktNum, pktSize | OD, NDDA, NDR, NP, LT |
| Alpha flows  | A host sends massive network flows to a destination host | srcIP, destIP, destPort | OD, NDR, NP |

Table 3: Feature definitions of network traffic.

| Notations | Definitions |
|-----------|-------------|
| OD        | Out degree of specific source IP address: the number of flows a specific source IP address holds in a time window $\Delta T$ |
| NDDA      | Number of different destination IP of specific source IP address: the number of different destination IP that connects a specific source IP in a time window $\Delta T$ |
| NDR       | Number of different responses of specific source IP: the number of different flows when a specific source IP is regarded as the destination IP in a time window $\Delta T$ |
| NDDP      | Number of different destination ports of specific IP address: the number of different destination ports held by a specific IP address in a time window $\Delta T$ |
| NP        | Number of packets of the specific source IP address: the average number of the packets held by a specific source IP address in a time window $\Delta T$ |
| SNF       | Size of the network flow of the specific source IP address: the average size of the network flow held by a specific source IP address in a time window $\Delta T$ |
| LT        | Life time of specific source IP address: the average life time of a specific source IP address in a time window $\Delta T$ |

Figure 2: Block port scan.

Table 4: An example of similarity calculation.

| IP      | Decimal   | Binary                      | Results |
|---------|-----------|-----------------------------|---------|
| $i p_1$ | 202.117.112.60 | 11001010.01110101.01110000.00111100 |         |
| $i p_2$ | 202.117.112.176 | 11001010.01110101.01110000.10110000 |         |
| $m$     | 255.255.255.0  | 11111111.11111111.11111111.11111111.00000000 |         |
| $d$     | 65535      |                             | 24      |
| $s$     |           |                             | 0.75    |
reconstruct an attribute matrix, and the reconstructed matrix can indicate the major pattern of the original attribute matrix. The process can be formulated as follows:

\[
\min_{W,R} \|X - WX - R\|_F^2 + \alpha \|W\|_{2,1} + \beta \|W^T\|_{2,1} + \gamma \|R\|_{2,1},
\]

where \( W = \text{CUR} \in \mathbb{R}^{n \times d}, \ C \in \mathbb{R}^{n \times n}, \ U \in \mathbb{R}^{n \times d}, \) and \( R \in \mathbb{R}^{d \times d} \) are three low-rank matrices, \( R \) indicates the residual matrix, \( \alpha \) and \( \beta \) control the row and column sparsity of the matrix \( W \), and \( \gamma \) is used to control that of matrix \( R \). If \( C \) is a diagonal matrix and its \( k \) elements are 1, the other elements are 0, and \( XC \) keeps \( k \) columns of \( X \) unchanged and sets \( n-k \) columns to zero vectors. Similarly, \( WX \) can be regarded as a coefficient matrix, and \( X(:,i) \) is chosen as a representative source IP address when \( W \) of \( WX \) is not a zero vector. Apparently, \( \|W\|_{2,1} \) ensures that only a few source IP addresses are chosen.

5.2. Model for Adjacency Matrix Decomposition. We use adjacency matrix \( A \) to capture the characteristics of network environment, and one simple example is shown as follows:
where the rows and columns are the specific IP addresses. If two users hold IP addresses with high similarity, they should have similar behavior patterns in the residual matrix $\bar{R}$. The matrix decomposition model constructed between the residual matrix and adjacency matrix is shown as follows:

$$\frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{n} \left( \bar{R}^T(i,:) - \bar{R}^T(j,:) \right)^2 A(i,j) = tr(\bar{R}LR^T),$$

(5)

where $A$ is the adjacency matrix and $L$ is the Laplacian matrix of it.

### 5.3. Anomaly Detection Model Construction

#### 5.3.1. Anomaly Detection Model

Based on the attribute matrix and adjacency matrix constructed, we develop an anomaly detection model, which is shown as follows:

$$\min_{W, R} \|X - XWX - \bar{R}\|^2_F + \alpha \|W\|_{2,1} + \beta \|W^T\|_{2,1}$$

$$+ \gamma \|R^T\|_{2,1} + \delta \text{tr}(\text{tr}(\bar{R}LR^T)),$$

(6)

where $\delta$ is a parameter to indicate the importance of the adjacency matrix. Formula (6) may be non-convex if $W$ and $R$ change simultaneously, but we can get the optimal values of them by fixing one of them. If we fix $W$, we can rewrite formula (6) to (7) by setting the derivative of $\bar{R}$ to zero.

$$\bar{R} = (X - XWX)(I + \gamma D_R + \delta L)^{-1},$$

(7)

where $I$, $\gamma D_R$, and $\delta L$ are diagonal matrices with positive diagonal entries, and $D_R$ is a diagonal matrix with the $i$th elements $D_R(i,i) = 1/2[\bar{R}(i,:)]_2$. Similarly, if we fix $R$ and set the derivative of $W$ to zero, we can obtain the following equation:

$$X^T X W X X^T + a D^T W + b D^T W = X^T X X^T - X^T R X^T,$$

(8)

where $D^T W$ and $D^T W$ are diagonal matrices with $D^T W(i,i) = 1/2[\bar{W}(i,:)]_2$ and $D^T W(i,i) = 1/2[\bar{W}^T(i,:)]_2$, respectively. We can use the two lemmas below to calculate $W$.

**Lemma 1.** For any matrix $A \in \mathcal{O}_{mxn}$, if the matrix is right multiplied by a diagonal matrix $B \in \mathcal{O}_{nxm}$, the expression $AB$ can be rewritten as $B^T \otimes A$, where $B^T(i,j) = B(j,i), \forall i = 1,2, \ldots, m, \forall j = 1,2, \ldots, n$, and $\otimes$ denotes the Hadamard product.

**Lemma 2.** For any formula as (9) about matrix $H$:

$$C_1 H C_2 + a D_1 H + b D_2 H = V,$$

(9)

where $C_1$ and $C_2$ are symmetric and positive semidefinite matrices, $D_1$ and $D_2$ are two diagonal matrices, and $a$ and $b$ are two parameters. Formula (9) can be rewritten as follows:

$$\Theta_1 K \Theta_2 + a C^T D_1 C K + b D_2 \otimes K = C^T V R^T,$$

(10)

where $C_1 = C \Theta_1 C^T$, $C_2 = R^T \Theta_2 R$, $K = C^T H R^T$, and $C$ and $R$ are two orthogonal matrices, $\Theta_1$ and $\Theta_2$ are two diagonal matrices composed of eigenvalues. According to the two lemmas, we pre-multiply formula (8) by $C^T$ and post-multiply it by $R^T$.

Equation (8) can be reformulated as follows:

$$\Theta_1 U \Theta_2 + a Z U + b D^T W (1, i) U = M,$$

(11)

where $X^T X = C \Theta_1 C^T$, $X X^T = R^T \Theta_2 R$, $Z = C^T D^T W$, $U = C^T W R^T$, $M = C^T (X^T X X^T - X^T R X^T) R^T$, and $C$ and $R$ are orthogonal matrices composed of the eigenvectors of $X^T X$ and $XX^T$, respectively. $\Theta_1$ and $\Theta_2$ are two diagonal matrices composed of the eigenvalues of $X^T X$ and $XX^T$, respectively. Then, we define $U = [u_1, u_2, \ldots, u_d] \in \mathcal{O}_{nd}$, $M = [m_1, m_2, \ldots, m_d] \in \mathcal{O}_{nd}$, and $Z = [z_1, z_2, \ldots, z_d] \in \mathcal{O}_{nd}$. According to Lemma 1, formula (11) can be expressed as follows:

$$\Theta_1 (i,i) \Theta_2 u_i + a Z u_i = b D^T W (1, i) U = m_i.$$

(12)

For each column vector, it can be denoted as follows:

$$u_i = (\Theta_1 (i,i) \Theta_2 + a Z + b D^T W (1, i))^{-1} m_i.$$

(13)

We can get the matrix $U$ by calculating each column vector $u_i$, and the matrix $W$ can be obtained according to the formula $U = C^T W R^T$.

$$W = CUR.$$

(14)

#### 5.3.2. Running Flowcharts

The detailed running process of the developed model is illustrated in algorithm 1. Firstly, the attribute matrix $X \in \mathbb{R}^{n \times n}$ and the adjacency matrix $A \in \mathbb{R}^{m \times n}$ are constructed and selected as input. After setting other parameters, the algorithm iteratively selects some representative IP addresses and features by means of the variable matrix $W$ to reconstruct the attribute matrix $X$. Lines 1 to 4 build the Laplacian matrix $L$ and initialize the identity matrices $D_R$, $D_W$, and $D_W$, the residual matrix $\bar{R}$, orthogonal matrices $C$ and $R$, and diagonal matrices $\Theta_1$ and $\Theta_2$. As shown in lines 5 to 11, the optimal residual matrix $\bar{R}$ can be obtained when the objective function in equation (6) converges by iteratively updating $W$ and $\bar{R}$. Line 12 calculates the anomaly scores for each IP address by means of its $\ell_2$ norm in the residual matrix $\bar{R}$. Lines 13 to 17 judge whether the IP addresses are anomalies or not.

### 6. Performance Evaluation with Public Data

#### 6.1. Dataset and Evaluation Metrics

#### 6.1.1. MAWI Dataset

It is important to find a public dataset with reliable ground truth to evaluate anomaly detection methods. However, some existing datasets are outdated due to lack of new attack trends or traffic patterns, such as the
Input: attribute matrix $X$, adjacency matrix $A$, parameters $\alpha, \beta, \gamma, \delta, \theta$

Output: top $k$ source IP addresses satisfying the condition $\theta$

(1) Build Laplacian matrix $L$ from the adjacency matrix $A$;
(2) Initialize $D_w, D'_w$ and $D''_w$ to be the identity matrix;
(3) Initialize $R = X(I + \gamma D_w + \delta I)^{-1}$;
(4) Build orthogonal matrices $C, \Theta$ by the eigenvectors of $XX^T$ and $XX^T$, and diagonal matrices $\Theta_1, \Theta_2$ through the eigenvectors of $XX^T$ and $XX^T$, respectively;
(5) while the objective function in equation (6) does not converge do
(6) Update $W$ by equation (14);
(7) Update $D''_w$ by setting $D''_w(i, i) = 1/2[WW^T(., i)]_2$;
(8) Update $D'_w$ by setting $D'_w(i, i) = 1/2[W^T(., i)]_2$;
(9) Update $R$ by equation (7);
(10) Update $D_w$ by setting $D_w(i, i) = 1/2[R^T(., i)]_2$;
(11) end
(12) Compute the anomaly score for the $i$th source IP address as $||R(., i)||_2$;
(13) if the score of the $i$th source IP address $> \theta$ then
(14) Output the source IP address;
(15) end
(16) end
(17) end

Algorithm 1: Residual analysis for anomaly detection on network traffic.

1998/99 DARPA dataset [51] and KDD CUP 99 dataset [52]. Other datasets, such as DDoS 2016 [53], are created in a simulated network environment, and LBNL dataset [54] is publicly available but does not provide attack labels. Although UNB ISCX dataset [55] provides attack labels, it is not always publicly available. Due to the drawbacks of the datasets, we use the MAWI dataset to validate the performance of our proposed approach. The dataset is collected from the WIDE backbone network, a trans-Pacific transit link between Japan and the USA, and it is updated daily to include new patterns of new applications or anomalies, and the payloads of the packets are removed due to privacy protection issues. Furthermore, a graph-based methodology that combines different anomaly detectors is used to label the dataset [20]. Firstly, it uses a similarity estimator to uncover the relations between the outputs of different anomaly detectors, then construct an undirected graph using the anomalies detected by different anomaly detectors, and mine different communities from the undirected graph. Secondly, they employ the confidence score and combination strategies to decide whether one specific community corresponds to an anomalous or not.

We employ the dataset collected from 2019-05-05 14:00:00 to 2019-05-05 14:15:00 to evaluate the proposed method. The statistical information about the dataset is presented in Table 6, where #anomalies represents the number of anomalies, #flows is the total number of flows, #diffsrcIP denotes the number of different source IP addresses, #diffsrcPort is the number of different source ports, #diffdstIP denotes the number of different destination IP addresses, and #diffdstPort is the number of different destination ports. GRE denotes generic routing encapsulation protocol, while encapsulating security payload (ESP) is a typical protocol in IPsec architecture.

We analyze the percentage of different anomalies in the datasets, and the results are shown in Figure 5. From the figure, we can find that the top 4 anomalies occupy more than 90% in the MAWI dataset, including ntscUDPUDPPrp (network_scan_udp_udp_response), mptp (multipoint_to_point), mptmp (multipoint_to_multipoint), and ptmp (point_to_multipoint). Furthermore, those anomalies are typical and representative, such as mptmp that usually employs many controlled hosts to attack several destination hosts coordinately, which is similar to Bonet and Worm attacks. Thus, we mainly employ the anomalies with ntscUDPUDPPrp, mptp, mptmp, and ptmp labels to evaluate our approach.

6.1.2. Evaluation Metrics. We use precision ($P$), recall ($R$), and $F_1$, which are widely used to evaluate accuracy of many approaches [56]. Their definitions are provided as follows.

$P$ is the ratio between the number of true anomalies detected and total number of anomalies detected, which is shown in the following formula, where $TP$ is the number of true anomalies detected and $PA$ is the total number of anomalies detected.

$$P = \frac{TP}{PA} \times 100\%,$$  \hspace{1cm} (15)

where $R$ is the ratio between the number of true anomalies detected and the total number of anomalies in the dataset. Its definition is presented in the following formula, where $TA$ represents the total number of anomalies in the dataset.

$$R = \frac{TP}{TA} \times 100\%,$$  \hspace{1cm} (16)

where $F_1$ is the harmonic mean between $P$ and $R$, which is defined as follows:
6.2. Parameter Selection. There are six parameters in the proposed algorithm, the parameter \( s \) presents the similarity of two IP addresses, and we employ this parameter to ensure that two IP addresses are similar enough; thus, the similarity threshold \( \text{thresh} \) for the adjacency matrix is 0.9. The parameters \( \alpha, \beta, \) and \( \gamma \) are used for controlling the sparsity of the attribute matrix and residual matrix, and the parameter \( \delta \) is used for controlling the importance of the adjacency matrix, and their values are important for the model performance. We firstly keep one of the parameters increasing and other three parameters fixed and then employ the changing trends of \( F_1 \) to select the parameter values. The results are shown in Figure 6. We can find that \( F_1 \) increases rapidly when parameter \( \delta \) is becoming larger and then decreases quickly and tends to be stable, which means that the IP address structure plays an important role in anomaly detection. For the parameters \( \alpha, \beta, \) and \( \gamma, \) \( F_1 \) increases rapidly with the parameters increasing and then tends to stable. Therefore, we set the initial values of the four parameters in the algorithm according to the changing trends of \( F_1, \alpha = 0.017, \beta = 0.015, \gamma = 0.012, \) and \( \delta = 0.018. \)

\[
F_1 = \frac{2 \times P \times R}{P + R} \tag{17}
\]

The parameter \( \theta \) has more impacts on the final accuracy. To select optimal \( \theta \) for different time windows, we calculate the \( P, R, \) and \( F_1 \) with different \( \theta. \) For the two evaluation metrics \( P \) and \( R, \) they are existing constrained relation, and we employ \( F_1 \) to make some trade-offs. The results of specific time window are presented in Table 7. From the table, we can find that \( R \) is larger than \( P \) when the threshold \( \theta \) is smaller than 0.005. Contrarily, \( R \) is smaller than \( P \) when the threshold \( \theta \) is larger than 0.005. Therefore, the optimal threshold \( \theta \) is selected as 0.005 with the largest \( F_1 \). We can obtain the optimal thresholds for each time window based on the above analysis, which is shown in Figure 7.

To achieve the goal of optimal parameter \( \theta \) automatic selection, we use the exponentially weighted moving average (EWMA) [57] method to predict optimal parameter \( \theta. \) Assume \( \theta_{t-1} \) and \( \tilde{\theta}_{t-1} \) are the optimal \( \theta \) and the forecast \( \theta \) for the time window \( t - 1 \), respectively. The EWMA method can be given as follows:

\[
\tilde{\theta}_t = b\theta_{t-1} + (1 - b)\tilde{\theta}_{t-1}, \tag{18}
\]

where \( 0 \leq b \leq 1 \) is the weight factor. We employ the mean squared error (MSE) to evaluate the effectiveness of the prediction for optimal thresholds, and the definition of the MSE is shown as follows:

\[
MSE = \frac{1}{m} \sum_{i=1}^{m} (\theta_i - \tilde{\theta}_i)^2, \tag{19}
\]

where the parameter \( m \) is the number of windows, and \( \theta_i \) and \( \tilde{\theta}_i \) are the optimal threshold and the prediction threshold, respectively. We employ the manually selected optimal \( \theta \) (results of the beginning fifteen time points) to obtain the weight factor \( b \) used for EWMA, and then, the EWMA is used to predict the threshold \( \theta \) used for the following time points. The analysis results are shown in Figure 7.
Table 7: P, R, and F1 with different θ of specific time window.

| θ     | P (%) | R (%) | F1   |
|-------|-------|-------|------|
| 0.0010| 0.864 | 0.978 | 0.921|
| 0.0037| 0.883 | 0.974 | 0.926|
| 0.0045| 0.914 | 0.966 | 0.940|
| 0.0050| 0.921 | 0.964 | 0.943|
| 0.0067| 0.918 | 0.933 | 0.925|
| 0.0070| 0.914 | 0.889 | 0.901|
| 0.0072| 0.759 | 0.212 | 0.332|
| 0.0077| 0.717 | 0.171 | 0.276|
| 0.0083| 0.587 | 0.096 | 0.165|
| 0.0161| 0.500 | 0.065 | 0.115|

Figure 7: Optimal and prediction thresholds for different time windows.

6.3. Performance Evaluation. Network traffic is a kind of time-series data, the traffic volume usually is massive, and it is very difficult to process in real time. To improve the practicality, we employ time window mechanism and the size of the time window is set at 6 seconds in the provided figure. We have 150 time windows for the MAWI dataset and analyze the anomalies in each time window using different anomaly detection approaches. For matrix decomposition methods, we use the proposed method and SVD and SPCA approaches to assign anomaly value for traffic in each time window and employ the threshold θ to identify anomalies; if the anomaly value is larger than the threshold θ, the corresponding network traffic is regarded as anomalies. Then, we can calculate their P, R, and F1 in each time window as the dataset is labeled. Similarly, we also apply the LOF, CBLOF, ROS, and COF approaches to assign anomaly value for traffic in each time window, and we can obtain the F1 values of these methods in different time windows.

6.3.1. Comparison with Matrix Decomposition Methods. We select two popular abnormal detection methods using matrix decomposition to evaluate our methods. Wang employed singular value decomposition (SVD) to compute the eigenvalues and their corresponding eigenvectors of the covariance matrix [42] and then select the k eigenvectors with the top k largest eigenvalues to form a new matrix. For a new vector, SVD first projects it to the k-dimensional subspace and then calculates the distance between the new data vector and its reconstruction using squared Euclidean distance. If the distance exceeds a given threshold, the vector is identified as anomalous.

6.3.2. Comparison with Other Data Mining Methods. Tuan proposed an anomaly detection method based on the local outlier factor (LOF) [58]. The LOF is defined as the ratio between the local reachability densities of object p’s k-nearest neighbors and the local reachability density of p. A larger LOF means that the local reachability density of p is smaller than the local reachability densities of p’s k-nearest neighbors, and the object p has a higher probability to be an anomaly. In this study, we select k as the square root of all the data to perform abnormal detection, which has been proven to be an optimal option [59].

Erichson proposed the SPCA approach to improve the interpretation of low-rank matrix decomposition [43]. The developed methods can be formulated as a regression-type optimization problem of the PCA method. Sparse loadings are obtained by imposing the lasso constraint or elastic net on the regression coefficients. The modified principal components are therefore further obtained based on sparse loading. In this study, we apply the SPCA to the attribute matrix and obtain a new matrix composed of principal components. Then, the distance between the features extracted in the coming time window and its reconstruction is calculated and abnormal detection is performed.

The experimental results are presented in Figure 8. As the figure shows, our method outperforms the other two methods. The attribute matrix is used to capture the dynamics of the traffic patterns, while the adjacency matrix is used to characterize the environment. Thus, our method holds higher accuracy by including more consideration during the process of matrix decomposition. Furthermore, the SVD method lacks interpretability as the eigenvectors are usually a linear combination of all columns of the original matrix. As the SPCA method, although it improves the interpretability based on sparse principal components, it cannot interpret the results completely. The proposed method can easily interpret the detection process by selecting representative rows and columns, in turn, tracing the anomalies for efficient network management.
Tang proposed an outlier detection scheme by calculating the chaining distance \[61\]. They use the connectivity-based outlier factor (COF) to indicate the probability of a being detected object to be an anomaly. The probability of a being detected object \(p\) to be an anomaly is defined as the ratio between the chaining distance of \(p\)'s \(k\)-nearest neighbors and the average of the chaining distance of its neighbor’s \(k\)-distance neighbors.

Pei proposed a reference-based outlier detection approach, which is capable of reducing the number of distance calculations compared with the LOF method \[62\]. They first calculate the distance between each reference point and the data points, then find the \(k\) reference-based nearest neighbors for each data point, and compute their average neighborhood density. Then, the minimum of the neighborhood density of a data point is used to define the reference-based outlier score (ROS). Data points with higher ROS are considered to be anomalies. In this study, we select the parameter \(k\) similar to LOF and select the reference points using the grid vertices \[62\].

The experimental results are shown in Figure 9. As the figure shows, \(F_1\) of the proposed method is larger than those of the LOF, CBLOF, ROS, and COF. The four approaches only employ user behavior characteristics to mine anomalies, while the proposed method considers user behavior and network environment simultaneously; thus, the proposed method is more practicable. Furthermore, compared with them, the matrix decomposition not only can mine the abrupt changes caused by the abnormal behavior patterns but also that of slight changes.

6.4. Analysis of the Time Complexity. The time complexity of the proposed scheme mainly consists of two parts: feature matrix establishment and matrix decomposition. There are two matrices used to profile the traffic patterns, including the attribute matrix and the adjacency matrix. If we do not use any data structure to optimize the establishment process, both of the establishment time complexity of them are \(O(n^2)\), where \(n\) is the total number of unique IP addresses in the monitoring network. If we use the hash method to optimize the established process, the time complexity of the attribute matrix establishment can be reduced to \(O(n)\). As for the adjacency matrix, as the IP addresses of the monitored network are fixed, and we can obtain the IP similarity by offline calculation. The most time-consuming operation involved in the proposed algorithm is the matrix inverse operation, whose computational complexity is \(O(n^2d)\) when updating the residual matrix \(\mathbf{R}\) at each iteration. Moreover, the computational complexity of updating \(\mathbf{W}\) is \(O(n^2)\). The total time complexity of the developed method is \(m \times O(n^2)\), where \(m\) is the number of iterations, \(n\) is the number of unique IP addresses in the monitoring network, and \(d\) is the dimension of attribute extracted. We can find that time complexity of the proposed method is irrelevant to the traffic volumes of the monitoring network, and only the increment of the number of IP addresses in the monitoring network can increase the time complexity. However, for an enterprise network, the total number of IP addresses is usually fixed. Furthermore, the developed method does not require any prior knowledge, which can be used for unknown anomaly detection. Another advantage of the developed method is that it can trace the anomalies easily, which can greatly improve the management efficiency. In conclusion, the proposed method is suitable for online security monitoring for medium-sized enterprise networks.

7. Application to Actual Network

7.1. Network Environment and Anomaly Mining Method

7.1.1. Campus Network Selected. We apply the proposed method to the campus network of Xi’an Jiaotong University, the northwest center of CERNET. The selected network
contains thousands of users with self-governed IP addresses, including students, faculty members, and contract personnel from service-providing companies. The services used include HTTP, email, FTP, and VoIP. To evaluate our method, we collect a trace lasting seven days from the campus network, named as CERNET dataset. The basic statistical information on the dataset is shown in Table 8. We mainly employ this dataset to verify the practicality of the proposed methods.

Figure 11: Residual matrix and residual values of specific time window. (a) Residual matrix and (b) residual values.
7.1.2. Anomaly Mining Approach. The selected monitoring network contains 1000 hosts with public IP addresses. The obtained residual matrix and the residual values of a specific time window are shown in Figure 11. Figure 11(a) is the residual matrix, \( x \)-axis is IP addresses, \( y \)-axis is seven features, and \( z \)-axis represents the changing range of residual values. As the figure shows, most of the residual values of the residual matrix are very small. It is difficult to identify abnormal hosts based on the analysis of residual values of each feature. Therefore, we calculate the residual values of each host using \( \ell_2 \) norm. The results are shown in Figure 11(b), \( x \)-axis is IP addresses, and \( y \)-axis is residual values. We analyzed the IP addresses with larger residual values to identify anomalies, which can not only determine whether an IP address is an anomaly or not but also identify the detailed patterns of the anomaly by analyzing its specific features.

We analyze the host that holds the largest residual value as an example to explain our method, its IP address is 115.154.XXX.XXX, and the residual value is 1.28. Both residual values of the OD and the NDDA are the largest in the residual matrix. It means that the host sends massive connections to different destination hosts. We can claim that the host may perform network scan attacks, as an attack host will send a great number of scan packets to multiple different destination hosts when network scan attack appears. The host with the 202.117.XXX.XXX holds the second largest residual value. The values of the NDR and the SNF of the host are 0.84 and 0.61, respectively, which shows that many different hosts send a lot of packets to this host. For DDoS attacks, they usually employ many hosts controlled by hackers to send a lot of packets to the destination host in a short time period, which makes the destination host cannot provide normal service for users as the attack consumes most of the resources \[63, 64\]. Features NDR and SNF in the residual matrix are larger compared with the others, which means that many different hosts may send massive packets to the specific host in a short time period, and these characteristics are in accordance with that of DDoS attacks. Therefore, we can conclude that the host may suffer DDoS attacks.

7.1.3. Threshold Analysis. User behaviors exhibit periodic characteristics of day and night; thus, we should select different thresholds for the day and night time. To set suitable thresholds, we select five different time windows in the daytime and nighttime to analyze their respective residual values. The results are presented in Figure 12, \( x \)-
axis is IP addresses, and y-axis is residual values. As Figure 12(a) shows, the residual values of some IP addresses are larger than 0.2 in different time windows, which occupies about 0.1%. Those behaviors are regarded as anomalies, and we set the threshold to 0.2 for daytime monitoring. As Figure 12(b) shows, the residual values of several IP addresses are larger than 0.3 in different time windows. We can set the threshold to 0.3 for nighttime monitoring. We find that the daytime threshold is smaller than that of the nighttime, as more people use the Internet in the daytime, and causes more complicated user behavior.

7.2. Performance Evaluation. We select five time windows from the daytime and nighttime, respectively. Taking the time window 13:00:00 as an example, eight residual values of the IP addresses are larger than 0.2. Seven of these IP addresses are identified as anomalies based on the analysis of their residual matrices and residual values. For the IP address 202.117.XXX.XXX, the values of NDR and LT in the residual matrix are large, which indicates that the IP address has been attacked by other hosts. For the IP address 115.154.XXX.XXX, the values of OD and NDDA are large, which means that the host is performing scan attack. Similarly, we analyze other IP addresses using their residual matrices and the results are presented in Table 9, where top $k$ denotes the number of anomalies detected by our proposed method, and $\#\text{Anomalies}$ presents the number of the true anomalies by analyzing the detected anomalies. From the table, we can determine that the precision of our proposed method sits at around 90%.

8. Conclusion
Detecting and controlling the network anomalies are one of the most important problems for network management. In this study, we propose a novel anomaly detection method based on matrix decomposition. By analyzing the behavior characteristics of attacks, we extract seven features from the network traffic to construct an attribute matrix for use in characterizing the difference between normal and abnormal user behavior patterns. We combine the attribute matrix and adjacency matrix to construct an anomaly detection model and then employ CUR matrix decomposition to mine user behavior patterns and obtain a residual matrix to identify anomalies. We use two datasets, MAWI and CERNET, to evaluate the performance of our proposed method. The experimental results show that the proposed method achieves a detection accuracy larger than 90%, which verifies that it outperforms other related methods. Moreover, the developed method can not only locate anomalies and interpret the anomaly detection process but also can identify new anomalies without any prior knowledge. In future work, we will focus on how to reduce the computational complexity and improve the practicality of the algorithm.

Data Availability
In this study, the public data can be available through the URL http://www.fukuda-lab.org/mawilab/v1.1/2019/05/20190505.html and the data collected from CERNET are available from the corresponding author upon request.

Conflicts of Interest
The authors declare that they have no conflicts of interest.

Acknowledgments
The authors would like to express their grateful thanks to Liang Zhao for his data preprocessing efforts, as well as to Guodong Li, from the campus network center, for his efforts in traffic trace collection. The research presented in this study is supported in part by the National Natural Science Foundation of China (62172324 and 62102310) and China Postdoctoral Science Foundation (2020M683689XB).

References
[1] B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen, “Sketch-based change detection: methods, evaluation, and applications,” in Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement, pp. 234–247, Miami Beach FL USA, October 2003.
[2] A. Lakhina, M. Crovella, and C. Diot, “Mining anomalies using traffic feature distributions,” ACM SIGCOMM - Computer Communication Review, vol. 35, no. 4, pp. 217–228, 2005.
[3] X. Guan, T. Qin, W. Li, and P. Wang, “Dynamic feature analysis and measurement for large-scale network traffic monitoring,” IEEE Transactions on Information Forensics and Security, vol. 5, no. 4, pp. 905–919, 2010.
[4] C. Yang, “Anomaly network traffic detection algorithm based on information entropy measurement under the cloud computing environment,” Cluster Computing, vol. 22, no. S4, pp. 8309–8317, 2019.
[5] W. Wang, Y. Shang, Y. He, Y. Li, and J. Liu, “BotMark: automated botnet detection with hybrid analysis of flow-based and graph-based traffic behaviors,” Information Sciences, vol. 511, pp. 284–296, 2020.
[6] M. Mahoney and P. Chan, “Packet header anomaly detection for identifying hostile network traffic,” in Proceedings of the 2003 ACM Symposium on Applied Computing, pp. 346–350, Melbourne, FL, USA, 2001.
[7] F. Iglesias and T. Zsely, “Analysis of network traffic features for anomaly detection,” Machine Learning, vol. 101, no. 1-3, pp. 59–84, 2015.
[8] N. Moustafa, B. Turnbull, and K. K. R. Choo, “An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things,” IEEE Internet of Things Journal, vol. 6, no. 3, pp. 4815–4830, 2019.
[9] A. Youssef and A. Emam, “Network intrusion detection using data mining and network behaviour analysis,” International Journal of Computer Science and Information Technology, vol. 3, no. 6, pp. 87–98, 2011.
[10] B. Selvakumar and K. Muneeswaran, “Firefly algorithm based feature selection for network intrusion detection,” Computers & Security, vol. 81, pp. 148–155, 2019.
16 Security and Communication Networks

[11] G. Marin, P. Casas, and G. Capdehourat, “Rawpower: deep learning based anomaly detection from raw network traffic measurements,” in Proceedings of the ACM SIGCOMM 2018 Conference on Posters and Demos, pp. 75–77, 2018.

[12] W. A. Ali, K. N. Manasa, M. Aljunid, M. Bendechache, and P. Sandhya, “Review of current machine learning approaches for anomaly detection in network traffic,” Journal of Telecommunications and the Digital Economy, vol. 8, no. 4, pp. 64–95, 2020.

[13] N. Hoque, D. K. Bhattacharyya, and J. K. Kalita, “Botnet in DDoS attacks: trends and challenges,” IEEE Communications Surveys & Tutorials, vol. 17, no. 4, pp. 2242–2270, 2015.

[14] J. Zhang, R. Perdisci, W. Lee, X. Luo, and U Sarfraz, “Building a scalable system for stealthy P2P-botnet detection,” IEEE Transactions on Information Forensics and Security, vol. 9, no. 1, pp. 27–38, 2014.

[15] J. Zhang, X. Luo, R. Perdisci, and G. Gu, “Boosting the scalability of botnet detection using adaptive traffic sampling,” in Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, pp. 124–134, 2011.

[16] J. Zhang, R. Perdisci, W. Lee, U. Sarfraz, and X. Luo, “Detecting stealthy P2P botnets using statistical traffic fingerprints,” in Proceedings of the 2011 IEEE/IFIP 41st International Conference on Dependable Systems & Networks (DSN), pp. 121–132, IEEE, 2011.

[17] L. Akoglu, H. Tong, and D. Koutra, “Graph based anomaly detection and description: a survey,” Data Mining and Knowledge Discovery, vol. 29, no. 3, pp. 626–688, 2015.

[18] K. Xie, X. Li, X. Wang et al., “On-Line anomaly detection with high accuracy,” IEEE/ACM Transactions on Networking, vol. 26, no. 3, pp. 1222–1235, 2018.

[19] C. Li, X. Wang, W. Dong, J. Yan, Q. Liu, and H. Zha, “Joint active learning with feature selection via CUR matrix decomposition,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 41, no. 6, pp. 1382–1396, 2019.

[20] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda, “Mawilab: combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking,” in Proceedings of the 6th International Conference on Cyberprints, pp. 1–6, IEEE, 2016.

[21] C. C. Aggarwal, “Outlier analysis,” Data Mining, pp. 237–263, Springer, Cham, Switzerland, 2015.

[22] M. Ahmed, A. Naser Mahmood, and J. Hu, “A survey of network anomaly detection techniques,” Journal of Network and Computer Applications, vol. 60, pp. 19–31, 2016.

[23] B. Li, S. Zhang, and K. Li, “Towards a multi-layers anomaly detection framework for analyzing network traffic,” Concurrency and Computation: Practice and Experience, vol. 29, no. 14, Article ID e3955, 2017.

[24] Z. Benjelloun, A. A. Lahcen, and S. Belfkih, “Outlier detection techniques for big data streams: focus on cyber security,” International Journal of Internet Technology and Secured Transactions, vol. 9, no. 4, p. 446, 2019.

[25] M. Ahmed, A. N. Mahmood, and M. R. Islam, “A survey of anomaly detection techniques in financial domain,” Future Generation Computer Systems, vol. 55, pp. 278–288, 2016.

[26] G. van Capelleveen, M. Poel, R. M. Mueller, D. Thornton, and J. van Hilligersberg, “Outlier detection in healthcare fraud: a case study in the Medicaid dental domain,” International Journal of Accounting Information Systems, vol. 21, pp. 18–31, 2016.

[27] K. Anand, J. Kumar, and K. Anand, “Anomaly detection in online social network: a survey,” in Proceedings of the 2017 International Conference on Inventive Communication and Computational Technologies (ICICCT), pp. 456–459, IEEE, 2017.

[28] S. Agrawal and J. Agrawal, “Survey on anomaly detection using data mining techniques,” Procedia Computer Science, vol. 60, pp. 708–713, 2015.

[29] L. Kong, G. Huang, and K. Wu, “Identification of abnormal network traffic using support vector machine,” in Proceedings of the 2017 18th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAST), pp. 288–292, IEEE, 2017.

[30] J. Gu, L. Wang, H. Wang, and S. Wang, “A novel approach to intrusion detection using SVM ensemble with feature augmentation,” Computers & Security, vol. 86, pp. 53–62, 2019.

[31] M. Swarnkar and N. Hubballi, “OCPAD: one class Naive Bayes classifier for payload based anomaly detection,” Expert Systems with Applications, vol. 64, pp. 330–339, 2016.

[32] X. Han, L. Xu, M. Ren, and W. Gu, “A Naive Bayesian network intrusion detection algorithm based on Principal Component Analysis,” in Proceedings of the 2015 7th International Conference on Information Technology in Medicine and Education (ITME), pp. 325–328, IEEE, 2015.

[33] L. Nie, D. Jiang, and Z. Lv, “Modeling network traffic for traffic matrix estimation and anomaly detection based on Bayesian network in cloud computing networks,” Annals of Telecommunications, vol. 72, no. 5–6, pp. 297–305, 2017.

[34] E. Hodo, X. Bellekens, A. Hamilton et al., “Threat analysis of IoT networks using artificial neural network intrusion detection system,” in Proceedings of the 2016 International Symposium on Networks, Computers and Communications (ISNCC), pp. 1–6, IEEE, 2016.

[35] D. Kwon, K. Natarajan, S. C. Suh, H. Kim, and J. Kim, “An empirical study on network anomaly detection using convolutional neural networks,” in Proceedings of the 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), pp. 1595–1598, IEEE, 2018.

[36] R. Vinayakumar, K. P. Soman, and P. Poornachandran, “Evaluation of recurrent neural network and its variants for intrusion detection system (IDS),” International Journal of Information System Modeling and Design, vol. 8, no. 3, pp. 43–63, 2017.

[37] Y. S. Thakare and S. B. Bagal, “Performance evaluation of K-means clustering algorithm with various distance metrics,” International Journal of Computer Application, vol. 110, no. 11, pp. 12–16, 2015.

[38] R. Kumari, M. K. Singh, R. Iha, and N. K. Singh, “Anomaly detection in network traffic using K-means clustering,” in Proceedings of the 2016 3rd International Conference on Recent Advances in Information Technology (RAIT), pp. 387–393, IEEE, 2016.

[39] Z. Chen, C. K. Yeo, and B. S. Lee, “Autoencoder-based network anomaly detection,” in Proceedings of the 2018 Wireless Telecommunications Symposium (WTS), pp. 1–5, IEEE, 2018.

[40] M. Said Elsayed, N. A. Le-Khac, S. Dev, and A. D. Jurcut, “Anomaly detection and description: a survey,” IEEE Transactions on Knowledge Discovery and Data Mining, vol. 29, no. 3, pp. 626–688, 2015.

[41] J. Gu, L. Wang, H. Wang, and S. Wang, “A novel approach to intrusion detection using SVM ensemble with feature augmentation,” Computers & Security, vol. 86, pp. 53–62, 2019.

[42] W. Wang and R. Battiti, “Identifying intrusions in computer networks with principal component analysis,” in Proceedings
M. Deng and B. Wu, “Self-adaptive threshold traffic anomaly detection based on ϕ-entropy and the improved EWMA model,” in Proceedings of the 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), vol. 1, pp. 725–730, IEEE, Chongqing, China, 2020.

N. N. Tuan, N. D. Nghia, P. H. Hung et al., “An abnormal network traffic detection scheme using local outlier factor in SDN,” IEEE, in Proceedings of the 2020 IEEE Eighth International Conference on Communications and Electronics (ICCE), pp. 141–146, 2021.

J. Auskalnis, N. Paulauskas, and A. Baskys, “Application of local outlier factor algorithm to detect anomalies in computer network,” Elektronika ir Elektrotechnika, vol. 24, no. 3, pp. 96–99, 2018.

Z. He, X. Xu, and S. Deng, “Discovering cluster-based local outliers,” Pattern Recognition Letters, vol. 24, no. 9-10, pp. 1641–1650, 2003.

J. Tang, Z. Chen, A. W. Fu, and D. Cheung, “A robust outlier detection scheme for large data sets,” in Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Taipei, Taiwan, 2002.

Y. Pei, O. R. Zaiane, and Y. Gao, “An efficient reference-based approach to outlier detection in large datasets,” in Proceedings of the Sixth International Conference on Data Mining (ICDM’06), pp. 478–487, IEEE, 2006.

C. Wang, T. T. N. Miu, X. Luo, and J. Wang, “SkyShield: a sketch-based defense system Against application layer DDoS attacks,” IEEE Transactions on Information Forensics and Security, vol. 13, no. 3, pp. 559–573, 2018.

X. Luo, E. W. W. Chan, and R. K. C. Chang, “Detecting pulsing denial-of-service attacks with nondeterministic attack intervals,” EURASIP Journal on Applied Signal Processing, vol. 2009, Article ID 256821, 2009.

Y. Jiang, Y. Chen, F. Hu, G. Zhang, and P. Lin, “Behavior similarity awared abnormal service identification mechanism,” in Proceedings of the International Conference on Computer Engineering and Networks, pp. 316–324, Springer, Cham, 2018.

R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, “The 1999 DARPA off-line intrusion detection evaluation,” Computer Networks, vol. 34, no. 4, pp. 579–595, 2000.

M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 data set,” in Proceedings of the 2009 IEEE symposium on computational intelligence for security and defense applications, pp. 1–6, IEEE, 2009.

M. Almseidin, “Detecting distributed denial of service attacks using data mining techniques,” International Journal of Advanced Computer Science and Applications, vol. 7, no. 1, pp. 436–445, 2016.

I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in Proceedings of the ICISSp, pp. 108–116, 2018.

A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, “Toward developing a systematic approach to generate benchmark datasets for intrusion detection,” Computers & Security, vol. 31, no. 3, pp. 357–374, 2012.

M. Hossin and M. N. Sulaiman, “A review on evaluation metrics for data classification evaluations,” International Journal of Data Mining & Knowledge Management Process, vol. 5, no. 2, p. 1, 2015.