MSCA: An Unsupervised Anomaly Detection System for Network Security in Backbone Network

Yating Liu, *Student Member, IEEE*, Yuantao Gu, *Senior Member, IEEE*, Xinyue Shen, Qingmin Liao, *Senior Member, IEEE*, and Quan Yu

***Abstract*—Anomaly detection is a crucial topic in network security which refers to automatically mining known and unknown attacks or threats. Many detectors have been proposed in the last decade. Nonetheless, a practical solution, which is able to provide a high True Positive Rate (TPR) with an acceptable False Positive Rate (FPR) without any prior information, is still challenging due to the complexity and variability of anomaly pattern. In this article, we propose a novel unsupervised detection system called MSCA which applies multiple sketches, K-means++ unsupervised clustering, and association rule mining to detect traffic anomalies and analyze anomalous features and correlations. It can blindly identify known and unknown traffic anomalies without any labeled traffic or prior signatures about data distribution. Rich traffic data is first aggregated and compacted to traffic flows by sketches, and further detected by the combination of clustering algorithm and voting strategy. Then association rule mining is finally utilized to find the anomalous frequent item-sets and association rules. Numerical experiments on MAWILAB datasets demonstrate that the proposed detection method outperforms other reference unsupervised detection methods. It achieves an accuracy of 99.86%, 99.97%, 97.08%, and 95.19% in overall four detection types including IP and port of source and destination.**

***Index Terms*—Anomaly detection, association rule mining, backbone network, clustering, random projections, sketches, traffic anomalies.**

I. Introduction

ANOMALY detection identifies attacks based on their significant deviations from normal activities, which will cause unreasonable resource allocation and even network congestion. Backbone Network is a central pipe designed to transmit network traffic at high speed, which usually connects local area networks (LANs) and wide area networks (WANs). It is essential to maximize the reliability and performance of large-scale and long-distance communication. Therefore, in order to ensure regular network operations and the quality of backbone network service, it is crucial to identify anomalies automatically and precisely.

However, anomaly detection is highly challenging with the rapid development of Internet. First, anomaly behavior tends to be highly complicated and diverse [1], [2], [3], [4]. They can be caused by known attacks such as Flash Crowd, Port Scan, DDos, and unknown ones, whose behaviors are significantly different in terms of duration and attack target. Second, network traffic data exhibits a wild variability at time granularity (from microsecond to daily), geographic space (single router or core networks), and statistical properties such as heavy tails and long range dependency. Third, precisely constructing a traffic reference is non-trivial, since the characteristics of anomalies are various for different types of Internet platforms [2], [3], [5]. Fourth, processing a huge amount of high-dimensional and noisy traffic data effectively requires low computational cost and online techniques [6]. Moreover, the existent types of anomalies are constantly shifting, and at the same time, new types of anomalies are born every day, which indicates that the anomaly family keeps changing, making the detection becomes even harder [7]. Last but not least, data privacy may also be an obstacle to anomaly detection [8].

In order to address the aforementioned problems, efforts have been made by different researchers, whose proposed methods can be roughly classified into two main categories. The first one is about supervised detection methods, which mainly focus on machine learning-based methods [9], [10], [11], [12], [13], [14] by using normal and anomalous labeled data. However, there exist many remarkable differences in the features of anomalous flows in various scenarios, which makes labeling data much time-consuming and laborious. Therefore, it is rather impractical to maintain a precise and up-to-date classifier, especially when anomalies are constantly changing. The second category is about unsupervised detection methods, which do not require data preparation and can be directly imposed on the online traffic data with little prior information. They can be further classified into statistics-based methods [2], [3], [15], [16] and unsupervised machine learning methods [7], [17], [18]. Compared with the supervised methods, the unsupervised ones could lead to an improvement in detecting unknown anomalies, which are more convenient to update anomaly types, and thus come into widespread use in network management systems in recent years.

Manuscript received 13 November 2021; revised 9 August 2022; accepted 5 September 2022. Date of publication 14 September 2022; date of current version 6 January 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 61971266, Grant from the Guoqiang Institute, Tsinghua University, and in part by the Clinical Medicine Development Fund of Tsinghua University. Recommended for acceptance by Dr. Gaoxi Xiao. *(Corresponding author: Yuantao Gu.)*

Yating Liu is with Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China, and also with Peng Cheng Laboratory, Shenzhen 518055, China (e-mail: liuyatin21@mails.tsinghua.edu.cn).

Yuantao Gu is with the Department of Electronic and Engineering, Tsinghua University, Beijing 100084, China (e-mail: gyt@tsinghua.edu.cn).

Xinyue Shen was with the Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China (e-mail: xinyueshen@outlook.com).

Qingmin Liao is with Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China (e-mail: liaoqm@tsinghua.edu.cn).

Quan Yu is with the Peng Cheng Laboratory, Shenzhen 518055, China (e-mail: quanyu@ieee.org).

Digital Object Identifier 10.1109/TNSE.2022.3206353

Previous works [2], [3], [7], [15], [16], [17], [18] have shown that unsupervised learning methods in anomaly detection can achieve good results. However, many available detection methods are still based on prior information such as the hypothesis about data distribution and domain knowledge or the intuition for anomaly type. Another challenge is the high ratio that the normal flows are misjudged as abnormal due to the imbalanced dataset. This paper aims to design a new unsupervised anomaly detection method, which uses the traffic time series of IP/Port packets in discrete time bins, and deals with timestamps and five-tuples plus packet number without any label and priori information, where a five-tuple includes source IP (IPsrc), destination IP (IPdst), source Port (PortSrc), destination Port (PortDst) and protocol.

Actually, voting strategy has been used in supervised learning techniques such as Random Forest [19] that consists of multiple decision trees and outputs the results which obtain more votes to handle classification problem. We propose an unsupervised anomaly detection system called Multiple Sketches, Clustering, and Apriori (MSCA) to incorporate multiple sketches (random projections) and voting strategy to increase the robustness of detection by synthesizing multiple independent results. Clustering is used as its identification standard need not be tuned according to specific traffic data and anomaly patterns compared with the aforementioned unsupervised algorithms. Furthermore, association rule mining algorithm is applied to discover the deep-rooted anomalous features and correlations.

An early partial version of this work was presented in [20], which proposed a fundamental method of combining multiple sketches and clustering in anomaly detection. Compared with the workshop paper, this paper establishes a complete detection framework including raw data processing, anomaly detection procedure and root-cause analysis, and elaborates the function of each module. Meanwhile, experiments for parameters selection, multiple fine grained performance comparisons including protocol classification and anomaly classification, feature comparison (Packet v.s. Entropy), unsupervised and supervised methods comparison, and anomalous flows rules mining are complemented. Experimental data is expanded from two months to six months and updated in August 2019.

In summary, our contributions are as follows.

1) Propose a novel clustering-based detection method called Multiple Sketches and Clustering (MSC), which utilizes multiple sketches, multiple K-means++ clustering, and voting strategy to improve TPR and reduce FPR of anomaly detection effectively without any prior information about traffic data.
2) Construct a fully functional anomaly detection application system (MSCA), which is capable of performing the entire detection process including flow generation from raw packets, anomaly detection of objects (IP address or port number) and feature extraction of anomalous flows. In addition, sliding time window is adopted in this system to avoid processing massive data directly and make online model possible in real scenario.
3) Perform extensive experiments to validate the performance superiority of the proposed method and provide explicit discussion about the detection results.

The paper is organized as follows. The existing anomaly detection methods and the commonly used open datasets are briefly reviewed in Section II. Useful preliminaries are introduced in Section III, and the proposed framework including three procedures are described in Section IV. Afterwards, extensive experiments are performed to verify the effectiveness of the proposed method and compare with existing methods in Section V, and conclusions are drawn in Section VI.

II. Literature Survey

A. Anomaly Detection Methods

As we have introduced, there are roughly two categories of network anomaly detection methods, namely, the supervised methods and the unsupervised methods. Existing supervised anomaly detection mainly utilizes supervised machine learning methods such as support vector machine (SVM) [10], adaboost [11], random forest [21], and the deep learning-based method [22] using Long-Short-Term-Memory (LSTM) integrated with Genetic Algorithm (GA), etc. The other supervised method type is ensemble learning to enhance the performance of single classifier such as the Super Learner [12] composed of SVM, decision trees, KNN, perception neural network, and naive Bayes, etc. The usage of aforementioned supervised anomaly detection methods are limited by the difficulty of labeling the network traffic data. Hence, unsupervised anomaly detection methods have drawn increasing attention which can be summarized into two categories: statistics-based methods and unsupervised machine learning methods.

1) Statistics-Based Methods: Principal Component Analysis (PCA)-based methods [23], [24], [25] identify anomalous flows according to normal and abnormal subspaces by separately extracting the principal components and residual of a traffic matrix via PCA. Then, it detects an anomaly if the squared prediction error generated by the residual of a flow deviates from a regular threshold determined by *Q-statistic*. In [15], Feature Multi-scale PCA (Feature-MSPCA) and Packet Multi-scale PCA (Packet-MSPCA), which incorporate the sketch and the wavelet transform, are proposed as an extension of traditional PCA-based methods. Explicitly, in Feature-MSPCA, the statistical features of the three dominant protocols, TCP, UDP, and ICMP, are used as the input. While, Packet-MSPCA only uses the packet number as the input feature. A Kullback-Leibler (KL)-histogram method [16] finds the anomalous time slots and their corresponding anomalous objects by calculating the KL divergence of the traffic histograms of every two adjacent time slots. Then, the anomalous time slots and objects with prominent changes in traffic volume are distinguished from the normal ones based on the assumption that the first-order difference of the KL divergence series follows standard Gaussian distribution. Sketching and multi-resolution Gamma modeling [2] essentially perform transformation in time domain based on sketch flows and Gamma distribution. It detects a flow as anomaly, when the Mahalanobis distance between the flow and the standard flow exceeds an adaptive threshold. The work presented in [3] detects anomalies by extracting coefficients of wavelet transform. When the

Euclidean distance between the coefficients and the relative average coefficients is distant from a computed reference, the corresponding traffic is judged as anomalous. Other statistics-based unsupervised detection methods use entropy estimation [26], the histogram [27], or other distribution feature [28].

2) Unsupervised Machine Learning-Based Methods: A Hough-based method [17] detects anomalies by recognizing straight lines in a 2-D scatter using Hough-transform, where the 2-D scatter is composed of a combination of two objects selected from IPsrc, IPdst, PortSrc, and PortDst. The objects corresponding to the coordinates of the straight lines are reported as anomalous. Another type utilizes clustering methods such as K-means, hierarchical clustering, and spectral clustering for anomaly detection [29]. In practice, K-means is often more suitable for anomaly detection than other clustering algorithms due to its sensitivity to outliers and isolated samples, but its randomly generated initial cluster centers affect the clustering results and the convergence rate. An improved algorithm is introduced in [30] known as K-means++, which makes the initial clustering centers as far from each other as possible, and thus, avoids the instability of results and accelerates the iterative speed of clustering. Therefore, in this paper, we choose K-means++ as the clustering method in our detection framework. An innovative model of anomaly detection in network intrusion detection system based on unsupervised learning technique (K-means) is first described in [31]. Meanwhile, the density-grid-based clustering detection method is proposed in [32]. A detection method is designed by applying subspace clustering to anomaly detection [33], based on which an improved detection framework is introduced in [7] and achieves good performance on MAWILAB dataset. A detailed and comprehensive introduction to anomaly detection based on clustering is presented in [34]. A recent study on the combination of anomaly detection methods is conducted in [18], which is a hybrid method that utilizes four state-of-art detection methods effectively including PCA-based [23], Gamma-based [2], KL-based [16] and Hough-based [17] methods, and has achieved good performance on MAWILAB dataset.

B. Aggregation Methods

Data flows should be aggregated and stored as network flows efficiently to reduce traffic interference [15]. There are two major approaches of traffic aggregation: deterministic methods and methods based on random projection. The former includes Origin-Destination (OD) flow aggregation and binary flow aggregation. OD flow [24] is the traffic that enters the network at the origin Points of Presence (PoP) and exits at the destination PoP. Each link traffic in the network is derived from the superposition of multiple OD flows. The relationship between link traffic and OD flow is established by the routing matrix. However, it is a demanding task to capture OD flows directly and concisely in a backbone network [35]. Another deterministic method is binary flow aggregation [17], which uses a deterministic hash table. As a consequence, this method is highly limited because of its fixed and artificial mapping rule. To overcome the shortcomings of the previous works, a new aggregation strategy is designed by using random projection (sketch) [23]. Random projection [36], [37], namely, the process of universal hashing or sketching, is used to aggregate flows from different random seeds. It overcomes the inherent drawback in the extraction of OD flows. Compared with the binary aggregation, multiple sketches are performed to reduce FPR by synthesizing multiple detection results. There are other works which exploit random projection [2], [3], [38], [39]. Hence, random projection is utilized to aggregate the initial flows in our framework.

C. Datasets

A reliable benchmark dataset is crucial to the evaluation of anomaly detection methods, and has been a main focus in the study. DARPA'98 [40] dataset is an earlier dataset which is collected from the synthetic US Air Force local area network (LAN) for nine weeks by the Lincoln Laboratory. Subsequently, researchers further parse the network packets of DAPRA'98 to obtain the KDD Cup 1999 dataset (KDD'99) with 41 dimensions. The later has been extensively studied in all kinds of simulated experiments for supervised anomaly detection methods.

In this paper, we fully exploit the MAWILAB [41] dataset, which is captured by MAWI Labs across the Pacific backbone network between Japan and the USA. It has several critical advantages over DAPRA'98. First, MAWILAB contains a long-term traffic data ranging from 2001 until now, and keeps up-to-date over time. Thus, anomalous types are extremely various, since it possesses more recent anomalies than DAPRA'98. Second, MAWILAB provides a more representative real-world network traffic, whereas DARPA'98 is military simulation data. Third, the network size of MAWILAB, which is captured from a large backbone network and provides rich information at the spatial level, is obviously larger than that of DARPA'98 standing for LAN. Hence, MAWILAB dataset is adopted in this paper to obtain authentic and reliable results.

III. Preliminaries

A. Random Projection (Sketch)

To compactly store the voluminous network flows in an efficient manner [25], we apply k-universal hash functions [36] to IP address or port number so that their transformed hash value can be mapped to a specific set ($\{0, 1, \ldots, M-1\}$). Then the IPs or ports which have same hash value are respectively aggregated in traffic. The aforementioned procedure is called random projection or sketch. IP address and port number are used in our method as the targets to be detected, where the corresponding algorithms for sketch are given as follows.

Port number. Denote the port number as an integer $port$, with the aid of 1-universal hash functions, $port$ can be mapped as

$$h_{a,b}(port) = ((a \cdot port + b) \bmod p) \bmod M, \qquad (1)$$

where $h_{a,b}(port)$ is the hash value of port number, p is a random large prime so that each hash value of $port$ falls between $0 \sim p-1$ at first, M denotes the size of the hash table, the

operation "mod" represents taking the residue, and $a, b \in \{0, 1, \ldots, p-1\}$.

IP address. Each IP is a 32-bit 4-tuples $\langle x_1, x_2, x_3, x_4 \rangle$, where $x_i \in \{0, 1, \ldots, 255\}$. We define $a = \langle a_1, a_2, a_3, a_4 \rangle$ with $a_i \in \{0, 1, \ldots, p-1\}$. Then, 4-universal hash functions can be formulated as

$$h_a(ip) = \left(\left(\sum_{i=1}^{4} a_i x_i \right) \bmod p \right) \bmod M, \tag{2}$$

where $h_a(ip)$ is the hash value of IP address and the other definitions including p and M.

B. K-Means++ Clustering

We use K-means++ algorithm to overcome the shortcoming of K-means that the initial clustering centers are not stable and thus the convergency and efficiency is poor. The basic principle of K-means++ selecting the initial clustering centers is that the distance among those centers should be maximized. It consists of the following steps.

Step 1: A sample is randomly selected from the multi-dimension data as the first cluster center.

Step 2: For each sample i, calculate the distance from i to the existing cluster centers and choose the minimum as the nearest distance $D(i)$ of i.

Step 3: The sample with the maximal $D(i)$ is chosen as a new cluster center to ensure that the sum of the distance among the clusters has relative maximum.

Step 4: Repeat Step 2 and Step 3 until K clustering centers are selected.

Step 5: Standard K-means clustering is performed after determining the K initial clustering centers.

C. Apriori

Item-set depicts a combination containing one or more objects. Apriori algorithm [42], typically used in data mining, can discover the item-sets that frequently emerge together and the association rules that describe strong correlation between the aforementioned item-sets.

1) Frequent Item-Sets Generation: By defining the support of an item-set as the frequencies of transactions containing this item-set, a k-item-set represents the combination with k elements. The detailed procedure for generating frequent item-sets F are as follows.

Step 1: For $k = 1$, calculate the support of all 1-item-sets and select the frequent 1-item-sets whose support are not less than the given threshold $minSup$.

Step 2: For $k > 1$, k-dimensional candidates (k-item-sets) are generated according to the frequent (k-1)-item-sets. Next, we traverse the database to obtain the supports of the candidates. Finally, all frequent k-item-sets F are selected under the constraint of $minSup$.

Step 3: Repeat step 2 until no more frequent item-set is generated.

2) Association Rules Generation: By defining confidence as the probability that one item-set is included in another item-set, the association rules R are derived from frequent item-sets F as below.

Step 1: For each frequent item-set $I \in F$, we generate the non-empty subsets J of I.

Step 2: For each non-empty subset J of I, if $P(I)/P(J) \geq minConf$, then the association rule $J \Rightarrow I - J$ is generated, where $P(S)$ represents the proportion of the number of transactions containing the set S in total transactions and $minConf$ is the threshold of confidence. The higher $P(I)/P(J)$ indicates the stronger correlation between J and $I - J$.

TABLE I
DEFINITION OF SYMBOLS

symbol	description
$object$	source IP (IPsrc) destination IP (IPdst) source port (PortSrc) destinition port (PortDst)
Δt	time scale
$m \in \{0, 1, 2, \ldots, M-1\}$	sketch output number (M = the size of hash table)
$n \in \{1, 2, \ldots, N\}$	sketch number (N = number of hash functions)
T	sliding time window
L	the threshold of votes
K	the number of clusters

IV. METHODOLOGY

The three procedures for the proposed system are flow generation, anomaly detection, and feature extraction, which are shown in detail in Fig. 1.

Flow generation processes the original tcpdump data packets and generates multiple time series of each object (IP address or port number) to be detected. Next, we propose an unsupervised anomaly detection method which uses multiple random projections and clustering to deal with multiple series of objects and detect anomalies. Finally, Apriori algorithm is applied to discover and summarize the frequent item-sets and association rules of anomalous flows, so that the anomalous features are accurately extracted and conveniently understood by network operators.

A. Flow Generation

Flow generation as depicted in Fig. 2, as the basis of the subsequent steps, aims at converting raw tcpdump data to packet number time series for each IP or port under a specific time scale.

Step 1 transaction generator. Tcpdump data is converted to transactions by a network protocol analyzer called *Tshark*, which is designed to dump and analyze raw network traffic. A transaction is composed of arrival timestamp, five-tuple, and the size of network packet.

Step 2 discrete flow generator. The transactions are grouped by a time scale Δt, and in each group the packet size and number of the transactions with the same five-tuple are aggregated to generate multiple discrete flows, where a discrete flow or flow indicates the total packet size and packet number of a five-tuple in a time slot. Note that there are two aggregation operations in our method. The aggregation in this step is

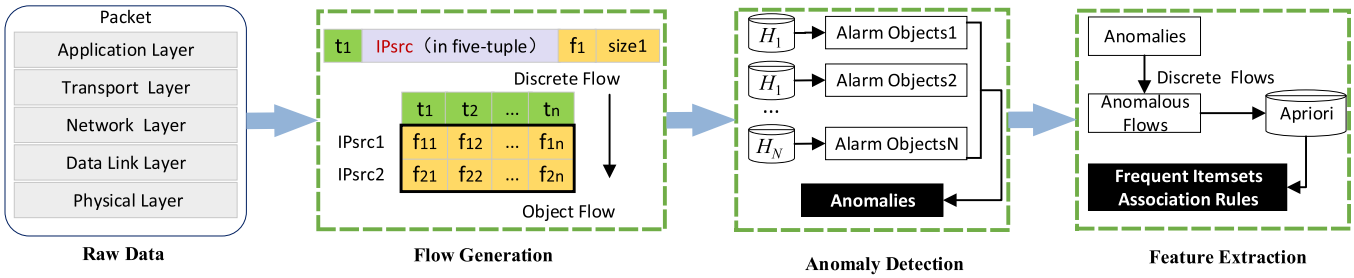


Fig. 1. Flowchart of the proposed method. In flow generation, each timestamp t_i is a slot spanning a time scale Δt, such as the first slot t_1. The combination of a timestamp, five-tuple, frequency (packet number) f_i, and packet size is called discrete flow. f_{ij} is the frequency of the object (IPsrc, IPdst, PortSrc, or PortDst), and a numerical time series by a certain object, such as $f_{11}, f_{12}, \ldots, f_{1n}$ is called object flow.

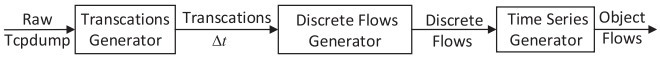


Fig. 2. Diagram of the flow generation.

definite, and the other that will be introduced in anomaly detection is random.

Step 3 time series generator. The packet numbers of discrete flows with same object are added up in each time slot to get the corresponding time series for each object, which is defined as object flow (OF). In fact, aside from packet number, the packet size can be also utilized to calculate OF. However, the packet size is significantly different for various protocols and applications, and we focus on the anomalous activity of one object more than its context here.

B. Anomaly Detection

We specify the proposed unsupervised anomaly detection method in this subsection. First, for each object flow, random projection is utilized as an aggregation method to compact and store original data structure from different random hash views. Then, we adopt K-means++ clustering to process the compressed data flows and get an alarm object set for each time window, which is referred to as alarm detection. The final output of anomaly detection is a list of anomalies obtained by voting for all alarm objects of multiple random projections and multiple alarm detections. The details of the method are given as follows step by step.

1) Alarm Detection Based on Clustering: As shown in Fig. 3, alarm detection as the critical process of anomaly detection, applies random projection, K-means++ clustering and versa hashing to get alarm objects under a deterministic hashing rule and a specific sliding time window. The main scheme is as follows.

ASF generation. Each object is mapped by sketch to get its corresponding hash value called Sketch Output Number (SON) symbolled as m. Packet numbers of OFs which correspond to a same SON and same time slot are added up together. Then, the time series of packet number for each SON is generated under a specific sliding time window of length T, which is defined as Aggregated Sketch Flow (ASF).

K-means++ clustering. Next, K-means++ clustering is performed on a set of multiple ASFs, where each ASF of SON is a sample with length T.

In each time window, the elements included in the largest cluster are filtered, and are considered to be normal. Meanwhile, the other SONs are labeled as alarm SONs, which are suspected to be abnormal, and are fed into the subsequent anomaly detection procedures. The above-mentioned operations are based on the large amount of experimental analysis in [7] and the criterion that the number of anomalous flows are much smaller than the normal ones.

Versa hashing. Alarm SONs are transformed to alarm objects by versa hashing operation, which is an inverse process of random projection.

2) Multiple Sketches and Clustering (MSC): As shown in Fig. 1, a central issue in anomaly detection lies in performing multiple alarm detection based on clustering and voting strategy to find the anomalous objects. The detailed description is illustrated as follows.

Step 1 sliding-time-window. The sliding time window of length T is used to separate the total slots S into multiple equal ranges to ensure online processing. Then, C clustering detections under the nth sketch will be implemented in Step 2, where C is the result of S divided by T, and denotes the total number of sliding time windows.

Step 2 alarm detection based on clustering. All detectable objects are randomly mapped under the nth sketch to get its corresponding SON(m), where $n \in \{1, 2, \ldots, N\}$. For each sliding window $c \in \{1, 2, \ldots, C\}$, we carry out alarm detection on the multiple ASFs and output the suspicious objects $alarmObj_n^c$.

Step 3 alarm sets union. The aforementioned multiple alarm sets in the nth hashing during the entire detection time period are merged as an alarm object set $alarmObj_n$, defined as

$$alarmObj_n = \bigcup_{c=1}^{C} alarmObj_n^c. \tag{3}$$

Step 4 voting strategy. We implement Step2 to Step3 N times under N hashing functions to obtain N alarm object sets. Then, voting strategy is applied to filter the normal objects mixed in N alarm object sets and obtain the frequent anomalous object set $anomalyObj_f$, defined as

$$anomalyObj = \{O | count(O) > L, O \in \bigcup_{n=1}^{N} alarmObj_n\} \tag{4}$$

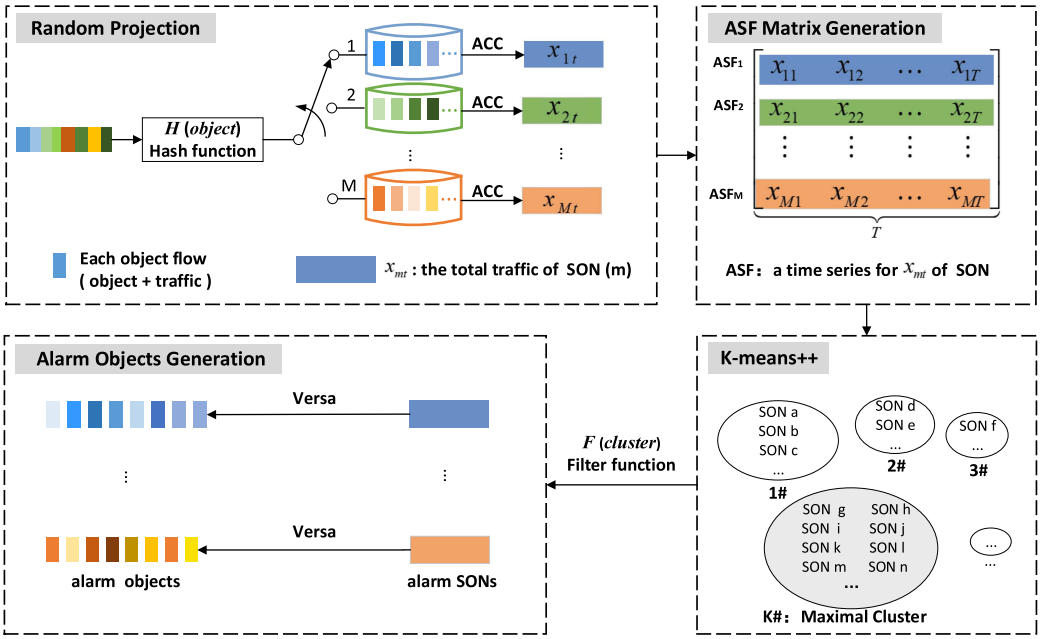


Fig. 3. Diagram of the alarm detection. An object (IP address or Port) is hashed into a Sketch Output Number (SON), which is between 0 and $M-1$. x_{mt} is the accumulation of same hash value for multiple objects. Aggregation Sketch Flow (ASF) is the time series of x_{mt}, such as $x_{m1}, x_{m2}, \ldots, x_{mT}$. For instance, if object IPsrc 168.111.21.31 and 168.8.21.31 have same hash value, their corresponding traffic can be accumulated into one SON.

where $count$ is the corresponding statistical times of object in N alarm detections and L is the threshold of votes. An object is detected as anomalous when its $count > L$.

Multiple sketches play a key role as they decrease the probability of random collisions, and therefore improve the randomness. It is seen that, with the randomness improved, anomalies with more votes can present higher abnormality degree, and hence guarantee higher detection accuracy than a single sketch.

The average number of collisions diminishes exponentially with the increase of N. If M is the size of hash table, N_o is the total number of object captured in the corresponding sliding time window, then the average number of collisions $\#C$ is

$$\#C = N_o M^{-2\,N}. \tag{5}$$

Meanwhile, the probability of an object being judged correctly is

$$P_t = \sum_{i=L}^{N} \binom{N}{i} {p_t}^i (1-p_t)^{N-i}, \tag{6}$$

and the probability of an object being misjudged is

$$P_f = \sum_{i=L}^{N} \binom{N}{i} {p_f}^i (1-p_f)^{N-i}, \tag{7}$$

where p_t is the True Positive Rate (TPR), p_f denotes False Positive Rate (FPR) in one experiment, and $L = N/2+1$ is the threshold of votes. If $p_t > 0.5$, then $P_t \to 1$ as $N \to +\infty$. If $p_f < 0.5$, then $P_f \to 0$. Therefore, in theory we can improve P_t and reduce P_f by adopting multiple sketches.

Multiple Sketches and Clustering (MSC) is summarized in Algorithm 1.

Algorithm 1: Multiple Sketches and Clustering (MSC).

Parameter setting
Input: tcpdump file
1: Parse raw data by *Tshark* to get multiple transactions and generate OFs.
2: **for** $n = 1$ to N; **do**
3: Aggregate OFs with same hash value under nth sketch in Section III-A to get ASFs.
4: **for** $c = 1$ to C; **do**
5: Apply alarm detection based on K-means++ clustering in Section IV-B1 on ASFs and get cth suspicious objects $alarmObj_n^c$.
6: **end for**
7: Merge C alarm set to one set $alarmObj_n$ by (3).
8: **end for**
9: Use voting strategy by (4) to get frequent anomalous objects $anomalyObj$
Output: definite anomalies $anomalyObj$

C. Feature Extraction of Anomalous Flows

As shown in Fig. 1, to extract associated features of anomalous flows, Apriori (introduced in Section III-C) is applied to flows of anomalies in each sliding time window to get not only the anomalous item-sets but also the anomalous rules within the multiple objects, which is based on the assumption that anomalies are essentially generated by similar features in network traffic data. Related definitions are given as follows.

- Abnormality Degree AD: the frequencies of packets containing an item-set, which is based on support of Apriori. For instance, the AD of two-item-set $\{IPsrc_a, IPdst_b\}$ is the packet frequencies including both $IPsrc_a$ and $IPdst_b$.

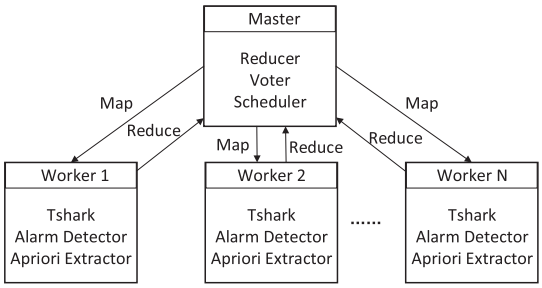


Fig. 4. Diagram of deployment architecture.

- Anomalous Item-set AI: an item-set of whose AD value is not less than the given threshold $minAD$, which indicates a group of anomalous features.
- Anomalous Rule AR: the rule describing the correlation between every two anomalous item-sets, whose confidence has its minimal confidence value $minConf$.

Specifically, when $P(IPsrc_a, IPdst_b) \geq minAD$, it indicates $IPsrc_a$ and $IPdst_b$ almost exist simultaneously. In this case, the reason causing anomaly is not single object, but the combination of multiple objects. If $P(B|A.) \geq minConf$, then $A \Rightarrow B$. The symbol "$\Rightarrow$" shows that the item-set on the right-hand-side of "$\Rightarrow$" is anomalous with the probability $P(B|A.)$, given that the item-set on the left-hand-side of "$\Rightarrow$" is anomalous. Greater confidence represents stronger correlation between the two item-sets.

The detailed procedures after MSC are as follows.

Step 1 anomalous flow generation. We filter flows in a time window and get the anomalous flows including detected anomalies. Each output flow includes five-tuple and packet number. Take IPsrc as an example, A set of anomalous IPsrcs is firstly detected, denoted as S_{IPsrc}, then the flow including any IPsrc in S_{IPsrc} is considered as anomalous flow AF.

Step 2 anomalous item-set generation. According to the recursive step in Apriori algorithm in Section III-C, we calculate the packet frequencies of all item-sets including 1-item-sets and k-item-sets, and any set whose abnormality degree larger than $minAD$ is selected as anomalous item-set AI.

Step 3 anomalous rule generation. The anomalous rule AR is derived from anomalous item-set AI. For each non-empty subset S of AI, if $P(AI)/P(S) \geq minConf$, then we get one anomalous rule: $S \Rightarrow AI - S$.

In summary, feature extraction of anomalous flows can conveniently access multiple features of anomalous flows and associations between multiple objects rather than an isolated anomalous object. For example, we need not detect object in set B if set A is anomalous and the confidence between these two sets exceeds $minConf$.

D. Deployment Strategy

As shown in Fig. 4, the overall deployment strategy draws on the idea of map-reduce. The master plays the role of reducer, voter and scheduler. Tshark, alarm detector and Apriori extractor are deployed on multiple workers. The results of multi-point workers are uniformly reduced and voted in the master. Meanwhile, the master is designed for scheduling among multiple workers in various phases.In anomaly detection stage, the distributed processing architecture is designed to ensure that MSC has a good detection efficiency when processing large-scale data. The sketch phase of alarm detection is the most time-consuming. In order to reduce the computational overhead caused by multiple sketches, the deployment strategy is as follows: 1) Make multiple copies of flows, and transfer these to workers. 2) Deploy alarm detection detection algorithm on each worker. 3) Integrate the results of multiple sketches running on the master, perform the voting operation, and obtain the final anomalies. In the feature extraction stage, the flows to be mined can be mapped into chunks and distributed to each worker. The local items and rules are mined, and then the global results are reduced on the master until all the frequent itemsets and association rules are obtained.

V. Numerical Experiments

Experimental setup is firstly presented and parameter selection experiments are carried out to determine the critical parameters of system. Next, MSC is compared with other related unsupervised and supervised algorithms. Moreover, the detection results under different network protocols, anomaly types, clustering algorithms, and input features are further investigated to demonstrate the validity of the method. Finally, anomalous feature combinations and association rules are mined and deeply analyzed to explore the anomalous root-cause.

A. Experimental Setup

Simulation setup. To build the MSC model and evaluate different detection algorithms, we considered AMD EPYC 7542 32-core Processor with 224 GB RAM workstation. In Flow Generation, transaction generator was pre-processed in Tshark, which is a network protocol analyzer, and discrete flow generator and time series generator were implemented in Python environment based on the perspective of map-reduce. In Anomaly Detection and Feature Extraction of Anomalous Flows, the algorithms were implemented in Python Scikit-learn, Scipy, Statsmodels, Numpy packages, etc. In particular, the supervised algorithms based on deep learning to compare with MSC were implemented in Python Pytorch package with GeForce RTX 2080 Ti with 64 GB RAM workstation.

Baseline methods. 1) Unsupervised baseline methods including MSC, Feature-MSPCA [15], Packet-MSPCA [25], KL-based [16], Hough-based [17] methods do not have training procedure. Experiments were conducted on both earlier (in 2007 from January to May) and later (in August 2019) MAWI-LAB traffic data to evaluate performance. Parameter setting was implemented in Receiver Operating Characteristic (ROC) curves by Control Variates on January 1st based on the assumption that optimal parameter are relatively stable during the same detection period and space. 2) Supervised baseline methods such as SVM-based [10], RF-based [21], RNN-based [22] and MLP-based [12] methods were implemented on dataset of August 2019. In our experiments, focal loss function [43] is adopted in deep learning about RNN-based and MLP-based

methods to solve uneven data problem. Note that the dataset was split into training and testing set in a ratio of 3:1.

Evaluation metrics. The metrics include Confusion Matrix, TPR, FPR, TNR and Accuracy, as discussed below:

- Confusion Matrix: a common approach for evaluating the performance of a detector. In our work, True Positive (TP) in matrix is the number of anomalous objects detected as anomalous, True Negative (TN) is the number of normal objects detected as normal, False Positive (FP) is the number of normal objects detected as anomalous, False Negative (FN) is the number of anomalous objects detected as normal.
- True Positive Rate (TPR)/ Recall: the proportion of true positives that are correctly detected as positive, $TPR = TP/(TP + FN)$.
- False Positive Rate (FPR): the proportion of true negatives that are incorrectly detected as positive, $FPR = FP/(FP + TN)$.
- True Negative Rate (TNR)/ Specificity: the proportion of true negatives that are correctly classified as negative, $TNR = TN/(FP + TN)$.
- Accuracy (ACC): a synthetic metric to measure both true positives and true negatives that are correctly detected, $ACC = (TP + TN)/(TP + FP + FN + TN)$.

B. Experimental Dataset and Ground Truth

We present the results of MAWILAB traffic [41] in 2007 from January to May, which include the daily backbone network packets consisting of 15-minutes-long traffic traces captured at 2 p.m., time in Japan. The performance hold even if we choose traffic traces from other time periods. In particular, the average size of daily packet file is about 500 MB including about 6 million traffic packets, which contain various packet information about TCP/IP layer, such as IPsrc, IPdst, Portsrc, Portdst, packet size and flags (SYN, ACK, FIN, etc).

MAWILAB provides labeled traffic data, which are classified to three categories including "normal," "anomalous," and "suspicious" by using the combination method [18], where the object labeled as anomalous has a higher abnormal degree compared with suspicious. Therefore, to ensure the reliability of the results, we evaluate the proposed method according to the accuracy of identifying the objects labeled as "anomalous". Furthermore, MAWILAB uses a mature multi-classification criterion [1] to manually calibrate the anomalies and obtain various types of anomaly such as Port scan, Alpha flow, HTTP, Multi points, etc.

A stacked figure (Fig. 5) depicts the detailed packet frequencies information of definite anomalies under different protocols in January 2007 by MAWILAB. We can clearly observe that most of the anomalies emerged in the process of TCP transmission, accounting for 93.59% of the anomalies per day in average. ICMP (ping attack) anomalies primarily occurred in the first half of January. A large number of UDP traffic anomalies occurred on 21 January. DNS anomalies mainly took place on the January 4, 5, 10, 13, 15, and 16. This is due to IPV4 exception in network layer, which is drawn by observing the true data.

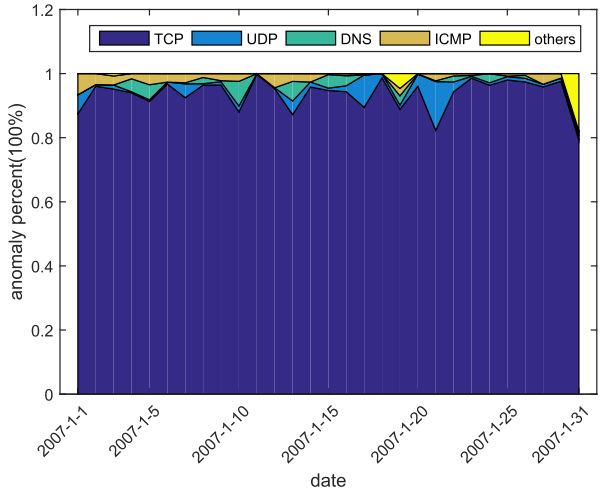


Fig. 5. Anomaly classification based on protocol in January 2007.

C. Parameter Selection

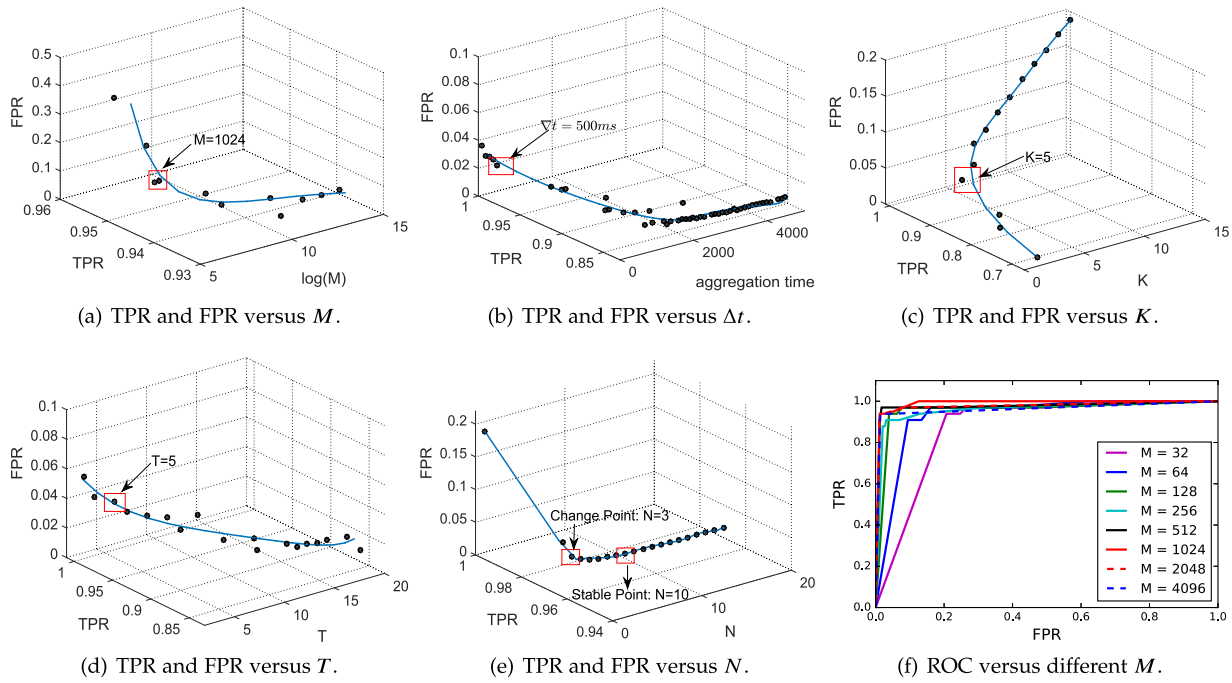
To determine the critical parameters, we carry out a large number of experiments based on variable control by changing one single parameter and controlling the others every time and further elaborating the detection results. Note that we only present the results of parameter setting on January 1st, since optimal parameters are relatively stable during the same period.

1) Hash Table Size: To study the impact of the size of hash table on the detection results, Fig. 6(a) depicts the curve of TPR and FPR against the logarithm of $M = 2^i, i \in [5, 15]$, when fixing other parameters $\Delta t = 500\ ms, T = 5, N = 10, L = 5, K = 5$. We can observe that M has little effect on TPR and has a significant effect on FPR. With the increase of M, FPR is gradually decreasing, and $M = 1024$ is the transition point. The reason is that when M is small, the same SON is used to aggregate excessive objects, resulting in an increase of FPR. Intuitively, an overlarge hash table size directly leads to an increase of the number of samples for clustering, and therefore, enhance the complexity of the algorithm. Thus, to balance between the FPR and the complexity of the algorithm, an optimal hash table size is set to $M = 1024$.

Fig. 6(f) is the ROC curve when changing voting times for different M, where the area under the ROC curve (AUC) was used to compare the detection results. Larger AUC can represent better performance of a certain M. The maximum AUC 0.9974 locates at $M = 1024$, and therefore we can also conclude that applying $M = 1024$ has better detection performance.

2) Time Scale: To investigate the impact of time scale on the detection results, Fig. 6(b) depicts the curve of TPR and FPR with time scale $\Delta t \in [100, 5000) ms$ by an increment of 100 ms, when controlling other parameters as $M = 1024$, $T = 5, N = 10, L = 5, K = 5$.

With the increase of Δt, both TPR and FPR gradually decrease and eventually tend to be stable, and the variation



(a) TPR and FPR versus M. (b) TPR and FPR versus Δt. (c) TPR and FPR versus K.

(d) TPR and FPR versus T. (e) TPR and FPR versus N. (f) ROC versus different M.

Fig. 6. Experiments about all parameters setting. Subplots (a-e) respectively correspond to the procedure of selecting appropriate M, Δt, K, T, and N. Subplot (f) presents the ROC curve under different M, which shows another validation method to determine the optimal parameters.

range of FPR is smaller than that of TPR. The reason is that when time scale becomes larger, sudden anomalous flows are covered up by taking summation within a large time scale, which naturally leads to the reduction of TPR. Taking both real-time and accuracy into account, we choose $\Delta t = 500\ ms$ in our experiment.

3) Cluster Number: To study the impact of the number of clusters on detection results, Fig. 6(c) depicts the curve of TPR and FPR with cluster number $K \in [2, 15]$, where $\Delta t = 500\ ms, T = 5, N = 10, L = 5, M = 1024$. We can observe that TPR and FPR become larger with increase of K. Specifically, the flows in the largest cluster decrease as the number of clusters increases, which will cause an increase in the number of anomalous flows detected, thus enlarging both TPR and FPR. We select the transition point $K = 5$ in our experiment considering TPR and FPR.

4) Sliding Time Window: To study the impact of sliding time window on detection results, Fig. 6(d) shows the curve of TPR and FPR with $T \in [3, 20]$, when we fix other parameters to be $\Delta t = 500\ ms, K = 5, N = 10, L = 5, M = 1024$.

It is obvious that when T increases, TPR decreases rapidly, and FPR decreases slowly. The reason is that excessive sliding time window will cover up the characteristics of sudden anomalies so that they cannot be properly identified, and thus TPR reduces with the increase of T. Indeed, the appropriate sliding time window is required to ensure that a real-time detection has high TPR and low FPR. We finally choose $T = 5$ in our experiment considering real-time, TPR and FPR.

5) Number of Random Projections: To study the impact of sketch number on the detection results, Fig. 6(e) depicts the curve of TPR and FPR with $N \in [1, 20]$ when fixing the parameters $\Delta t = 500\ ms, K = 5, M = 1024, T = 5$. In this part, we set $L = N$ to eliminate the influence of L on N. It can be observed that TPR is almost independent on N, and FPR decreases gradually with the decrease of N, where $N = 3$ is the transition point, and the value of FPR becomes stable when $N \geq 10$. Thus, $N = 10$ is determined to balance TPR and FPR, and voting times $L = 5$ is set to ensure a low FPR while keeping TPR relatively stable.

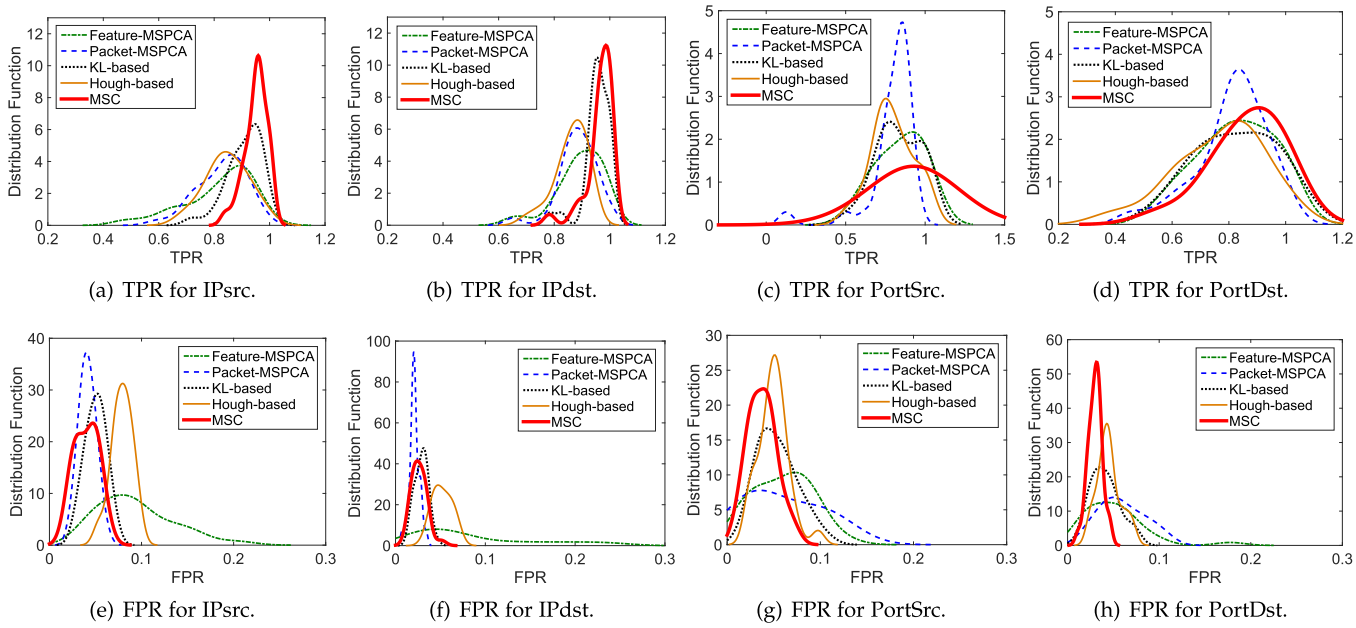
D. Performance Comparison

We compare the performance of the unsupervised detection against different previously used unsupervised detection techniques in MAWILAB [15] including related PCA-based detectors [15], [15] (note that Feature-MSPCA [15] and Packet-MSPCA [25] as improved PCA approaches represent two types features in input feature), KL-divergence Histogram with association rules [16], and Adaptive Hough-transform [17].

First, TPR and FPR distribution and scatter are presented to verify the effectiveness of the proposed method. Second, protocol classification and anomaly classification experiments are conducted to further explore the performance under multiple fine-grained levels. Finally, we compare Packet-MSC with Entropy-MSC to obtain the relatively proper input feature.

1) TPR and FPR Distribution: To have a comprehensive comparison, we further use kernel density estimation (KDE)[1] to fit the distributions of the resultant TPR and FPR from the detection of IPsrc, IPdst, PortSrc and PortDst on the January data. The probability density functions (PDFs) are shown in

[1] Kernel density estimation (KDE) is a non-parametric way to estimate the probability density function of a random variable. Compared with other parametric ways, it cannot mask too much useful information when fitting the probability density.



(a) TPR for IPsrc. (b) TPR for IPdst. (c) TPR for PortSrc. (d) TPR for PortDst.

(e) FPR for IPsrc. (f) FPR for IPdst. (g) FPR for PortSrc. (h) FPR for PortDst.

Fig. 7. Comparison of the detection results for IP and Port using different algorithms in January 2007. Subplots in the first row present the TPR distributions and those in the second row show the FPR distributions. From left to right, various columns respectively correspond to the comparative results on IPsrc, IPdst, PortSrc, and PortDst.

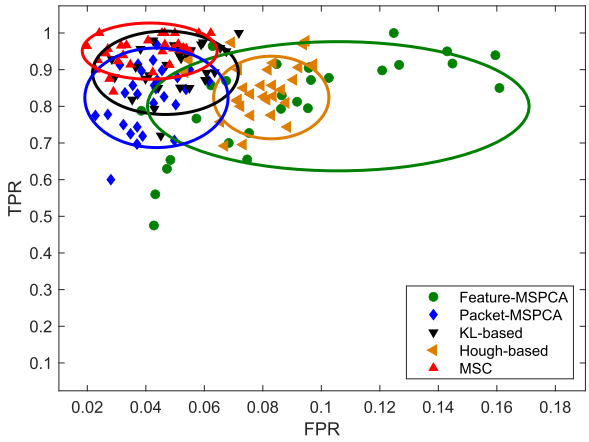


Fig. 8. Scatter figure of TPR and FPR for IPsrc in January 2007. The points of each shape represent the results of the corresponding method for IPsrc within one month, and the ellipses depict distributions of the points.

Fig. 7. We can obtain that TPR and FPR of MSC are more likely to reveal better performance than other methods.

2) TPR and FPR Scatter: To compare MSC with other methods in terms of daily detection results, a scatter figure of TPR vs. FPR is presented in Fig. 8. For MSC, the TPR is between 0.8 to 1.0, and the FPR is about 0.02. The points are mainly concentrated at the most upper left corner of the figure, which is in an ellipse with the smallest area compared with those of the other methods, indicating that the performance of MSC is the most stable.

3) Protocol Classification: To compare the detection performance of different methods under different protocols, Table II shows anomaly frequency of the detection on January data using different protocols and detection methods. Obviously, the frequency of the proposed method is superior to the others in

TABLE II
DETECTION RESULTS UNDER DIFFERENT PROTOCOLS

Frequency / Protocol / Method	TCP	UDP	DNS	ICMP	others
Feature-MSPCA	90818372	2310770	999041	1426582	410036
Packet-MSPCA	76189519	2274946	1438585	1483124	1209425
KL-based	92001361	2345867	1438356	1519960	1209541
Hough-based	90260331	2297848	1308503	1527058	1209607
MSC	92067681	2350125	1438090	1530682	1210199

the detection of TCP, UDP, ICMP, and others, except for the fact that the anomaly frequency of MSC (1438090) is slightly lower than KL-based method (1438356) and Packet-MSPCA (1438585) under DNS. For the two methods based on MSPCA, Feature-MSPCA is superior to Packet-MSPCA under main communication transport layer protocols (TCP and UDP). According to those results, network operators can select the appropriate approach to monitor communication under specific protocol.

4) Anomaly Classification: This part aims at showing the detection performance in terms of various anomaly types. MAWILAB employs a anomaly classification technique for a better understanding of identified anomalies in work [1] and URL [41], where they intuitively provide a detailed taxonomy of backbone traffic anomalies based on protocol headers and connection patterns. The anomalies in MAWILAB webpage [41] are with respect to Alpha flow, HTTP, Multi points, Network scan TCP, Network scan UDP, Network scan ICMP, etc. For instance, anomalous HTTP is relevant to traffic on port 80/TCP, with SYN flag $\leq 30\%$. In our experiments, we quoted the results of anomaly classification in MAWILAB webpage and made a detailed statistical analysis for accumulating the number of anomalous IPsrcs on dataset of one month to demonstrate the classification results of different methods on various anomalous types clearly.

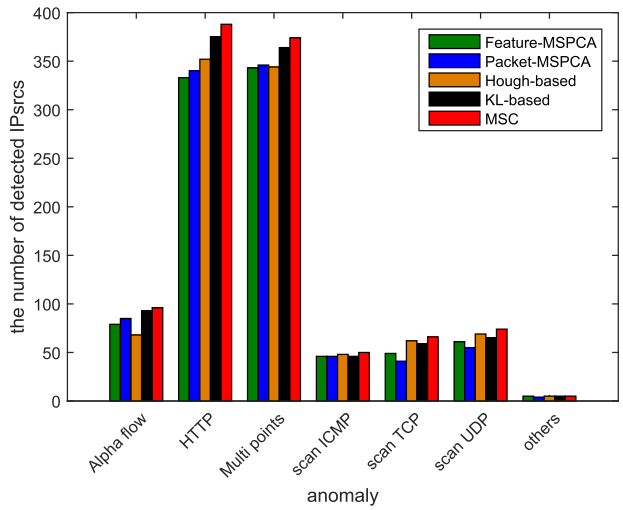
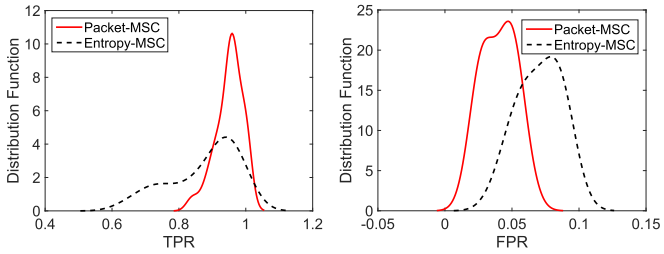


Fig. 9. Comparison of the detected anomalies classification results in January 2007. The horizontal axis depicts the six labeled types of anomalies based on MAWILAB's taxonomy [41]. The vertical axis presents the number of detected IPsrcs under different algorithms.



(a) TPR in January. (b) FPR in January.

Fig. 10. Comparison of detection results for IPsrc using Packet and Entropy features in January 2007. Subplots (a) and (b) respectively correspond to the TPR distribution and FPR distribution under the two different features.

The results in Fig. 9 depict the numbers of detected anomalous IPsrcs of the January data which are correctly judged by each detector. It shows that anomalies labeled as HTTP or Multi points account for the vast majority of all detected anomalies. MSC has better performance than the other methods, since it can detect a wide range of anomalies and obtain results close to the MAWILAB true labeled data. To sum up, the number of total detected anomalies ranks as follows: MSC (1053) $>$ KL-based (1007) $>$ Hough-based (948) $>$ Feature-MSPCA (917) $>$ Packet-MSPCA (916).

5) Packet-MSC vs. Entropy-MSC: In this part, we utilize different input features (Packet and Entropy) in the MSC method, yielding two corresponding detection methods Packet-MSC and Entropy-MSC, whose effectiveness is compared in Fig. 10. Packet-MSC adopts packet number as the feature of flows, whereas Entropy-MSC applies the entropy of the packet number distribution of all objects in each SON.Fig. 10 presents that Packet-MSC has higher stability, better TPR and FPR distribution in comparison with Entropy-MSC.

6) Time Cost Comparision: As Table III, we evaluate different detection algorithms under AMD EPYC 7542 32-core Processor with 224 GB RAM workstation. In actual experimental scenarios, Feature-MSPCA consume maximum time

TABLE III
TIME COST WITHIN MSC AND OTHER METHODS

Time cost(s) \ Object / Method	IPsrc	IPdst	PortSrc	PortDst
Feature-MSPCA	3157.40	3733.60	2103.20	2109.19
Packet-MSPCA	945.50	955.70	808.15	815.15
KL-based	1025.20	1172.00	796.70	800.00
Hough-based	1861.50	1861.50	1051.60	1051.60
MSC	1027.80	1211.80	824.80	825.90

TABLE IV
COMPLEXITY WITHIN MSC AND OTHER METHODS

Method	Complexity
Feature-MSPCA	$O\left(\frac{T_{all}}{T}N\left(US+\left(T^2M+T^3\right)(L+2)\right)\right)$
Packet-MSPCA	$O\left(\frac{T_{all}}{T}N\left(S+\left(T^2M+T^3\right)(L+2)\right)\right)$
KL-based	$O\left(N\left(\frac{T_{all}}{T}S+M(\frac{T_{all}}{T}-1)\right)\right)$
Hough-based	$O(\frac{T_{all}}{T}N\left(S_{ipsrc}+S_{ipdst}+3M^2\right)/2)$
MSC	$O(\frac{T_{all}}{T}N(S+MTKi))$

due to its complex 21-dimension input feature extraction compared with a single packet feature. Due to the sliding time window and sketch strategy, the input data of clustering algorithm in the detection step of MSC is a $M \times T$ small-scale matrix. Thus, the time cost of MSC only performs slightly worse than KL-based, Packet-MSPCA. Hough-based method costs more time as it iterates over all attribute pairs such as IP pairs(IPsrc, IPdst) and Port pairs (PortSrc, PortDst).

Table IV presents overall time complexity of different detection methods, where each formula consists of two main parts corresponding to sketch and core detection. Specifically, S is the number of object flows, and the complexity of a sketch is $O(S)$ in general. For MSC, the complexity of single Kmeans++ is $O(MTKi)$, where M is the number of samples after sketching, T is the number of features in one clustering detection and the length of sliding time window, K is the number of clusters, and i is the number of iterations. T_{all} is the time span, and N is the number of sketch. For MSPCA-based method, the sketch complexity of Feature-MSPCA is U times that of Packet-MSPCA since Feature-MSPCA holds rich 21-dimension features. MSPCA contains $(L+2)$ PCAs, where $L = \log_2 M - 5$. For KL-based method, it focuses on calculating the distance within multiple neighboring histogram distributions, and needs $(T_{all}/T-1)$ times distance calculation under single sketch. For Hough-based method, its complexity is proportional to $3M^2$ in our experiment. Complexity dividing by 2 denotes average value for single IP object, as it can detect IPsrc and IPdst together. In above methods, sketch costs vast time since $S \gg M, S \gg T$. Meanwhile, $M \gg T, M \gg K, M \gg L$ in concrete experiments. Thus, Feature-MSPCA followed by Hough-based method holds highest complexity, and MSC differs little from two other methods.

E. Comparison Under Different Clustering Algorithms

Table V depicts the TPR and FPR of IPsrc based on multiple clustering algorithms together with MSC on the January 1st, 2nd and 3 rd data. The corresponding ROC curves are depicted in Fig. 11.

TABLE V
TPR AND FPR WITHIN CLUSTERING ALGORITHMS

	20070101		20070102		20070103	
	TPR	FPR	TPR	FPR	TPR	FPR
K-means++-MSC	**1.000**	**0.024**	**0.900**	**0.023**	**0.939**	**0.032**
Birch-MSC	0.970	0.027	0.900	0.023	0.910	0.038
Agglomerative-MSC	1.000	0.032	0.900	0.033	0.939	0.042
MeanShift-MSC	1.000	0.045	0.900	0.029	0.939	0.033
AP-MSC	1.000	0.123	0.875	0.112	0.848	0.109
DP-MSC	1.000	0.151	0.900	0.100	0.909	0.152

We find that compared with K-means++, hierarchical clustering (Birch and Agglomarative) displays slightly worse in ROC characteristic. It should be mentioned that complicated algorithms for automatic discovery of cluster centers such as AP (Affinity Propagation) and DP (Density Peak) are not applicable to MSC framework, because they generally work for separating normal samples into multiple clusters accurately and are not sensitive to anomalous data, which leads to a mix of anomalous and normal data, and further triggers a higher FPR. MeanShift has higher algorithm complexity and the slower convergence than K-means++. Considering the detection performance and computation cost, K-means++ is relatively suitable for MSC.

F. Detected Results on Other Time Periods

To verify the universality of the proposed method, we perform experiments by the five unsupervised methods on dataset from February to May in 2007 and recent dataset in August 2019. Fig. 12 shows the detection results of IPsrc. Obviously, the TPR distribution of MSC locating at the most right is superior than those of the others in February, March and May, while its FPR distribution also performs the best. Feature-MSPCA has the same performance as MSC in April, but its FPR distribution with the largest variance is extremely unstable among all methods. Fig. 13 displays the detection performance of IPdst. It presents that MSC also has obvious advantage in TPR distribution over the others on the March data, and its FPR distribution which locates at the most left performs the best. Fig. 14 displays the overall TPR, FPR, TNR and Accuracy distributions of IPsrc in August 2019. It presents that MSC which locates at the most right in TPR, TNR, and Accuracy and the most left in FPR also performs better on recent dataset. The overall evaluation is introduced when it is difficult to compare the performance merely by TPR and FPR. Table VI shows synthetic performance on recent dataset in August 2019. MSC is optimal in IPsrc, IPdst and PortSrc for the comprehensive metric Accuracy. Meanwhile, MSC is superior to Hough-based method in TPR, although it is slightly worse than Hough-based method in FPR and Accuracy. It also shows that MSC has limitations for dramatic uneven data distribution problem on port. As a whole, MSC outperforms other reference unsupervised detection methods.

G. Detected Results on MSC and Supervised Methods

To assess the gap between MSC and the supervised methods, data in August 2019 is randomly split into training and test data in a ratio of 3:1. Meanwhile, to alleviate the challenge about uneven data, the ratio of normal and abnormal data is adjusted to 100:1 in training phase. As shown in Fig. 15, SVM-based, RF-based, MLP-based, RNN-based methods and MSC are compared in test data. The detection results of IPsrc demonstrate that compared with RNN-based method, MSC performs better in FPR, TNR and Accuracy on condition that it has similar effect on TPR. On the other hand, compared with SVM-based, RF-based and MLP-based methods, MSC exhibits better performance in TPR, but it is worse in FPR, TNR and Accuracy. In overall, supervised methods which need a complete training database to support various anomaly types are severely influenced by the quality of training data, and updating the database is always difficult and consuming. MSC can yield excellent performance in TPR, although it is slightly worse in other metrics. It can detect unknown anomalies and do not rely on training data due to its characteristic of unsupervised learning.

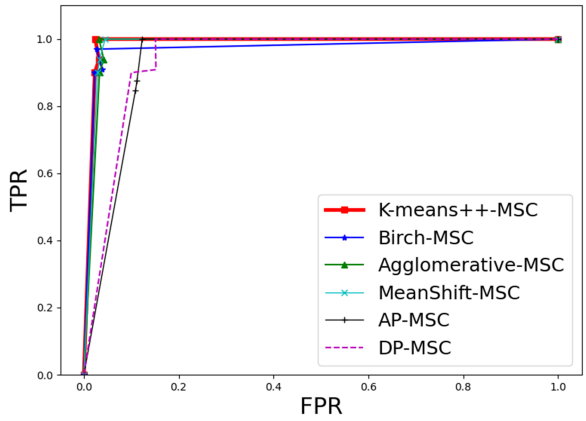
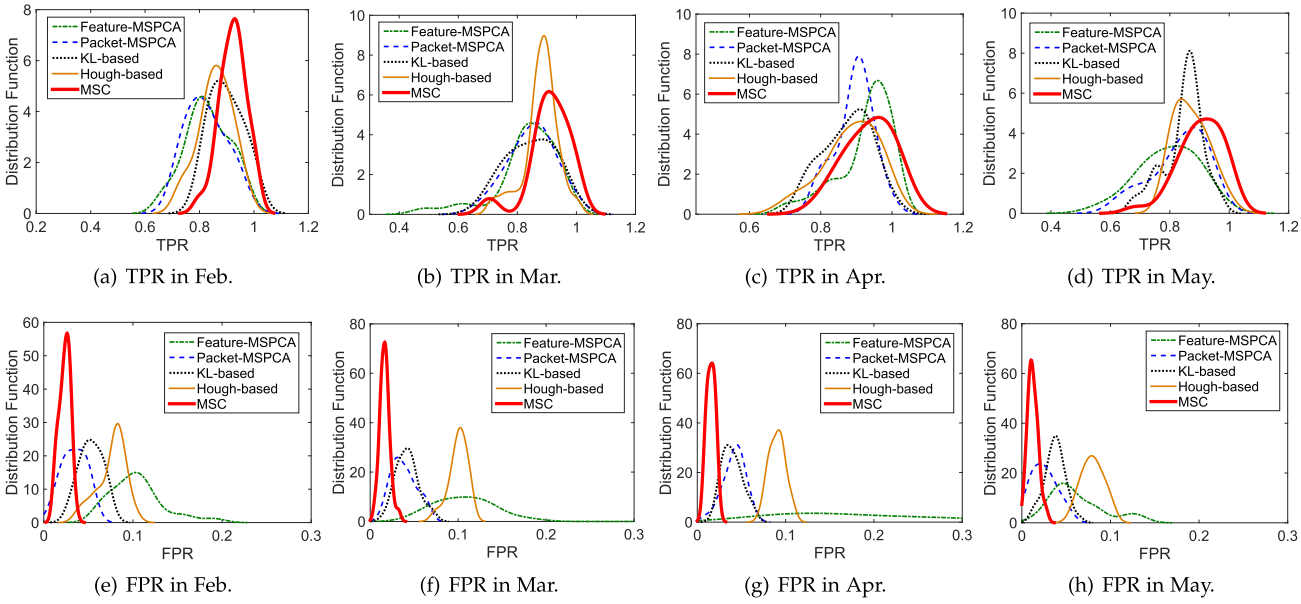


Fig. 11. ROC curves under different Clustering Algorithms. The K-means++-MSC ROC characteristic depicted by red curve that locates in the upper left corner having the maximum AUC presents best performance.

H. Feature Extraction Results of Anomalous Flows

As shown in Fig. 1, feature extraction of the anomalous flows allows us to filter out the flows of anomalies and obtain the anomalous item-sets and anomalous rules of multiple anomalous objects.

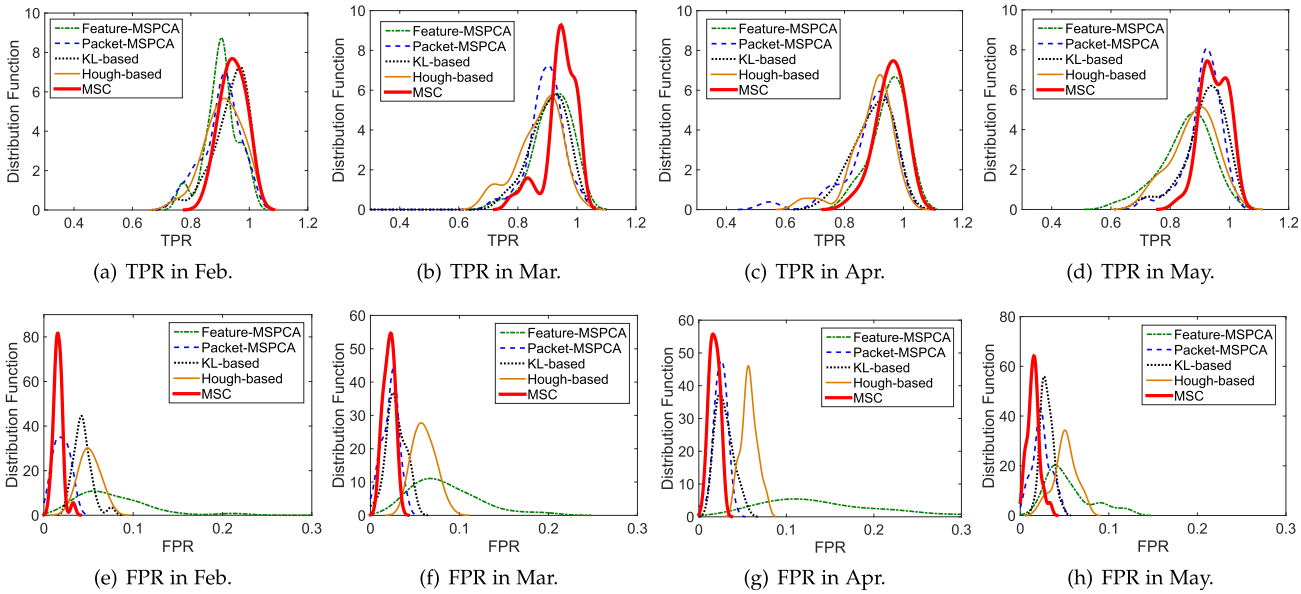
For example, we perform experiment on the flows of anomalous IPsrcs to extract feature combinations (anomalous item-sets) precisely in the first sliding window. The threshold of AD indicating that the frequency of packets in the corresponding sliding window is set to 500. As shown in Table VII, the AD of TCP protocol is 4498, which indicates that the number of TCP packets associated with anomalous packets is 4498. The ADs of DNS, ICMP, and UDP are 1116, 756, and 546, respectively. We can conclude that the main anomalous protocol is TCP in general in this sliding time window. TCP and PortSrc 80 constitute a two-item-set of which the AD is 2012, denoting that when the protocol is TCP and PortSrc is 80, the frequency of anomaly occurrence is higher than other combinations of protocol and port. IPsrc 164.89.55.232, protocol TCP, and PortSrc 80 contribute to a three-item-set of which the AD is 1448. In this case, when network operators detect the anomalous IPsrc 164.89.55.232 online, they can infer



(a) TPR in Feb. (b) TPR in Mar. (c) TPR in Apr. (d) TPR in May.

(e) FPR in Feb. (f) FPR in Mar. (g) FPR in Apr. (h) FPR in May.

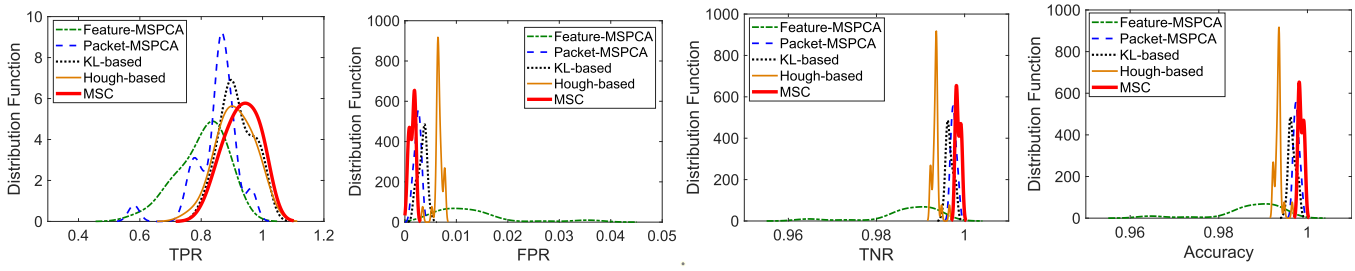
Fig. 12. Comparison of the detection results for IPsrc by different algorithms from February to May in 2007.



(a) TPR in Feb. (b) TPR in Mar. (c) TPR in Apr. (d) TPR in May.

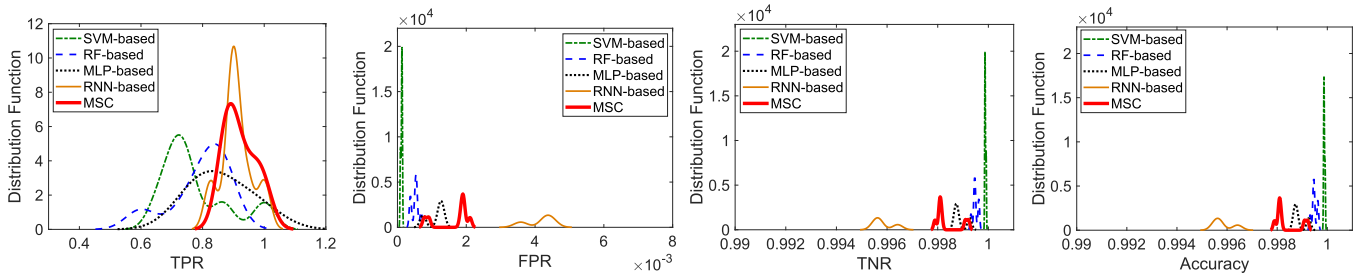
(e) FPR in Feb. (f) FPR in Mar. (g) FPR in Apr. (h) FPR in May.

Fig. 13. Comparison of the detection results for IPdst by different algorithms from February to May in 2007. Subplots in the first row present the TPR distributions and those in the second row show the FPR distributions. From left to right, various columns respectively correspond to the comparative results through February to May.



(a) TPR for IPsrc. (b) FPR for IPsrc. (c) TNR for IPsrc. (d) Accuracy for IPsrc.

Fig. 14. Comparison of the detection results for IPsrc using different algorithms in August 2019. Subplots from left to right present the TPR, FPR, TNR, and Accucacy distributions.



(a) TPR for IPsrc. (b) FPR for IPsrc. (c) TNR for IPsrc. (d) Accuracy for IPsrc.

Fig. 15. Comparison of the detection results for IPsrc using supervised algorithms and MSC in August 2019. Subplots from left to right present the TPR, FPR, TNR, and Accucacy distributions.

TABLE VI
OVERALL PERFORMANCE EVALUATION

Object	Method	TPR	FPR	TNR	Accuracy
IPsrc	Feature-MSPCA	0.8078	0.0130	0.9870	0.9870
	Packet-MSPCA	0.8545	0.0024	0.9976	0.9976
	KL-based	0.9219	0.0034	0.9966	0.9966
	Hough-based	0.9172	0.0066	0.9934	0.9934
	MSC	**0.9319**	**0.0014**	**0.9986**	**0.9986**
IPdst	Feature-MSPCA	0.8847	0.0094	0.9906	0.9906
	Packet-MSPCA	0.8572	0.0023	0.9977	0.9977
	KL-based	0.9515	0.0070	0.9930	0.9930
	Hough-based	0.9465	0.0040	0.9960	0.9960
	MSC	**0.9386**	**0.0003**	**0.9997**	**0.9997**
PortSrc	Feature-MSPCA –	–	–	–	–
	Packet-MSPCA	0.6938	0.1733	0.8267	0.8267
	KL-based	0.5867	0.0347	0.9653	0.9652
	Hough-based	0.6301	0.0330	0.9668	0.9667
	MSC	**0.8117**	**0.0292**	**0.9708**	**0.9708**
PortDst	Feature-MSPCA	–	–	–	–
	Packet-MSPCA	0.8982	0.3306	0.6694	0.6695
	KL-based	0.8653	0.1058	0.8942	0.8942
	Hough-based	0.7090	0.0144	0.9856	0.9855
	MSC	**0.7612**	**0.0481**	**0.9519**	**0.9519**

TABLE VII
ANOMALOUS ITEM-SETS IN THE FIRST WINDOW (0-2500 MS) ON JANUARY 1ST

IPsrc	IPdst	protocol	PortSrc	Portdst	*AD*
-	-	TCP	-	-	4498
-	-	TCP	80	-	2012
164.89.55.232	-	TCP	-	-	1542
164.89.55.232	-	TCP	80	-	1448
-	-	DNS	-	-	1116
-	-	ICMP	-	-	756
-	149.173.188.128	TCP	-	6881	704
-	215.37.121.18	TCP	-	-	583
214.138.179.238	-	TCP	-	-	557
-	-	UDP	-	-	546
133.208.154.172	-	TCP	19101	-	519
133.208.25.108	-	TCP	-	-	511

more rapidly that the anomalous protocol is TCP and the anomalous PortSrc is 80 with a larger AD.

Furthermore, anomalous rules are generated based on the above anomalous item-sets under the threshold of confidence $minConf$ 0.7. In Table VIII, for two frequent item-sets, $S = \{sport19101, sip133.208.154.172\}$ and $L = \{TCP\}$, we can obtain the confidence $P(L/S) = P(LS)/P(S) = 1$ of item-set L relative to item-set S, where $P(LS)$ represents the AD of the item-set L and S, and $P(S)$ is the AD for the item S. Therefore, we can induce a rule that when IPsrc 133.208.154.172 and PortSrc 19101 are identified as anomalous objects, the protocol of anomalous flow must be TCP($Conf = 1$). If IPdst is 149.173.188.128, we can conclude that the

TABLE VIII
ANOMALOUS RULES IN THE FIRST WINDOW (0-2500 MS) ON JANUARY 1ST

Item-Set 1⇒	Item-Set 2	Confidence
sport19101,sip133.208.154.172	TCP	1
sip164.89.55.232, sport80	TCP	1
sip133.208.25.108	TCP	1
sip214.138.179.238	TCP	1
dip149.173.188.128, dport6881	TCP	1
dip149.173.188.128	TCP, dport6881	0.981
sip164.89.55.232, TCP	sport80	0.939
sport80	sip164.89.55.232	0.720

associated anomalous protocol is TCP and the Portdst is 6881 with the probability 98.1%. If PortSrc is 80, with a probability of 72%, the associated anomalous IPsrc is judged as 164.89.55.232. Similarly, we generate other rules as Table VIII.

VI. CONCLUSION

Anomaly detection plays an important role in network security. In this paper, we introduce an improved unsupervised clustering detection system that incorporates multiple random projections, K-means++ clustering, and association rules mining.

The system which exploits K-means++ unsupervised clustering allows us to detect known and unknown anomalies without using any priori information and artificial analysis. In addition, multiple random projections (sketches) and voting strategy can ensure a high TPR while producing few false positives. Mining frequent item-sets and association rules on anomalous flows by Apriori algorithm can extract and summarize anomalous features precisely. Furthermore, the proposed anomaly detection scheme is compared with other existing methods in KDE distribution of TPR and FPR for different objects including IPsrc, IPdst, PortSrc, PortDst, TPR-FPR scatter distribution, protocol classification, and anomaly classification. It is demonstrated that the proposed method outperforms benchmarks in the experiments.

There are still a number of possible directions for our future relevant study. For example, useful information (SYN and ACK flags) of packets not only relying on the IP address and

port information could be utilized, the scalability and efficiency of the association rule mining algorithm could be improved to handle larger network traffic data, and distributed parallel processing system such as Apache Hadoop or Apache Spark engines could be introduced to boost the efficiency of data processing.

REFERENCES

[1] J. Mazel, R. Fontugne, and K. Fukuda, "A taxonomy of anomalies in backbone network traffic," in *Proc. 5th Int. Workshop TRaffic Anal. Characterization*, 2014, pp. 30–36.

[2] G. Dewaele, Y. Himura, P. Borgnat, K. Fukuda, and P. Abry, "Extracting hidden anomalies using sketch and non gaussian multiresolution statistical detection procedures," in *Proc. Workshop Large Scale Attack Defense*, 2007, pp. 145–152.

[3] R. Fontugne, P. Abry, K. Fukuda, and P. Borgnat, "Random projection and multiscale wavelet leader based anomaly detection and address identification in internet traffic," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2015, pp. 5530–5534.

[4] D. Jiang, L. Huo, and H. Song, "Rethinking behaviors and activities of base stations in mobile cellular networks based on Big Data analysis," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 1, pp. 80–90, Jan.–Mar. 2020.

[5] Y. X. Wan, K. Xu, F. Wang, and G. L. Xue, "Characterizing and mining traffic patterns of iot devices in edge networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 1, pp. 89–101, Jan.–Mar. 2021.

[6] G. Tian et al., "TADOOP: Mining network traffic anomalies with Hadoop," in *Proc. Int. Conf. Secur. Privacy Commun. Syst.*, 2015, pp. 175–192.

[7] J. Mazel, P. Casas, R. Fontugne, K. Fukuda, and P. Owezarski, "Hunting attacks in the dark: Clustering and correlation analysis for unsupervised anomaly detection," *Int. J. Netw. Manage.*, vol. 25, no. 5, pp. 283–305, 2015.

[8] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, and K.-K. R. Choo, "Blockchain-enabled authentication handover with efficient privacy protection in SDN-based 5G networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 1120–1132, Apr.–Jun., 2021.

[9] Y. Li and L. Guo, "A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems," *Expert Syst. Appl*, vol. 42, no. 5, pp. 2670–2679, 2015.

[10] W. Feng, Q. Zhang, G. Hu, and J. Huang, "Mining network data for intrusion detection through combining SVMs with ant colony networks," *Future Gener. Comput. Syst*, vol. 37, pp. 127–140, 2014.

[11] A. Shahraki, M. Abbasi, and Y. Haugen, "Boosting algorithms for network intrusion detection: A comparative evaluation of real adaboost, gentle adaboost and modest adaboost," *Eng. Appl. Artif. Intell.*, vol. 94, 2020, Art. no. 103770.

[12] J. Vanerio and P. Casas, "Ensemble-learning approaches for network security and anomaly detection," in *Proc. Workshop Big Data Anal. Mach. Learn. Data Commun. Netw.*, 2017, pp. 1–6.

[13] H. W. Du, Q. Ye, Z. P. Sun, C. Liu, and W. Xu, "Fast-ODT: A lightweight outlier detection scheme for categorical data sets," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 1, pp. 13–24, Jan.–Mar. 2021.

[14] C. D. Xuan, H. Thanh, and N. T. Lam, "Optimization of network traffic anomaly detection using machine learning," *Int. J. Elect. Comput. Eng.*, vol. 11, pp. 2360–2370, 2021.

[15] Z. Chen, K. Chai, B. Lee, and C. Lau, "A novel anomaly detection system using feature-based MSPCA with sketch," in *Proc. Wireless Opt. Commun. Conf.*, 2017, pp. 1–6.

[16] D. Brauckhoff, X. Dimitropoulos, A. Wagner, and K. Salamatian, "Anomaly extraction in backbone networks using association rules," *IEEE/ACM Trans. Netw.*, vol. 20, no. 6, pp. 1788–1799, Dec. 2012.

[17] R. Fontugne and K. Fukuda, "A hough-transform-based anomaly detector with an adaptive time interval," *ACM SIGAPP Appl. Comput. Rev.*, vol. 11, no. 3, pp. 41–51, 2011.

[18] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda, "Mawilab: Combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking," in *Proc. ACM Conf. Emerg. Netw. Experiments Technol.*, 2010, pp. 1–12.

[19] S. Mabu, S. Gotoh, M. Obayashi, and T. Kuremoto, "A random-forests-based classifier using class association rules and its application to an intrusion detection system," *Artif. Life Robot.*, vol. 21, no. 3, pp. 371–377, 2016.

[20] Y. Liu and Y. Gu, "A novel backbone network anomaly detector via clustering in sketch space," in *Proc. IEEE Data Sci. Workshop*, 2018, pp. 31–35.

[21] D. Yao, M. Yin, J. Luo, and S. Zhang, "Network anomaly detection using random forests and entropy of traffic features," in *Proc. Int. Conf. Multimedia Inf. Netw. Secur.*, 2013, pp. 926–929.

[22] I. S. Thaseen, A. K. Chitturi, and e. a. F. Al-Turjman, "An intelligent ensemble of long-short-term memory with genetic algorithm for network anomaly identification," *Trans. Emerg. Telecommun. Technol.*, 2020, Art. no. e4149.

[23] X. Li et al., "Detection and identification of network anomalies using sketch subspaces," in *Proc. ACM SIGCOMM Conf. Internet Meas.*, 2006, pp. 147–152.

[24] H. Ringberg, A. Soule, J. Rexford, and C. Diot, "Sensitivity of PCA for traffic anomaly detection," in *ACM SIGMETRICS Int. Conf. Meas. Model. Comput. Syst.*, San Diego, CA, USA, 2007, pp. 109–120.

[25] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau, "Detection of network anomalies using improved-MSPCA with sketches," *Comput. Secur.*, vol. 65, pp. 314–328, 2017.

[26] C. Callegari, S. Giordano, and M. Pagano, "An information-theoretic method for the detection of anomalies in network traffic," *Comput. Secur.*, vol. 70, pp. 351–365, 2017.

[27] A. Kind, M. P. Stoecklin, and X. Dimitropoulos, "Histogram-based traffic anomaly detection," *IEEE Trans. Netw. Service Manag.*, vol. 6, no. 2, pp. 110–121, Jun. 2009.

[28] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," in *Proc. ACM SIGCOMM Comput. Commun. Rev.*, 2005, pp. 217–228.

[29] W. A. Shewhart and S. S. Wilks, *Cluster Analysis*, 5th ed., London, UK: King's College London, 2011.

[30] D. Arthur and S. Vassilvitskii, "k-means :The advantages of careful seeding," in *Proc. 18th ACM-SIAM Symp. Discrete Algorithms*, 2007, pp. 1027–1035.

[31] S. Zanero and S. Savaresi, "Unsupervised learning techniques for an intrusion detection system," in *Proc. ACM Symp. Appl. Comput.*, 2003, pp. 412–419.

[32] K. Leung and C. Leckie, "Unsupervised anomaly detection in network intrusion detection using clusters," in *Proc. 28th Australas. Comput. Sci. Conf. Comput. Sci.*, 2005, pp. 333–342.

[33] J. Mazel, P. Casas, Y. Labit, and P. Owezarski, "Sub-space clustering, inter-clustering results association and anomaly correlation for unsupervised network anomaly detection," in *Proc. Int. Conf. Netw. Serv. Manage.*, 2011, pp. 73–80.

[34] M. Bhuyan, K. Bhattacharyya, and J. Kalita, "Towards an unsupervised method for network anomaly detection in large datasets," *Comput. Informat.*, vol. 33, no. 1, pp. 1–34, 2014.

[35] D. Jiang, W. Wang, L. Shi, and B. Song, "A compressive sensing-based approach to end-to-end network traffic reconstruction," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 1, pp. 507–519, Jan.–Mar. 2020.

[36] M. Thorup and Y. Zhang, "Tabulation based 4-universal hashing with applications to second moment estimation," in *Proc. 15th ACM-SIAM Symp. Discrete Algorithms*, 2004, pp. 615–624.

[37] S. Muthukrishnan, "Data streams: Algorithms and applications," in *Proc. 14th ACM-SIAM Symp. Discrete Algorithms*, 2003, pp. 117–236.

[38] Y. Kanda, R. Fontugne, K. Fukuda, and T. Sugawara, "Admire: Anomaly detection method using entropy-based PCA with three-step sketches," *Comput. Commun.*, vol. 36, no. 5, pp. 575–588, 2013.

[39] B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen, "Sketch-based change detection:methods, evaluation, and applications," in *Proc. 3rd ACM SIGCOMM Conf. Internet Meas.*, 2003, pp. 234–247.

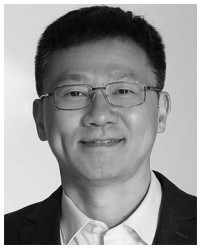
[40] L. Laboratory, "Darpa'98 dataset," 1998. [Online]. Available: https://www.ll.mit.edu//ideval/data/

[41] MAWILAB, "Network traffic anomaly classification," 2014. [Online]. Available: http://www.fukuda-lab.org/mawilab/classification/index.html

[42] W. Kosters, W. Pijls, and V. Popova, "Complexity analysis of depth first and FP-growth implementations of Apriori," in *Proc. Int. Workshop Mach. Learn. Data Mining Pattern Recognit.*, 2003, pp. 284–292.

[43] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, Feb. 2020.

Yating Liu (Student Member, IEEE) received the B.E. degree in communication engineering from Central South University, Changsha, China, in 2015, and the M.Sc. degree in electronic and communication engineering from Tsinghua University, Beijing, China, in 2018. She is currently working toward the Ph.D. degree with Tsinghua University and Peng Cheng Laboratory, Shenzhen, China. Her research interests include network anomaly detection and multimodal learning.



Yuantao Gu (Senior Member, IEEE) received the B.E. degree in electronic engineering from Xi'an Jiaotong University, Xi'an, China, in 1998, and the Ph.D. degree in electronic engineering (with hons.) from Tsinghua University, Beijing, China, in 2003. He joined the Faculty of Tsinghua University in 2003, and is currently a Professor with Department of Electronic Engineering. He was a Visiting Scientist with Microsoft Research Asia during 2005–2006, Research Laboratory of Electronics at Massachusetts Institute of Technology during 2012–2013, and he was with the Department of Electrical Engineering and Computer Science with the University of Michigan, Ann Arbor, MI, USA, during 2015. His research interests include high-dimensional statistics, sparse signal recovery, temporal-space and graph signal processing, related topics in wireless communications, and information networks. He has been a Senior Area Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING since 2019, an Elected Member of the IEEE Machine Learning for Signal Processing Technical Committee since 2019, and an Elected Member of the IEEE Signal Processing Theory and Methods Technical Committee since 2017. He was an Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING from 2015 to 2019 and a Handling Editor of *ELSEVIER Digital Signal Processing* from 2015 to 2017. He was the recipient of the Best Paper Award of IEEE GlobalSIP in 2015, the Award for Best Presentation of Journal Paper of IEEE ChinaSIP in 2015, Zhang Si-Ying (CCDC) Outstanding Youth Paper Award (with his student) in 2017, and the Best Paper Award of ICCSPS in 2019, Outstanding Reviewer Award and Outstanding Editorial Board Member Award for his services for IEEE ICASSP in 2019 and for IEEE TSP in 2021.

Xinyue Shen received the Ph.D. degree in 2018 and B. E. degree (with hons.) in 2013 from the Department of Electronic Engineering, Tsinghua University, Beijing, China. She was a Postdoctoral with Stanford University, Stanford, CA, USA, from 2019 to 2022, and also as a joint Postdoctoral with the Chinese University of Hong Kong, Hong Kong, from 2019 to 2021. She was a Visiting Student Researcher with Stanford University from 2015 to 2016. Her research interests include optimization algorithms and applications in statistics, signal processing, and machine learning.

Qingmin Liao (Senior Member, IEEE) received the B. S. degree in radio technology from the University of Electronic Science and Technology of China, Chengdu, China, in 1984, and the M.S. and Ph.D. degrees in signal processing and telecommunications from the University of Rennes 1, Rennes, France, in 1990 and 1994, respectively. Since 1995, he has been joining with Tsinghua University, Beijing, China. In 2002, he became Professor with the Department of Electronic Engineering, Tsinghua University. Since 2006, he has been the Director of the Shenzhen Key Laboratory of Information Science and Technology, with Shenzhen International Graduate School, Tsinghua University. More than the last 35 years, he has authored or coauthored more than 200 peer-reviewed journal and conference papers. His research interests include image/video processing, transmission and analysis, biometrics; and their applications to teledetection, medicine, industry, and sports.

Quan Yu received the B.S. degree in radio physics from Nanjing University, Nanjing, China, in 1986, the M.S. degree in radio wave propagation from Xidian University, Xi'an, China, in 1988, and the Ph.D. degree in fiber optics from the University of Limoges, Limoges, France, in 1992. He is currently a Research Professor with the Peng Cheng Laboratory. His main research interests include architecture of wireless networks and cognitive radio. He is an Academician of the Chinese Academy of Engineering and the founding Editor-in-Chief of the Journal of *Communications and Information Networks*.