# Fast Low-Rank Matrix Approximation with Locality Sensitive Hashing for Quick Anomaly Detection

Gaogang Xie[1], Kun Xie[2,3], Jun Huang[2], Xin Wang[3], Yuxiang Chen[2], Jigang Wen[1]
[1] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2] College of Computer Science and Electronics Engineering, Hunan University, Changsha, China
[3] Department of Electrical and Computer Engineering, State of New York University at Stony Brook, USA
xie@ict.ac.cn, xiekun@hnu.edu.cn, huangjun120801@gmail.com, x.wang@stonybrook.edu,
csyxchen@hnu.edu.cn, wenjigang@ict.ac.cn

***Abstract*—Detecting anomalous traffic is a critical task for advanced Internet management. The traditional approaches based on Principal Component Analysis (PCA) are effective only when the corruption is caused by small additive i.i.d. Gaussian noise. The recent Direct Robust Matrix Factorization (DRMF) is proven to be more robust and accurate in anomaly detection, but it incurs a high computation cost due to its need of singular value decomposition (SVD) for low-rank matrix approximation and the iterative use of SVD execution to find the final solution.**

**To enable the anomaly detection for large traffic matrix with the use of DRMF, we formulate the low-rank matrix approximation problem as a problem of searching for the subspace to project the traffic matrix with the minimum error. We propose a novel approach, LSH-subspace, for fast low-rank matrix approximation. To facilitate the matrix partition for the quick search of the subspace, we propose several novel techniques: a multi-layer locality sensitive hashing (LSH) table to reorder the OD pairs based on LSH function, a partition principle to guide the partition to minimize the projection error, and a lightweight algorithm to exploit the sparsity of the outlier matrix to update the LSH table at low overhead. Our extensive simulations based on real trace data demonstrate that our LSH-subspace is 3 times faster than DRMF with high anomaly detection accuracy.**

***Index Terms*—Low-Rank Matrix Approximation, Anomaly Detection**

## I. Introduction

Traffic anomalies, such as flash crowds, denial-of-service attacks, port scans, and the spreading of worms, can have detrimental effects on network services. These anomalies often lead to unusual and significant changes of network traffic levels, and the changes can span multiple links. Detecting and diagnosing these anomalies are critical to both network operators and end users.

A popular assumption in anomaly detection is that the normal data have close values, while outliers are far away from the others and lie in the low-density region of the data distribution [1], [2]. Based on the experiments of real traffic trace, the authors in [3] further reveal that traffic data have the features of temporal stability and spatial correlation. With the structure feature and similarity, normal traffic data will reside in a low-dimensional linear subspace and form a low-rank matrix, while the anomalies (outliers) will stay outside this subspace. Based on these observations, the authors propose to decompose the noisy traffic data into two parts, low-rank normal data and outlier data, and detect the anomaly by finding the outlier data.

Many efforts [4]–[15] have been made to develop various anomaly detection algorithms to separate the outlier data from the noisy traffic data. Among which, Principal Component Analysis (PCA) [4] is perhaps the best-known statistical-analysis technique for Internet anomaly detection. Although effective when the corruption is caused by small additive i.i.d. Gaussian noise, recent studies show that traditional PCA-based approaches fail under the large corruption, even if the corruption affects only very few of the observations [16].

To make PCA robust to large errors and outliers, Candès et. al. [17] proposed to approach Robust PCA (RPCA) via Principal Component Pursuit (PCP), which decomposes a given observation (noisy) matrix $X$ into a low-rank component $X'$ and a sparse component $E$. However, general RPCA solutions resort to some relaxation techniques, which may largely impact the accuracy of anomaly detection. Work in [18] proposes a direct robust matrix factorization (DRMF) which aims at minimizing the $L_2$ error of the low-rank matrix approximation subject to the condition that the number of outliers is small without using the relaxation techniques. DRMF is proven to be very effective in video activity detection and USPS anomaly detection [18]. Although promising, this method is very challenging to apply for practical anomaly-detection in Internet for several reasons:

- To obtain the low-rank component in the noisy traffic data, DRMF scheme involves singular value decomposition (SVD), which has $O\left(\min\left\{mn^2, nm^2\right\}\right)$ time complexity to handle a matrix of $\mathbb{R}^{m\times n}$. This renders the exact SVD operation impractical for large traffic trace data.
- To accurately separate the low-rank and the outlier components, DRMF scheme needs to iteratively execute SVD, resulting in a prohibitively high computation cost.

It is very important to efficiently detect network anomalies. In light of the importance of DRMF and the above challenges,

This work is supported by the National High Technology Research and Development Program of China (Grant No. 2015AA016101 and 2015AA015603), the National Natural Science Foundation of China (Grant Nos.61502462, 61572184), and U.S. NSF CNS 1526843.

978-1-5090-5336-0/17/$31.00 ©2017 IEEE

in this work, we propose a computationally efficient algorithm to enable network anomaly detection based on DRMF. More specifically, we formulate the low-rank matrix approximation problem as a subspace searching problem to find a subspace in which the projection of the traffic matrix has the minimum error. Specifically, to exploit the traffic features such as temporal stability and spatial correlation to efficiently search for the subspace, we propose a novel LSH-subspace scheme which iteratively partitions the traffic matrix into sub-matrices with each contributing a basis vector in the subspace. Our main contributions in this paper are listed as follows.

- We propose a novel multi-layer locality sensitive hashing (LSH) table to facilitate the matrix partition procedures. The table can reorder and buffer origin and destination (OD) pairs based on LSH function with various similarity levels in different layers, which allows the sub-matrices partitioned to hold OD pairs with higher correlations. This helps to further find the basis vectors to better represent the matrix.
- To speed up the subspace searching and minimize the projection error, we propose a novel partition principle, which further partitions the sub-matrix that is least represented by the partial-subspace found until the dimension of the subspace reaches the desired $k$.
- To reduce the overall computation cost in the iterative process for anomaly detection, we propose a lightweight algorithm which exploits the sparsity of the outlier matrix to reduce the overhead in updating the LSH table for the low-rank matrix approximation in each iteration round.
- We compare LSH-subspace scheme with the state of art anomaly detection algorithms using the real traffic trace data. Our simulation results demonstrate that LSH-subspace can achieve the high anomaly detection accuracy at much faster speed thanks to its lower computational cost.

The rest of the paper is organized as follows. Section II presents the related work. We present our system model, problem, and the challenges in Section III. We provide a solution overview in Section IV. We describe our multi-layer LSH table, adaptive subspace searching algorithm, and algorithm for fast subspace searching in iterative execution in Section V, Section VI, and Section VII, respectively. Finally, we implement the proposed LSH-subspace scheme and evaluate the performance using real traffic trace data in Section VIII, and conclude the work in Section IX.

## II. Related work

Despite a large body of literature on traffic characterization [6], [19]–[21], anomaly detection remains a challenge, and Principal Component Analysis (PCA) [4], [22] is perhaps the best-known statistical-analysis technique. PCA uses an orthogonal transformation to convert possibly correlated observed variables into a set of linearly uncorrelated variables called principal components, which constitutes a low-dimensional subspace and can compactly represent the multi-dimensional data set. Some recent papers that apply PCA to the traffic anomaly detection have shown some promising initial results [5]–[12]. PCA has also been combined with sketches [13], [14] and distributed monitors [15] to provide more efficient traffic anomaly detection.

A traditional PCA method gives the optimal estimate when the corruption is caused by additive i.i.d. Gaussian noise whose magnitude is small, but breaks down under a large corruption even if it affects only very few of the observations [16]. To make PCA robust to large errors and outliers, Candès et. al. [17] proposed to approach Robust PCA (RPCA) via Principal Component Pursuit (PCP), which decomposes a given observation (noisy) matrix $X$ into a low-rank component $X'$ and a sparse outlier component $E$. To make the problem solvable, the work in [23] replaces the matrix rank and the cardinality ($\|\|_0$) functions with their convex surrogates, the nuclear norm $\|\|_*$ (i.e., the sum of its singular values) and the $L_1$ norm $\|\|_1$, and solves the following convex optimization problem

$$\begin{aligned} &\min_{X',E} \{\|\mathrm{X}'\|_* + \lambda\|E\|_1\} \\ &st. X' + E = X \end{aligned} \tag{1}$$

where $\lambda$ is a positive weighting parameter. To decompose the data into low-rank component and sparse component, these methods resort to some relaxation techniques which may largely impact the accuracy of anomaly detection.

Recently, work in [18] proposes a direct robust matrix factorization (DRMF) which aims at minimizing the $L_2$ error of the low-rank approximation subject to the condition that the number of outliers is small. To solve DRMF, a block coordinate descent method is adopted which includes a singular value decomposition (SVD) and an efficient threshold procedure. DRMF is simple to implement and is proven to be very effective in video activity detection and USPS anomaly detection. However, the solution involves the iterative execution of the SVD decomposition, which will bring very high computation cost and is not scalable to large traffic data.

Given the importance of DRMF in anomaly detection and its potential efficient application in Internet, this paper focuses on the reduction of computation complexity for its low-rank matrix approximation. Specially, we propose several novel techniques, including a multi-layer LSH table to reorder and buffer OD pairs based on LSH function, an iterative and adaptive matrix partition algorithm to efficiently search the subspace for low-rank matrix approximation, and a quick LSH table updating algorithm which takes advantage of the sparsity of the outlier matrix to reuse the reordered OD pairs of the previous iteration. The simulation results on the traffic trace data demonstrate that with these novel techniques, our scheme can achieve significantly better performance with high anomaly detection accuracy under much lower computation cost compared with other state of art anomaly detection schemes.

## III. Problem and challenge

For matrix $A \in \mathbb{R}^{m\times n}$, we write $A_{(i)}$ (i.e. subscript) for its $i$th row and $A^{(j)}$ (i.e., superscript) for its $j$th column. We use $\mathbb{O}^{m\times n}$ to represent the subset of $\mathbb{R}^{m\times n}$ whose columns are orthonormal. Since the columns of $V \in \mathbb{O}^{m\times n}$ are an orthonormal basis, we sometimes use "the subspace $V$" to refer to the subspace spanned by the columns of $V$. Given a real number $x$, $floor(x) = \lfloor x \rfloor$ is the largest integer less than or equal to $x$ and $ceiling(x) = \lceil x \rceil$ is the smallest integer greater than or equal to $x$.

### *A. System model and problem*

Given a network consisting of $N$ nodes, this paper models the traffic data with a traffic matrix $X \in \mathbb{R}^{m \times n}$ ($m = N \times N$), where a row of $X$ represents the time evolution of a single OD pair and a column represents the traffic data of all OD pairs at one time slot. $n$ denotes the total number of time slots captured in the matrix.

The data captured by a traffic matrix tend to be noisy and are subject to outliers and arbitrary corruptions. As discussed in the introduction, the traffic data have the features of temporal stability and spatial correlation. Thus normal traffic data will reside in a low-dimensional linear subspace and form a low-rank matrix, while the anomalies (outliers) will stay outside this subspace. Accordingly, we formulate the anomaly detection problem as a constrained optimization problem:

$$\begin{aligned} & \min_{L,S} \|(X - S) - L\|_F^2 \\ & \text{s. t.} \operatorname{rank}(L) \leq k \\ & \|S\|_0 \leq e \end{aligned} \tag{2}$$

where $S$ is the matrix of outliers, $L$ is the low-rank approximation of matrix $X - S$, $k$ is the truncation rank, and $e$ is the maximal number of non-zeros entries in $S$ that cannot be ignored as outliers. We do not need the actual number of outliers, but only use $e$ to provide an upper limit. The formulation in (2) aims at minimizing the $L_2$ error of the low-rank approximation subject to the condition that the number of outliers is small, without any further assumptions. By excluding the outliers from the effort of low-rank approximation, we can ensure the reliability of the estimated low-rank structure. The anomaly can be easily detected after obtaining outlier matrix $S$.

Usually, optimization problems involving the rank or the $L_0$-norm i.e. set cardinality are difficult to solve. Some relaxation techniques are proposed to solve the low-rank matrix approximation, such as using the nuclear norm of matrix to replace $L_0$-norm. However, these relaxations may largely impact the estimation accuracy of low-rank matrix approximation and further impact the anomaly detection accuracy.

Rather than resorting to relaxation techniques, we directly solve the problem (2) with the constraints of the matrix rank and the cardinality of the outlier set for more accuracy anomaly detection. Following [18], we adopt the block coordinate descent strategy to solve the problem in (2) in an iterative way, as shown in Algorithm 1. In each iteration, we first fix the current estimate of the set of outliers $S$ and exclude them from the measurement $X$ to get the "clean" traffic data $C$, and then fit $L$ based on $C$. Next, we update the outliers $S$ based on the error $E = X - L$.

Specially, a theorem proven by Eckart and Young [24] shows that the error in approximating a matrix $A$ by $A_k$ can be written: $\|A - A_k\|_F^2 \leq \|A - B\|_F^2$ where $B$ is any matrix with rank $k$, $A_k$ is the rank-$k$ truncated SVD of matrix $A$. Therefore, a straightforward solution to the low-rank matrix approximation problem (3) is directly given by SVD according to Eckart and Young's theorem, and the solution to $L$ is simply the truncated SVD approximation to the "cleaned" traffic data $C$ given in (3).

**Algorithm 1** Anomaly Detection Based on Matrix Factorization

```
Input: $X$: the noisy traffic matrix
       $k$: the maximal rank of the matrix factorizaiton
       $e$: the maximal number of ourliers
       $S$: the outlier matrix initialized
Output: $L$: the low-rank matrix, $S$: the outlier matrix
while not converged do
    Solve the low-rank matrix approximation problem
        $L = \arg\min_{L} \|C - L\|_F^2$
        s.t. $C = X - S$                                  (3)
        $rank(L) \leq k$
    Solve the outlier detection problem:
        $S = \arg\min_{S} \|E - S\|_F^2$
        s.t. $E = X - L$                                  (4)
        $\|S\|_0 \leq e$
end while
```

Moreover, following the theorem in the work of [25] to solve the general problem of $L_0$-norm constrained minimization of the decomposable objective, the outlier detection problem in (4) can also be solved efficiently. As solving problem in (4) is not the focus of this paper, we apply the theorem in the work of [25] to solve the problem.

### *B. Challenge on computation complexity*

Algorithm 1 requires a truncated SVD approximation to solve the problem in (3) in each iterative step. Such an operation, however, introduces high computation cost and is not scalable to deal with large traffic data.

Given a matrix $X \in \mathbb{R}^{m \times n}$, SVD decomposes the matrix into three factors:

$$X = U\Sigma \mathrm{V}^T = \sum\nolimits_{i=1}^{l} \sigma_i u_i v_i^T, \tag{5}$$

where $l = min(m,n)$, $\sigma = [\sigma_1, \cdots, \sigma_l]$ is the vector of singular values of $X$ in the descending order, columns of $U = [u_1, \cdots, u_l] \in R^{m \times l}$ and $V = [v_1, \cdots, v_l] \in R^{n \times l}$ are the corresponding left and right singular vectors. We can find a reduced rank approximation (or truncated SVD $X_k$ with its truncation rank $k$ ) to $X$ by setting all but the first $k$ largest singular values equal to zero and using only the first $k$ columns of $U$ and $V$.

Although promising, the exact SVD has $O\left(\min\left\{mn^2, nm^2\right\}\right)$ time complexity. This is highly unscalable, rendering the straightforward way of obtaining truncated SVD through the exact SVD impractical for large traffic data. Additionally, as the low-rank matrix approximation is executed iteratively in Algorithm 1, the accumulated computation cost will be very high. Therefore, the following two issues become the key challenging problems:

- How to reduce the computation cost of the low-rank matrix approximation?
- How to reduce the total computation cost of the whole iteration process?

### C. Subspace searching problem

The optimal rank-$k$ matrix approximation, i.e., minimizing the squared error $\left\|A - \hat{A}\right\|_F^2$ where $A, \hat{A} \in \mathbb{R}^{m \times n}$ and $\hat{A}$ is a rank-$k$ matrix, is the rank-$k$ truncation of the SVD:

$$A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T = U_k \Sigma_k {V_k}^T, \tag{6}$$

which projects $A$'s rows onto the subspace spanned by the top $k$ right singular vectors, i.e., $A_k = AV_kV_k^T$. The optimality of $A_k$ implies that the columns of $V_k$ span the subspace of dimension at most $k$ in which the squared error of $A$'s row-wise projection is minimized. Therefore, to obtain the optimal rank-$k$ matrix approximation, we can seek to find a subspace in which $A$'s projection has sufficiently low error:

$$\begin{gathered} \min_{V_k} \|A - A_k\|_F^2 \\ s.t. A_k = AV_kV_k^T \\ V_k \in \mathbb{O}^{n \times k} \end{gathered} \tag{7}$$

Therefore, instead of solving the problem in (3) through the truncated SVD, this paper intends to minimize the gap between $A$ and its projection matrix $A_k$ (i.e.,$\|A - A_k\|_F^2$) by searching the subspace $V_k$ with the dimension $k$. In this paper, we call $\|A - A_k\|_F^2$ the projection error.

As the anomaly event in a network seldom happens, the outlier matrix $S$ in Algorithm 1 is a sparse matrix with none zero values at most $e$ locations. Although the "clear" traffic matrix $C$ in (3) is iteratively updated through $X - S$ with the change of $S$ in each iteration, as $S$ is a sparse matrix, only a few entries have none zero values and thus only a few entries change in the "clear" traffic matrix $C$ in the subsequent iterative steps. Therefore, the low-rank matrix approximation for sequent matrices (i.e., $C[t]$, $C[t+1]$) in two sequential iterations (i.e., $t, t+1$) must have some relationship. This provides an opportunity for us to reuse the results and structure of the previous step in the current step to reduce the computation cost of the whole iterative process.

In the next section, we will present our algorithms to quickly calculate the rank-$k$ matrix approximation, and reuse the data structure in iterative steps for fast Internet anomaly detection.

## IV. Solution overview

The key in Algorithm 1 is to solve the problem in (3), which can be further transformed to problem (7) to search the subspace $V_k$ with the dimension $k$ to best approximate the matrix of interest. Our algorithm exploits the structure features and similarities hidden in the traffic data to efficiently search the subspace $V_k$.

In our system model, the data of an OD pair correspond to one row in the traffic matrix. Specially, the recent study in [26], [27] demonstrates that network paths starting from nearby end nodes often have overlapping path segments or go through some common network nodes, especially in the Internet core which has a simple topology. As a result, data from network measurements often have correlations.

To take advantage of these correlations to efficiently search for the subspace $V_k$, we propose a novel LSH-subspace scheme which iteratively partitions the original monitoring matrix into sub-matrices with each contributing a basis vector in the subspace. We design a novel multi-layer LSH table to reorder and buffer the OD pairs based on LSH function so that OD pairs with higher correlations are stored in a sub-matrix. To speed up the subspace searching procedure and minimize the projection error, LSH-subspace well leverages the structure of the traffic matrix by first partitioning the matrix that is least represented by the partial subspace found until the dimension of the subspace reaches $k$. LSH-subspace mainly includes the following three technique components:

- A novel multi-layer LSH table, which reorders and buffers OD pairs in a fast and effective way with only hash calculations. The good property of the LSH guarantees that similar OD pairs are grouped and packed into the same bucket in the LSH table. We utilize well-designed LSH function with different bucket width for different hash table layers, therefore, the OD pairs in the different layers are grouped with different similarity levels. The good property provides a very efficient way to facilitate adaptive matrix partition to search for the subspace.
- An iterative and adaptive partition algorithm, which partitions the large traffic matrix into multiple sub-matrices based on the multi-layer LSH table. As each sub-matrix corresponds to one basis vector in the subspace, one time of matrix partition corresponds to one time of subspace expanding. We also propose a partition principle to select the sub-matrix not well represented to further partition, which helps expand the subspace to maximally reduce the projection error.
- A lightweight LSH table update algorithm, which can well exploit the sparsity of the outlier matrix to reduce the number of items to update in the hash table in each round thus the overall computation cost of the whole iteration process in Algorithm 1.

With these strategies, LSH-subspace can provide quick and highly accurate anomaly detection. In the following two sections, we present our three key techniques: multi-layer LSH table, matrix partition and subspace searching, quick multi-layer LSH table update to reuse partial data from the previous iterative step.

## V. Multi-layer LSH table

To exploit the correlations among OD pairs for quick search of the subspace, similar OD pairs need to be grouped together. We propose to use a multi-layer LSH table to reorder OD pairs based on the LSH function with various similarity levels at different layers.

### A. LSH function

To achieve the low-rank matrix approximation with a quick and accurate search of the corresponding subspace, we reorder the OD pairs and make the OD pairs with higher correlations to locate closely in the LSH table with a higher probability. This can be realized with the help of the LSH function. An LSH function can keep the locality of elements in a data set by mapping close elements into the same hash bucket with a high probability. This is achieved by choosing a hash function from an LSH function family. The ”locality sensitive” can be defined as follows [28]:

*Definition 1:* **(LSH Function Family) [28]:** $\mathbb{H} = \{g : \Upsilon \rightarrow U\}$ is called $(R, cR, P_1, P_2)$ - sensitive for any $p, q \in \Upsilon$

- If $||p,q||_s \leq R$ then $\Pr_{\mathbb{H}}[g(p) = g(q)] \geq P_1$.
- If $||p,q||_s \geq cR$ then $\Pr_{\mathbb{H}}[g(p) = g(q)] \leq P_2$.

where $\|p,q\|_s$ is the distance of elements $p$ and $q$, $\Upsilon$ is the domain of elements. In this paper, OD pairs are the elements that need to be reordered according to their distances. In the LSH, $c > 1$ and $P_1 > P_2$.

We use the row vector of the monitoring matrix as the OD pair vector and apply the LSH function to reorder the OD pairs. Given an OD pair $\mu_i \in \mathbb{R}^n$ $(1 \leq i \leq m)$ where $m$ is the total number of OD pairs, $n$ is the total number of time slots monitored. According to [28], we define the following LSH function $h_{a,b} : \mathbb{R}^n \rightarrow U$ based on Euclidean distance to map the OD pair in the hash table:

$$h_{a,b}(\mu) = \left\lfloor \frac{a^T \mu + b}{W} \right\rfloor \tag{8}$$

where $a$ is a $n$-dimensional random vector with each component chosen independently from a Gaussian distribution, $W$ is the width of a hash bucket, and $b$ is a real number randomly selected from the interval $[0, W)$.

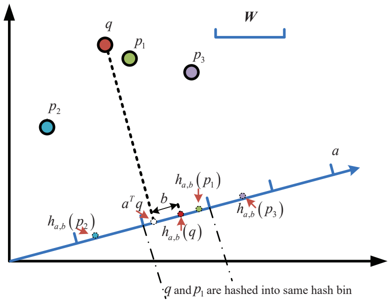


Fig. 1. 3-way traffic tensor

When the hashed addresses of OD pairs are the same, i.e., there is a collision in hashing, the pairs are mapped to the same bucket. According to the conclusion in [28], we have the probability that two OD pairs $p$ and $q$ collide for the LSH defined in (8) is

$$Pr_{a,b}\left[h_{a,b}(p) = h_{a,b}(q)\right] = \int_0^W \frac{1}{d} f_s\left(\frac{t}{d}\right)\left(1 - \frac{t}{W}\right) dt \tag{9}$$

where $d = \|p - q\|_s$ is the distance between OD pairs $p$ and $q$, and $f_s(t)$ is the probability density function of $s$-stable distribution. Function in (9) is an increasing function with the bucket width $W$. Given two OD pairs with a distance $d$, when $W$ becomes larger, these two OD pairs can be mapped to the same bucket with a higher probability.

The LSH function in Eq(8) has a good property, that is, it can map similar OD pairs into the same hash bucket. The example of Fig. 1 illustrates the geometry result of the adopted LSH functions in a 2D space. Given vector $a$ and the query OD pair $q$ , $a^T q$ is the dot product of the two, which projects $q$ onto the vector $a$. The value $b$ in $h_{a,b}(q)$ is applied to further shift the projected point over a distance $b$, and is kept the same when the hash function is used to evaluate the closeness of two vectors. The vector line is divided into multiple buckets with the length of each being $W$. In such a transformation, proximate points, e.g., $q$ and $p_1$, have a high probability of being hashed into the same hash bucket.

### B. Mapping OD pair to multi-layer LSH table

The OD pair groups in LSH table are utilized to facilitate the subspace searching. Instead of directly using the row vectors corresponding to the OD pairs of traffic, we use the normalized row vector as the OD pair's key to calculate its location in the hash table. Denote the matrix $C \in \mathbb{R}^{m \times n}$ as the traffic matrix needs to approximate. Given an OD pair $C_{(i)} \in \mathbb{R}^n$ $(1 \leq i \leq m)$, it can be normalized as

$$C^*_{(i)} = \frac{C_{(i)}}{\sqrt{\sum_{j=1}^{n} C^2_{ij}}} \tag{10}$$

where $C_{ij}$ is the entry $(i, j)$ in the matrix $C$.

Take two OD pair vectors ($r_1 = [2, 3, 4]$ and $r_2 = [4, 6, 8]$) as example. The values (traffic data) of the two vectors are obviously very different, while both OD pairs have same traffic access pattern and thus similar traffic trade. These two vectors should be grouped into the same hash bucket to contribute one basis vector $\left[\frac{2}{\sqrt{29}}, \frac{3}{\sqrt{29}}, \frac{4}{\sqrt{29}}\right]$. Obviously, this basis vector can well represent these two vectors.

To facilitate adaptive subspace searching, instead of using one-layer LSH table, we design a multi-layer LSH table to reorder the OD pairs with various similarity levels (Fig. 1). Among the multiple hash-table layers, the bucket width of the top one is set to $W$, all the OD pairs are applied using the basic hash function (i.e., $h^{(a,b)}(C^*_{(i)}) = \left\lfloor \frac{a^T \cdot C^*_{(i)} + b}{W} \right\rfloor$), to map themselves in the hash table by inserting the OD pair IDs. We denote the top layer as the basic hash table. Other hash tables are all virtual hash tables (their hash bucket width being $2W, 4W, 8W, 16W, \cdots$) which are virtually built from the basic hash table rather than physically exist, and the LSH functions for these virtual hash table will be $h^{(a,b)}(C^*_{(i)}) = \left\lfloor \frac{a^T \cdot C^*_{(i)} + b}{2W} \right\rfloor$, $h^{(a,b)}(C^*_{(i)}) = \left\lfloor \frac{a^T \cdot C^*_{(i)} + b}{4W} \right\rfloor$, $h^{(a,b)}(C^*_{(i)}) = \left\lfloor \frac{a^T \cdot C^*_{(i)} + b}{8W} \right\rfloor$, $h^{(a,b)}(C^*_{(i)}) = \left\lfloor \frac{a^T \cdot C^*_{(i)} + b}{16W} \right\rfloor$, $\cdots$, correspondingly.

Fig. 2 shows an example to illustrate our multi-layer hash table. The hash address of $C_{(2)}$ in the basic hash table is 4. Its hash addresses in the following downward hash tables are 2, 1,1,1,$\cdots$. Specially, we use $F$ to denote the layer of the hash table, with the bottom layer denoted as $F = 1$ and its next upper layer as $F = 2$, etc.

Our multi-layer hash table has following two good properties.

- If we know the hash address of an OD pair in the basic hash table, its hash address in all the downward hash tables can be easily deduced, thus does not need physical building when we have the basic hash table. Given an OD pair, if it is located in the basic hash table with its hash address $l$, its location in the downward hash table

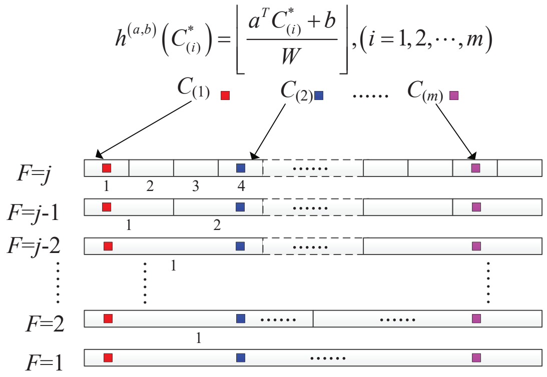

$h^{(a,b)}\left(C^*_{(i)}\right)=\left\lfloor \frac{a^T C^*_{(i)}+b}{W} \right\rfloor,(i=1,2,\cdots,m)$
$C_{(1)}$
$C_{(2)}$
......
$C_{(m)}$
F=j
1
2
3
4
F=j-1
1
2
F=j-2
1
F=2
1
F=1

Fig. 2. Multi-layer hash table

will be $\lceil l/2 \rceil$, $\lceil l/4 \rceil$, $\lceil l/8 \rceil$, $\lceil l/16 \rceil$, $\cdots$. In the opposite direction, if we know an OD pair address in the hash table is $l$, the hash address in its upforward hash table is in the set $\{2l-1, 2l\}$, i.e., $2l-1$ or $2l$.

- Different layers utilize the LSH function with different bucket widths (which exponentially increases from the top to the down hash table: $W, 2W, 4W, 8W, 16W, \cdots$), so hash tables at different layers have the OD pairs grouped in the bucket with different similarity levels. That is, the OD pairs grouped in the hash bucket of the top hash table is more similar than the ones grouped in the buckets of downward hash tables.

In the next section, we will exploit these two good properties to design adaptive-matrix-partition based quick subspace searching algorithm.

## VI. Adaptive subspace searching

Given the rank $k$ and the "real" traffic matrix $C$, our goal of rank-$k$ matrix approximation is to search the subspace $V_k$ which has $k$ orthogonal basis vectors to minimize the projection error $\left\|C - CV_kV_k^T\right\|_F^2$. With the help of multi-layer LSH table, we propose an adaptive matrix partition algorithm to iteratively search the subspace with each sub-matrix contributing one basis vector in the subspace.

In each iterative step, among all the sub-matrices resulted from the previous step, we will select a sub-matrix to partition for further subspace searching until $k$ basis vectors are found. The key problem is how to select the sub-matrix to pursue further processing.

Generally, after $p-1$ times of partitions, the large matrix $C$ has been partitioned into $p$ sub-matrices, denoted by $C_1, C_2, \cdots, C_p$ with $C_i \in \mathbb{R}^{m_i \times n}$. We denote the current partial subspace as $V_p \in \mathbb{O}^{n \times p}$, and the projection error of the large traffic matrix based on this subspace is $\left\|C - CV_pV_p^T\right\|_F^2$.

We use $row(C)$ to denote the row set in the matrix $C$. Obviously, we have $row\,(C) = row\,(C_1) \cup row\,(C_2) \cdots \cup row\,(C_p)$ and $row\,(C_i) \cap row\,(C_j) = \phi$ for $i \neq j, i, j = 1, 2, \cdots, p$. Therefore, we have

$$\begin{aligned}
&\left\|C - CV_pV_p^T\right\|_F^2 \\
&= \left\| \begin{pmatrix} C_1 & C_2 & \cdots & C_p \end{pmatrix}^T - \begin{pmatrix} C_1 & C_2 & \cdots & C_p \end{pmatrix}^T V_pV_p^T \right\|_F^2 \\
&= \left\|C_1 - C_1V_pV_p^T\right\|_F^2 + \left\|C_2 - C_2V_pV_p^T\right\|_F^2 + \cdots + \left\|C_p - C_pV_pV_p^T\right\|_F^2
\end{aligned} \tag{11}$$

As our goal is to search the subspace to minimize the projection error $\left\|C - CV_pV_p^T\right\|_F^2$, according to (11), we propose a ***partition principle***: among all sub-matrices, we select the one with largest projection error using current subspace to further partition and expand the subspace. The selected sub-matrix is

$$C_i = \underset{C_i, i=1,2,\cdots,p}{\arg\max} \left\|C_i - C_iV_pV_p^T\right\|_F^2. \tag{12}$$

That is, the partition is always focused on the sub-matrix with the maximum potential for error reduction.

Based on ***partition principle***, we design our adaptive partition algorithm for fast subspace searching in Algorithm 2.

**Algorithm 2** Adaptive Matrix Partition for Fast Subspace Searching

```
Input: multi-layer LSH table: $H$
Output: subspace $V \in O^{n \times k}$
 1: $Q = NULL$ //$Q$ is the list of sub-matrices that contribute basis vectors
 2: $Q$.insert($H[1,1]$) //$H[1,1]$ corresponds to the whole matrix
 3: $v_1$=centroid($H[F,f]$),$V_1 = \{normalized(v_1)\}$, $p = 1$ //$p$ denotes how many basis vectors found
 4: while $p < k$ do
 5:     According to partition principle, for each hash bucket in $Q$, denoting the corresponding matrix as $C_i$ for $i = 1, 2, \cdots, p$, identify the matrix needs to partition though $C_i = \arg\max_{C_i, i=1,2,\cdots,p} \|C_i - C_iV_pV_p^T\|_F^2$
 6:     $F = F(C_i)$ denotes the matrix's layer ID in Multi-layer LSH table $H$, $f = f(C_i)$ denotes the matrix's index in the $F$th layer.
 7:     $Q$.remove($H[F,f]$) //remove the hash bucket corresponding to $C_i$ from $Q$
 8:     Remove $C_i$'s contributed basis vector from $V_p$
 9:     Partition matrix $C_i$, the two sub-matrices ID in the Multi-layer LSH table $H$ are: table layer $F = F + 1$, indexes in this layer $ID = SET\{2f-1, 2f\}$
10:     for $f \in ID$ //using newly partition sub-matrices to update $Q$ and the subspace $V_p$ do
11:         $Q$.insert($H[F,f]$)
12:         $V_p = [V_p \;\; GS(V_p, H[F,f].centroid)]$ //GS=Gram-Schmidt orth-normalization
13:     end for
14:     $V_{p+1} = V_p$
15:     $p = p + 1$
16: end while
17: return $V_p$
```

In Algorithm 2, $Q$ is used to record the list of sub-matrices that contribute basis vectors. $V_p$ is the subspace found after $p-1$ times of partitions. As shown in line 1-3, starting with the lowest layer hash table, which contains all rows in the traffic matrix, we take its centroid as a representative to include in the subspace span and initialize $Q$ by inserting bucket ID (i.e., $H[1,1]$) corresponding to the whole traffic matrix.

The iterative subspace searching will continue until the whole matrix has been partitioned $k-1$ times and thus $k$ basis vectors are found. According to ***partition principle***, on line 5, each iterative step will select the sub-matrix that has the maximum projection error with the current subspace to partition. We use $C_i$ to denote the sub-matrix that needs to partition. We use $F = F(C_i)$ to denote the layer index of the matrix $C_i$'s table, and $f = f(C_i)$ to denote the matrix $C_i$'s index on the $F$ layer.

Before being further partitioned, we should first remove $H[F, f]$ from $Q$, remove $C_i$'s contributed basis vector from $V_p$. With the help of our multi-layer hash table, the matrix $C_i$ can be further partitioned into two sub-matrices which correspond to the hash buckets $H[F+1, 2f-1]$ and $H[F+1, 2f]$. With the good property held by the multi-layer hash table, these two sub-matrices have more similar OD pairs and the union set of OD pairs in both sub-matrices is equal to the OD pair set in the large matrix partitioned before. Line 9 is to locate the hash buckets of these two newly partitioned sub-matrices. On lines 10-13, the subspace is updated and expanded using the newly partitioned sub-matrices.

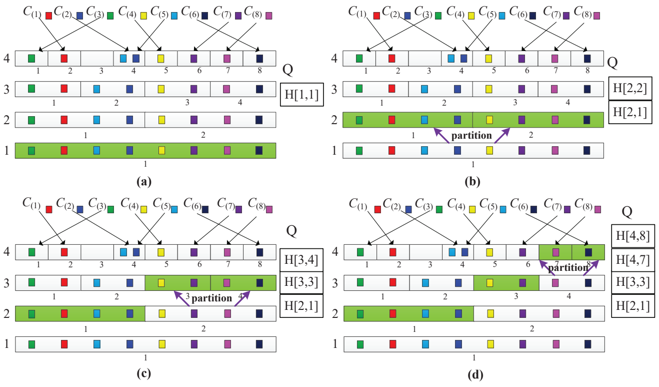


Fig. 3. An example to illustrate the partition based subspace search

Designed based on multi-layer LSH table, our subspace searching algorithm has following good characteristics:

- Facilitated by multi-layer LSH table, matrix partition can be easily achieved using the hash table from downward layer to the upward layer.
- As hash tables of different layers represent different similarity levels, matrix partition makes the OD pairs in the sub-matrices are more similar than the large matrix before the partitioning. Therefore, the basis vectors generated by these sub-matrices can be more representative than the basis vectors generated with a large matrix directly.

These good characteristics ensure the efficiency and effectiveness of our subspace searching algorithm.

We use Fig.3 to illustrate the proposed subspace search algorithm through adaptive matrix partition. To find the rank-4 approximation of the traffic matrix, in Fig.3(a), we take the centroid of the whole traffic matrix as the initialized subspace and insert the hash bucket (i.e., $H[1, 1]$) which corresponds to the whole traffic matrix into the list $Q$. In Fig.3(b), facilitated by the multi-layer LSH table, the whole matrix is partitioned into two sub-matrices with each consisting of more similar rows and contributing one basis vector. After this partition, the list $Q$ is updated to include two hash buckets (i.e., $H[2, 1]$, $H[2, 2]$) corresponding to these two sub-matrices. As the searched subspace only has two basis vectors, further partition is needed.

We assume the projection error of $H[2, 2]$ is larger than that of $H[2, 1]$, so the sub-matrix corresponding to $H[2, 2]$ needs to be further partitioned, and the list $Q$ includes three hash buckets $H[2, 1]$, $H[3, 3]$, $H[3, 4]$ after this partition as shown in Fig.3(c). As we need to search for a subspace with the dimension 4, we should further select one sub-matrix in Fig.3(c) to partition. Assume among the three sub-matrices (corresponding to $H[2, 1]$, $H[3, 3]$, $H[3, 4]$), the sub-matrix $H[3, 4]$ has the largest projection error. Facilitated by the multi-layer LSH table, the sub-matrix of $H[3, 4]$ is easily partitioned into two sub-matrices (i.e., $H[4, 7]$, $H[4, 8]$). The final subspace found includes the centroid vectors of $H[2, 1]$, $H[3, 3]$, $H[4, 7]$, and $H[4, 8]$.

As each sub-matrix contributes one basis vector, to let the subspace found in each iterative step to well represent the traffic matrix, the rows in all the sub-matrices in all iterative steps should cover all the rows in the traffic matrix. As shown in Fig.3, our matrix partition procedure satisfies the above requirement in each iterative step. In each iterative step (shown in Fig.3(a)-(d)), the green ones always cover the whole matrix.

## VII. Fast Subspace Searching for Iterative Execution

As shown in Algorithm 1, the low-rank matrix approximation problem (3) and the outlier detection problem (4) are alternately solved in each iterative step in the whole anomaly detection algorithm.

As the outlier matrix $S$ is updated in each iterative step, $C = X - S$ also changes. To search for the rank-$k$ matrix that can approximate $C$ in each step, the straight-forward way is to first build a new multi-layer LSH table to hold the newly updated matrix $C$, then apply Algorithm 2 to search the subspace. The computation cost is still high.

The outlier matrix $S$ is usually sparse with all entries being zero except at most $e$ entries. If we compare two outlier matrices obtained in two sequential steps, the large number of rows in the matrix should remain unchanged except only a few rows (at most $2e$ rows). To well utilize this feature, we propose Algorithm 3 to reuse the multi-layer LSH table built in the previous step in the current iterative step by only updating a few rows.

**Algorithm 3** Reuse the Multi-layer LSH Table for Fast Subspace Searching in Two Sequential Iterative Steps

```
Input: outlier matrices $S[t]$, $S[t+1]$ of two sequential steps $t$ and
    $t+1$, the multi-layer LSH table built for $C[t] = X - S[t]$
    The measurement traffic data: $X$
Output: the subspace of $C[t+1]$=$X - S[t+1]$
1: Scan $S[t]$ and $S[t+1]$, use the sets $R[t]$ and $R[t+1]$ to record
   the row index in $S[t]$ and $S[t+1]$ that have entry values not
   zero, respectively.
2: $R = R[t] \cup R[t+1]$
3: for each $r \in R$ do
4:     Delete row $C[t]_{(r)}$ from the multi-layer LSH table
5:     Insert row $C[t+1]_{(r)}$ into the multi-layer LSH table
6: end for
7: Apply Algorithm 2 to find the subspace for $C[t+1]$, denoted
   by $V$, and return $V$.
```

On lines 1-2, we scan the outlier matrices obtained in two sequential steps to identify all possible rows changed from matrix $C[t]$ to $C[t+1]$. On lines 3-6, we update the mapping of these possible rows in the multi-layer LSH table. After that, the subspace for the new matrix $C[t+1]$ can be obtained by applying Algorithm 2.

## VIII. Performance Evaluations

We use the public traffic trace data Abilene [29] to evaluate the performance of our proposed LSH-subspace. Before we present the simulation results, we first present the setting of our simulation.

For more efficient data processing, data normalization is often applied to scale the variables or features of data. We normalize the raw traffic data through $l_{i,j} = \frac{l_{i,j}-\min\limits_{u,v}\{l_{u,v}\}}{\max\limits_{u,v}\{l_{u,v}\}-\min\limits_{u,v}\{l_{u,v}\}}$ to make their values to be within the range [0,1] where $\max\limits_{u,v}\{l_{u,v}\}$ and $\min\limits_{u,v}\{l_{u,v}\}$ are the maximum and minimum values of all the traffic data, respectively.

To generate the corrupted synthesized data $X \in \mathbb{R}^{m\times n}$ from the raw trace data $L \in \mathbb{R}^{m\times n}$, we first generate the outlier matrix $S \in \mathbb{R}^{m\times n}$ by randomly selecting $\gamma\times(m\times n)$ locations as the outlier locations where $\gamma$ denotes the outlier ratio. Instead of following the Gaussian distribution, to evaluate how robust the proposed anomaly detection algorithm is in the presence of large errors, the outlier value is randomly generated between [0,10]. The synthesized data $X$ is the sum of the outlier data $S$ and the raw data $L$, that is $x_{i,j} = l_{i,j} + s_{i,j}$ for all $(i, j)$.

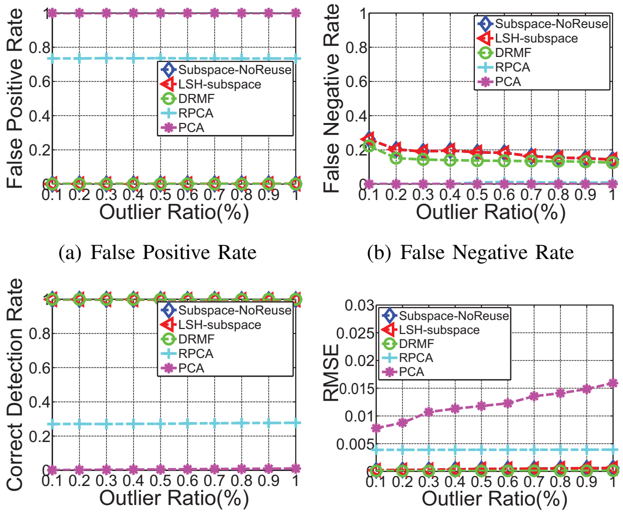
The following performance metrics are utilized to evaluate our proposed LSH-subspace:

- **False Positive Rate:** the proportion of non-outliers that are wrongly identified as outliers.
- **False Negative Rate:** the proportion of outliers that are not identified.
- **Correct Detection Rate:** the proportion of entries that are correctly identified as outlier or non-outlier.
- **RMSE On Outlier:** RMSE (root mean square error) is the standard deviation of the differences between outlier values detected and the raw outlier values.
- **Computation time:** the average number of seconds taken to detect anomalies.
- **Speedup:** Given the computation time under two different algorithms ($alg_1$ and $alg_2$), denoted as $T_1$ and $T_2$, the speedup in the computation time of the $alg_2$ with respect to the $alg_1$: $S_{1-2} = T_1/T_2$.

All simulations are run on a common PC, which is equipped with one Intel (R) I5-4590 CPU (3.3GHz) (4 Cores) and 16.00GB RAM. To measure the computation time, we insert a timer to all the implemented approaches.

To evaluate the performance of the proposed LSH-subspace, we implement five schemes for performance comparison. We first implement DRMF [18] with the truncated SVD + error thresholding iteratively executed to detect the anomalies. The second is our LSH-subspace, which iteratively partitions the traffic matrix into sub-matrices to search for the subspace following Algorithm 2. The truncated SVD for traffic matrix is calculated based on the subspace. Moreover, to reuse the reordered OD pairs of the previous iteration, LSH-subspace also includes an algorithm to quickly update the LSH table to hold the newly updated $C$. Different from LSH-subspace, in the third scheme (denoted as Subspace-NoReuse), a new LSH table is built to hold the newly update matrix $C$. Besides above three schemes, we also implement RPCA [17] and PCA to detect the anomaly in the traffic matrix.

In Fig.4, PCA achieves the worst performance with its false positive rate almost 1 and correct detection rate 0 although its false negative rate is low. As we generate the outlier value randomly in a large range, PCA is not robust to these outliers and fails to separate these corruption from normal data, which confirms the observation of [18]. Compared to DRMF, LSH-subspace and Subspace-NoReuse, the false positive rate and correct detection rate under RPCA is much worse. It uses the trace norm to relax the low-rank feature of traffic matrix, which largely impacts the detection performance. Higher false positive rate would result in false anomaly alarms, which may largely increase the network maintenance cost.



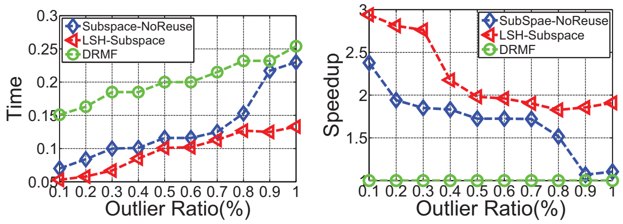
(a) False Positive Rate

(b) False Negative Rate

(c) Correct Detection Rate

(d) RMSE On Outlier

Fig. 4. Performance comparison.



(a) Time

(b) Speedup

Fig. 5. Speed comparison.

DRMF, LSH-subspace, and Subspace-NoReuse all use the matrix rank and set cardinality as the direct constraints in the anomaly detection procedure. They achieve similar good performance in terms of false positive rate, correct detection rate, and RMSE. Both our LSH-subspace and Subspace-NoReuse follow the Algorithm 1 and use our subspace searching (in Algorithm 2) to obtain the approximate low-rank matrix. As a result, they achieve the same accurate performance. Their false negative rates are slightly lower than that of DRMF as they obtain the truncated matrix not through an exact SVD.

As DRMF, LSH-subspace, and Subspace-NoReuse achieve similar good performance, in Fig. 5, we further compare their computation speeds. Specially, to calculate the speedup metric, we use DRMF as the baseline algorithm and set $alg_1 = DRMF$. With the OD pairs grouping through the LSH function and the search of the subspace through iterative matrix partition, our LSH-subspace and Subspace-NoReuse are up to 3 and 2.5 times faster compared with DRMF. Compared with Subspace-NoReuse, our LSH-subspace runs even faster to detect the anomalies as it reuses part of the LSH table in the previous iteration in the current iteration to further reduce the computation time.

All the simulation results show that the techniques proposed in LSH-subspace are very efficient and effective to quickly and accurately detect the traffic anomalies.

## IX. Conclusion

To facilitate quick low-rank matrix approximation for fast anomaly detection, we propose LSH-subspace to quickly search for the subspace to represent the traffic matrix. LSH-subspace iteratively partitions the traffic matrix into sub-matrices with each contributing a basis vector in the subspace. To facilitate the matrix partition for fast subspace searching, we propose a novel multi-layer LSH table which can reorder and buffer origin and destination (OD) pairs based on locality sensitive hashing (LSH), with OD pairs regrouped with various similarity levels in different layers. To speed up the subspace searching and minimize the projection error, the sub-matrix least represented by the partial-subspace found is selected for further partition. Finally, to reduce the overall computation cost in the iterative process for anomaly detection, we further propose a lightweight algorithm which exploits the sparsity of the outlier matrix to reduce the overhead in updating the LSH table in each iteration round. The simulation results based on the real trace data demonstrate the effectiveness and efficiency of LSH-subspace.

## References

[1] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *ACM sigmod record*, vol. 29, pp. 93–104, ACM, 2000.

[2] M. Zhao and V. Saligrama, "Anomaly detection with score functions based on nearest neighbor graphs," in *Advances in Neural Information Processing Systems*, pp. 2250–2258, 2009.

[3] K. Xie, L. Wang, X. Wang, G. Xie, J. Wen, and G. Zhang, "Accurate recovery of internet traffic data: A tensor completion approach," in *IEEE INFOCOM*, pp. 1–9, IEEE, 2016.

[4] H. Hotelling, "Analysis of a complex of statistical variables into principal components.," *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.

[5] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft, *Structural analysis of network traffic flows*, vol. 32. ACM, 2004.

[6] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *ACM SIGCOMM Computer Communication Review*, vol. 34, pp. 219–230, ACM, 2004.

[7] A. Lakhina, M. Crovella, and C. Diot, "Characterization of network-wide anomalies in traffic flows," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pp. 201–206, ACM, 2004.

[8] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," in *ACM SIGCOMM Computer Communication Review*, vol. 35, pp. 217–228, ACM, 2005.

[9] L. Huang, X. Nguyen, M. Garofalakis, M. I. Jordan, A. Joseph, and N. Taft, "In-network pca and anomaly detection," in *Advances in Neural Information Processing Systems*, pp. 617–624, 2006.

[10] L. Huang, X. Nguyen, M. Garofalakis, J. M. Hellerstein, M. Jordan, A. D. Joseph, N. Taft, *et al.*, "Communication-efficient online detection of network-wide anomalies," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pp. 134–142, IEEE, 2007.

[11] D. Brauckhoff, K. Salamatian, and M. May, "Applying pca for traffic anomaly detection: Problems and solutions," in *INFOCOM 2009, IEEE*, pp. 2866–2870, IEEE, 2009.

[12] C. Callegari, L. Gazzarrini, S. Giordano, M. Pagano, and T. Pepe, "A novel pca-based network anomaly detection," in *Communications (ICC), 2011 IEEE International Conference on*, pp. 1–5, IEEE, 2011.

[13] X. Li, F. Bian, M. Crovella, C. Diot, R. Govindan, G. Iannaccone, and A. Lakhina, "Detection and identification of network anomalies using sketch subspaces," in *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pp. 147–152, ACM, 2006.

[14] Y. Liu, L. Zhang, and Y. Guan, "Sketch-based streaming pca algorithm for network-wide traffic anomaly detection," in *Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on*, pp. 807–816, IEEE, 2010.

[15] X. Li, F. Bian, H. Zhang, C. Diot, R. Govindan, W. Hong, and G. Iannaccone, "Mind: A distributed multi-dimensional indexing system for network diagnosis.," in *INFOCOM*, Citeseer, 2006.

[16] Z. Lin, M. Chen, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices," *arXiv preprint arXiv:1009.5055*, 2010.

[17] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?," *Journal of the ACM (JACM)*, vol. 58, no. 3, p. 11, 2011.

[18] L. Xiong, X. Chen, and J. Schneider, "Direct robust matrix factorizatoin for anomaly detection," in *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pp. 844–853, IEEE, 2011.

[19] M. Thottan and C. Ji, "Anomaly detection in ip networks," *Signal Processing, IEEE Transactions on*, vol. 51, no. 8, pp. 2191–2204, 2003.

[20] A. O. Hero, "Geometric entropy minimization (gem) for anomaly detection and localization," in *Advances in Neural Information Processing Systems*, pp. 585–592, 2006.

[21] T. Ahmed, M. Coates, and A. Lakhina, "Multivariate online anomaly detection using kernel recursive least squares," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pp. 625–633, IEEE, 2007.

[22] K. Xie, X. Ning, X. Wang, D. Xie, J. Cao, G. Xie, and J. Wen, "Recover corrupted data in sensor networks: a matrix completion solution," *IEEE Transactions on Mobile Computing, DOI:10.1109/TMC.2016.2595569*, 2016.

[23] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[24] C. Eckart and G. Young, "The approximation of one matrix by another of lower rank," *Psychometrika*, vol. 1, no. 3, pp. 211–218, 1936.

[25] Z. Lu and Y. Zhang, "Penalty decomposition methods for l0-norm minimization," *preprint*, 2010.

[26] K. Xie, L. Wang, X. Wang, G. Xie, G. Zhang, D. Xie, and J. Wen, "Sequential and adaptive sampling for matrix completion in network monitoring systems," in *IEEE INFOCOM*, pp. 2443–2451, IEEE, 2015.

[27] K. Xie, C. Peng, X. Wang, G. Xie, and J. Wen, "Accurate recovery of internet traffic data under dynamic measurements," in *IEEE INFOCOM*, 2017.

[28] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proceedings of the twentieth annual symposium on Computational geometry*, pp. 253–262, ACM, 2004.

[29] "The abilene observatory data collections. http://abilene. internet2.edu /observatory/data-collections.html.,"