A New Approach to Multivariate Network Traffic Analysis

Jinoh Kim[1], Alex Sim[2]

[1] Texas A&M University, Commerce, TX, USA
[2] Lawrence Berkeley National Laboratory, Berkeley, CA, USA

A New Approach to Multivariate Network Traffic Analysis

Jinoh Kim[1,2], *Member, ACM, IEEE* and Alex Sim[2], *Senior Member, IEEE, Member, ACM*

[1]*Department of Computer Science, Texas A&M University, Commerce, 75428, USA*
[2]*Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA*

E-mail: jinoh.kim@tamuc.edu; asim@lbl.gov

Received month day, year

Abstract Network traffic analysis is one of the core functions in network monitoring for effective network operations and management. While online traffic analysis has been widely studied, it is still intensively challenging due to several reasons. One of the primary challenges is the heavy volume of traffic to analyze within a finite amount of time due to the increasing network bandwidth. Another important challenge for effective traffic analysis is to support multivariate functions of traffic variables to help administrators identify unexpected network events intuitively. To this end, we propose a new approach with the multivariate analysis that offers a high-level summary of the online network traffic. With this approach, the current state of the network will display patterns compiled from a set of traffic variables, and the detection problems in network monitoring (e.g., change detection and anomaly detection) can be reduced to a pattern identification and classification problem. In this paper, we introduce our preliminary work with clustered patterns for online, multivariate network traffic analysis with the challenges and limitations we observed. We then present a grid-based model that is designed to overcome the limitations of the clustered pattern-based technique. We will discuss the potential of the new model with respect to the technical challenges including streaming-based computation and robustness to outliers.

Keywords Network traffic analysis, multivariate analysis, time-series similarity, network monitoring

1 Introduction

Analyzing network traffic is an integral part of network operations and management for various purposes such as traffic engineering, resource provisioning, network security, usage statistics, and so forth. In particular, online traffic analysis is essential to identify any unexpected events in a real-time manner, including network anomalies, sudden changes, heavy hitters, etc, which would be an indication of cyber-attacks, misconfiguration of network devices, or network fault [1, 2, 3, 4]. For example, some anomalies may indicate performance bottlenecks with a huge number of simultaneous connections due to flash crowds, denial of service (DoS) attacks, or router/switch configuration failures. In addition, today's viruses and worms propagate very quickly, and it does not take more than several minutes to infect millions of machines on the Internet. Ideally, online analysis should be able to detect such indicative events in a timely manner to minimize the potential malignant impacts.

While online traffic analysis has been studied for a while, it is still intensively challenging due to several

[0]This work was supported in part by the Office of Advanced Scientific Computing Research, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231, and by the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists (WDTS) under the Visiting Faculty Program (VFP).

reasons. One of the primary challenges is the heavy volume of network traffic to analyze within a finite amount of time. A recent report forecasts that the Internet traffic will increase threefold over the next five years with an over 20% annual growth rate from 2015 to 2020[1]. The past observation already confirmed the traffic growth rate with a 27% annual increase of residential broadband traffic in 2007 [5]. With the today's computing trend, it is not hard to expect a greater use of mobile and IoT devices that will further contribute to the network traffic volume. For instance, recent DoS attacks were conducted by a botnet comprising hundreds of thousands of IoT devices[2]. To enable online traffic analysis against the large-scale data, streaming computation techniques have been widely studied, and the sketch [2, 6, 4, 7] is an example technique based on k-ary hashing. However, such existing methods are largely limited to a specific purpose such as a heavy hitter detection using a simple frequency counting method.

Another important challenge for effective traffic analysis is to support multivariate functions of traffic variables to help administrators identify unexpected network events in an intuitive way. Traditionally network traffic variables were independently analyzed, and combining the individual results is left to the administrators. For example, Opprentice [1] assumes three variables of key performance indicator (the number of page view, the number of slow responses and the 80-th percentile of search response time) in monitoring, and the work assumes that the variables are independently analyzed to identify anomalous events. The sketch mentioned earlier is also limited to give statistics for a single traffic variable, without any means to keep track of multiple variables in a combined way. The probabilistic density information has been considered to take a snapshot of the network traffic for change detection, but the current implementation is confined with a single dimensional variable due to the complication of the extension to multiple variables [3].

To address the above critical challenges to achieve effective online network traffic analysis, we propose a new approach that offers a high-level state summary of the network traffic from the multivariate features under consideration. With this approach, the current state of the network will display patterns compiled from a set of traffic variables. We define "network state" as a high-level summary of the network traffic with respect to the tracked variables to capture the current status of the network. The obtained pattern can be compared with another with the previously observed patterns. The detection problems in traffic analysis (e.g., change detection or anomaly detection) can thus be reduced to one of the pattern identification and classification problems. The key contributions of this paper can be summarized as follows.

- We present a new approach to multivariate, time-series network traffic analysis as an underlying technology for online monitoring applications, such as change detection and anomaly detection.
- We introduce the framework model for online network traffic analysis and our preliminary work using clustered patterns for network state representation and quantitative analysis with the challenges and limitations we observed.
- We present a grid-based approximation model for scalable, reliable-to-noise analysis with a quantitative measure to estimate the similarity of network states in different time windows.
- We demonstrate our proposed technique with the `tstat` network traffic measurement collection

[1] http://www.cisco.com/c/dam/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.pdf
[2] http://www.eweek.com/security/ddos-attack-snarls-friday-morning-internet-traffic.html
[3] https://www.es.net/

from the Energy Sciences Network (ESnet[3]) to see its applicability.

The preliminary results were reported in our past paper [8], and this paper extends it with further details and new experimental results with the ESnet measurement data. The organization of this paper is as follows. Section 2 provides a summary of the closely related studies. We present our framework model for online traffic analysis in Section 3, and introduce our preliminary approach based on clustered patterns with its potential and limitations in Section 4. We then discuss a new technique based on a grid approximation model for scalable, streaming-based analysis with our initial results in Section 5. In Section 6, we analyze the traffic trace collected from ESnet, the Department of Energy's dedicated research network, using the clustered pattern technique and the grid-based model. We discuss other topics including a brief comparison of the clustered patterns and grid-based techniques in Section 7. Finally, we conclude our presentation in Section 8.

2 Related Work

One of the widely studied methods for network traffic summarization is sketch, which is designed particularly for heavy-hitter detection based on the data streaming computation using a hash function [2, 4, 7]. Using a hash key extracted from the flow information (e.g., a 64-bit key composed by the source and destination IP addresses in the flow), the sketch maintains a hash table to keep the frequency information for each key. The statistics of the hashed results can then be used for the detection purpose (e.g., heavy-hitter). As discussed, the sketch technique is not capable for multivariate analysis and limited to give statistics for a single variable only.

In addition to sketch, other streaming data mining techniques as well as sampling methods and data reduction techniques were studied for network traffic analysis by frequency counting [9, 10, 11], histogram [12], clustering [13, 14, 15, 16], sliding windows [17, 18], wavelets [9, 19, 20], and dimensionality reduction [21, 22]. Many of these sampling methods provide a quick understanding of the monitored data stream, but characterizing accurate data patterns from the streaming data is still a challenge, especially with the recent hardware advances, which produces data records at a much higher rate. While the previous methods are limited to capture a single traffic variable and individual variables should be analyzed separately, the critical hurdle is how to combine analysis on multiple attributes for comprehensive analysis rather than single dimensional streaming data analysis as discussed earlier. The key difference of the proposed approach in this work is in the ability to capture the multivariate traffic attributes to provide a comprehensive view of the network state.

Visualization has also been widely accepted for network management with the power of intuitive analysis. CAIDA provides a tool for visualizing the Internet topology using the Autonomous Systems (ASes) information, which is helpful to understand the interconnectivity of routing systems over the global Internet.[4] Another tool provides a cyber-security map visualizing global cyber attacks with the source, target, and attack information in real time.[5] Additionally, the NeTraMark project [23] implemented tools for BLINK [24] and Traffic Dispersion Graphs [25], mainly for traffic classification.

3 Proposed Framework

In this section, we introduce our framework for online traffic analysis with its operational scenario. The proposed framework model is shown in Fig. 1.

The overall scenario is as follows. The raw traffic data comes into the first module ("pre-processor"

[4] `http://www.caida.org/research/topology/as_core_network/2014/`
[5] `http://map.norsecorp.com/#/`

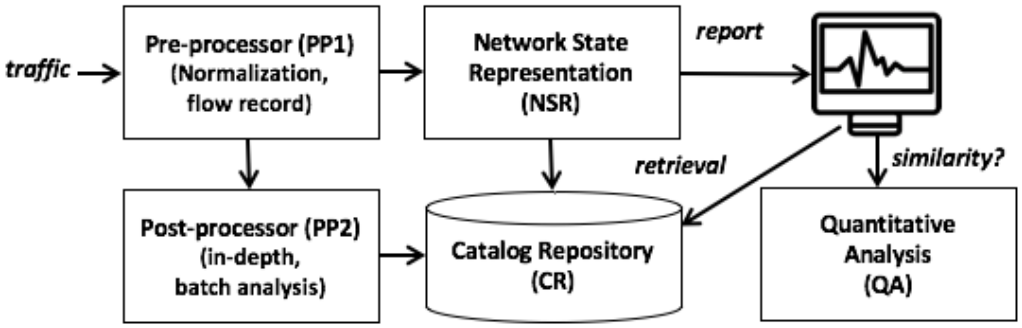


Fig. 1. Proposed framework for online network monitoring

or PP1) that performs the first-line of data processing including normalization and flow record construction. The output of PP1 is forwarded to 1) "post-processor" (or PP2) that performs in-depth analysis in a batch manner and 2) "Network State Representation" (NSR) that creates a pattern for each time window. We assume that the time domain is partitioned by a predetermined fixed interval, and NSR creates a pattern (or network state s_i) for the associated time window w_i from the collected data.

NSR then reports the created pattern to the administrator, and passes it to "Catalog Repository" (CR) that maintains the historic patterns ($S = \{s_i | i \geq 0\}$) for future reference. PP2 annotates the post-analysis information to the pattern stored in CR as soon as the batch processing is completed. The annotated information could be anomaly-related labels, traffic classification labels, etc., depending on the focus of analysis. The administrator can access CR to retrieve the patterns created in the past. For example, similar patterns to the current one can be searched to get an idea for interpretation. The component of "Quantitative Analysis" provides a tool to estimate the similarity of patterns in question. For example, $\Delta_{i,j}$ defines the degree of changes between two states s_i and s_j, as discussed in Section 4.

In this paper, we focus on discussing the core elements of NSR, QA, and CR in the framework. In Section 4, we introduce our initial observations with clustered patterns for network state representation and quantitative analysis, and then discuss the challenges and limitations.

4 Using Clustered Patterns

We initially studied clustering to capture the network state from the collected traffic data within a finite time interval. In this section, we briefly introduce the basic concept of the clustered patterns with the discussion of the challenges and limitations obtained from our observations. The details of this technique with two use cases of change detection and anomaly detection can be found from [26].

4.1 Clustering-Based Representation

We first describe how the clustered pattern represents a network state. For each time window in the monitoring process, a clustering is performed against the data points within the window, and the result of the clustering represents the high-level network state of that window. In this work, we employ the partitioning-based clustering to reduce the pattern information into a set of vectors, an element of which contains the cluster centroid position, population, and sum of squared errors. Specifically, we use the k-means technique for clustering that has maintained its popularity with the speed and simplicity, and can be scalable with parallelism (e.g., a parallel version of the k-means++ [27]).

This approach has the following potential benefits. First, the clustering method has the ability to combine multivariate attributes in a straightforward manner without an excessive extra computational cost, which has been one of the critical challenges for network traffic analysis. Second, it is flexible to configure the number of clusters ($k > 1$) regardless of the number of variables to be tracked, thus simplifying the analysis process. In addition, the network state information (with a set of vectors with length k) would be handy and possible to compare one another. Thus, comparing the similarity of given network states can be reduced to a problem of comparing two vectors.

We next discuss how to estimate the similarity of the clustered patterns under comparison in a quantitative manner. Measuring the similarity of two windows is the fundamental question in the change detection problem. We discuss the concept of "degree of change" (Δ) that estimates the changes between two clustered patterns representing the network states for the associated time windows. Δ is defined as a quantitative measure to estimate the similarity, calculated based on the move of the centroid positions between two time windows. This can be reduced to an assignment problem with the minimal cost (i.e., distance) between two patterns.

In detail, the clustered pattern of a time window W_i is a vector of clusters, $C_i = \{c_i^1, c_i^2, ..., c_i^k\}$. Similarly, the clustered pattern of W_j would be $C_j = \{c_j^1, c_j^2, ..., c_j^k\}$. Then $\Delta_{i,j}$ is defined as the minimal move of the centroid coordinates from W_i to W_j. Without knowing which cluster in W_i is mapped with one in W_j, we find a set of pairs showing the minimal move. Suppose a distance function $D : C_i \times C_j \rightarrow \mathbb{R}$. Then the problem is reduced to the assignment problem that finds a bijection $f : C_i \rightarrow C_j$ with the minimal distance function:

$$\Delta_{i,j} = \sum_{l \in C_i} D(l, f(l)).$$

We employ the Hungarian algorithm to solve this assignment problem, which has $O(k^3)$ of the computational complexity [28]. Note that it is true $\Delta_{i,j} = \Delta_{j,i}$ and $\Delta_{i,i} = 0$.

4.2 Example of Clustered Patterns and Analysis

To see how it works, we apply the clustering technique on a 16-hour trace excerpted from the UNIBS traffic trace, between 10AM on September 30, 2009 and 2AM on October 1, 2009 [29]. The dataset contains the information for network flows[6] with timing, and the ground-truth data with the associated application for each connection is provided [30]. As for statistics, the average number of flows is 789 flows/hour with a high degree of variance (min=20, max=7052).

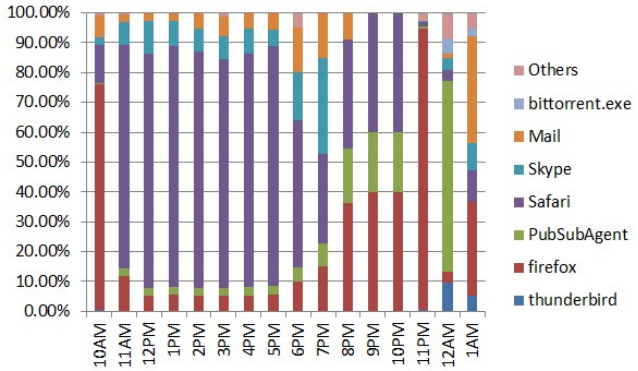


Fig. 3. Breakdown of applications for time windows (10AM–1AM), compiled from the ground-truth data in the UNIBS data trace.

Fig. 2 demonstrates the clustering results over 16 time windows (over 16 hours). From Fig. 2, we can see somewhat similar and dissimilar patterns over time. For example, the pattern for 10AM time window is quite different from the one for 11AM time window. In contrast, the clustered patterns from 11AM to 5PM

[6] A flow is identified with five tuples of source IP address, source port number, destination IP address, destination port number, and protocol in TCP/IP header

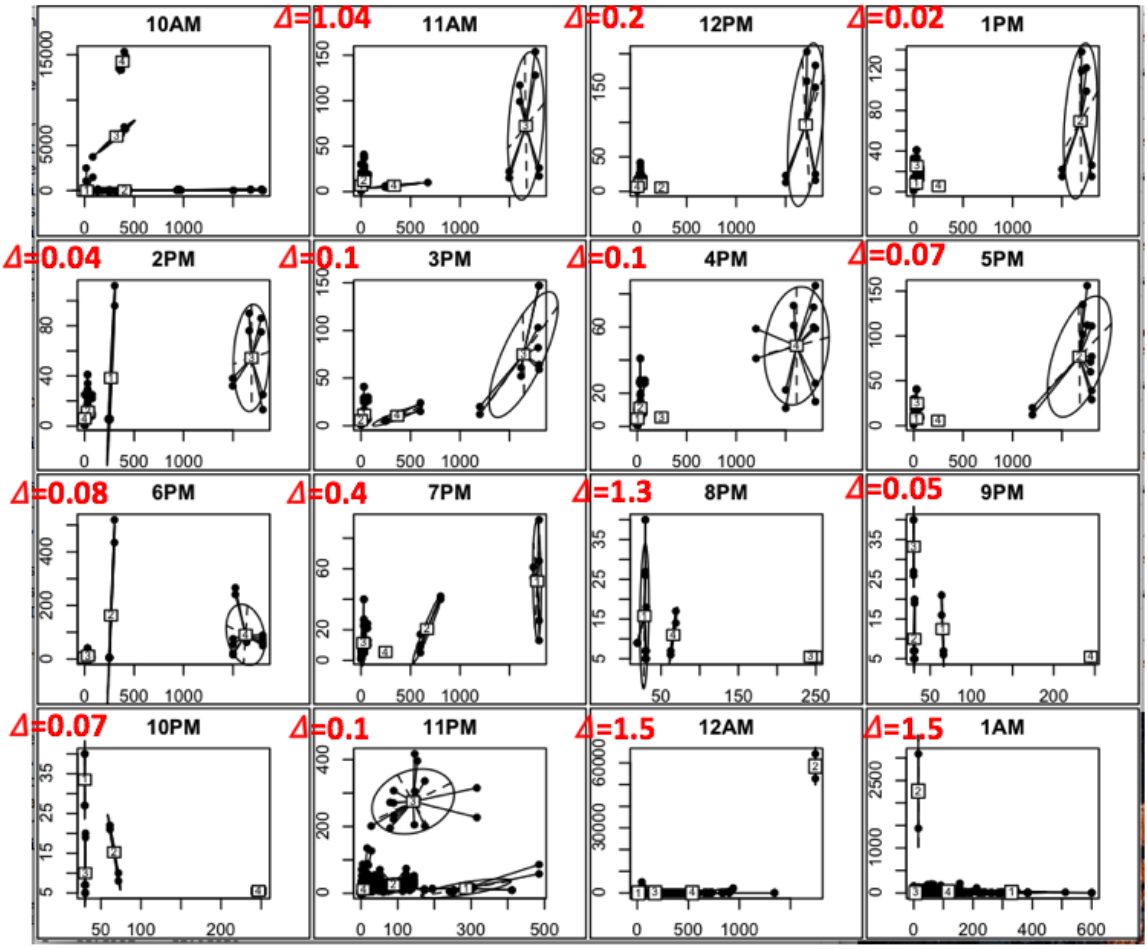


Fig. 2. Clustering results against a UNIBS data trace [29] for flow duration on x-axis and average number of packets in flow on y-axis. Note that cluster IDs were randomly selected by the clustering tool (R).

are visually similar. The three patterns for 8PM–10PM time windows are also resembling, whereas the last three time windows (11PM–1AM) have fairly distinctive patterns.

In Fig. 2, Δ is a quantitative measure to estimate the similarity, calculated based on the movement of the centroid positions between two time windows. Thus, it can be reduced to the assignment problem with the minimal cost (i.e., distance), which can be simply calculated using the Hungarian algorithm with $O(k^3)$ of the computational complexity [28]. We can see that the quantitative measure based on the centroid position movement shows strong correlations with the visual patterns.[7]

Fig. 3 shows the composition of applications for each window using the ground-truth information provided with the dataset. For example, the breakdown graph (Fig. 3) shows a high degree of similarity from 11AM to 5PM and from 8PM to 10PM, respectively, which agrees with the similarity of the clustered patterns in Fig. 2. On the other hand, there is a high degree of difference in the breakdown graph between 10AM and 11AM. Similarly, we can see huge differences from the windows of 11PM–1AM, suggesting strong correlations with the patterns in Fig. 2.

Our preliminary experiments show that the clus-

[7]We normalized the centroid position values based on the max coordinate value, and hence, Δ should not go beyond $k \cdot \sqrt{2}$ in this calculation (as one move cannot be greater than $\sqrt{2}$).

tered patterns would be helpful to summarize multivariate features in analysis to represent the associated network states. At the same time, we observed several limitations with this method as discussed in Subsection 4.3.

4.3 Challenges and Limitations

The clustered patterns are intuitive to interpret and lightweight with respect to the complexity since a pattern can be represented with a vector of clusters, each of which includes a centroid coordinate, sum of squared errors, and so forth. At the same time, we observed potential limitations. In this section, we discuss the primary challenges we observed from the clustering-based method: (1) robustness to sampling, (2) data stream processing, and (3) robustness to noise.

4.3.1 Robustness to Sampling

A key requirement for the network state representation is a high degree of scalability. In this regard, the clustered pattern method used in the preliminary study may not be a good option. For example, we observed that it takes 10 seconds to construct clusters with 16,000 data points in a commodity PC with the simple k-means that is known as a scalable method for clustering. As discussed, the traffic volume in a network becomes much heavier, and the MAWILab trace [31] contains 10,000 flows per second. To relax this concern, sampling could be considered like NetFlow [32] and sFlow [33]. However, we observed that sampling is not viable for clustered patterns, as can be seen from Fig. 4 that demonstrates the result of sampling. From Fig. 4, we can see that the random sampling results in a high degree of discrepancies, suggesting ineffectiveness for the continuous monitoring. Although not shown, we also computed Δs between sampled and non-sampled results and observed non-trivial variations.

4.3.2 Data Stream Processing

Another problem with the clustered pattern method is in its nature of the batch-style processing. That is, clustering can be executed when all the data points are available for the time window. However, data streaming processing is a desired property for online analysis with much greater scalability. One well-known streaming computation technique is the sketch [2, 6, 4, 7] that provides a probabilistic summary of a variable for analyzing network traffic data.

4.3.3 Robustness to Noise

A partition-based clustering for generating patterns may be in a high degree of sensitivity to outliers. Fig. 5 shows how only one or two outliers could significantly impact and construct somewhat different patterns. Although our initial observations with clustered patterns were interesting, the simple partitioning-based clustering would be ineffective to noises.

5 Grid-based Representation and Analysis

To relax the limitations of the clustered pattern-based technique, we investigated a grid-based structure [34] with its computational potential for data streaming support using approximation. In this section, we introduce the network state representation using a grid structure and a quantitative measure to estimate the similarity of the grid patterns. For the purpose of demonstration, we employ the KDDCup 1999 data ("kddcup.data_10_percent_corrected")[8] that has been widely used in the anomaly detection study. We formed 16 windows from the dataset, each of which contains 10,000 connections excerpted from the beginning of the data file in order. Table 1 shows the summary of the 16 windows with respect to traffic composition.

[8] `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`

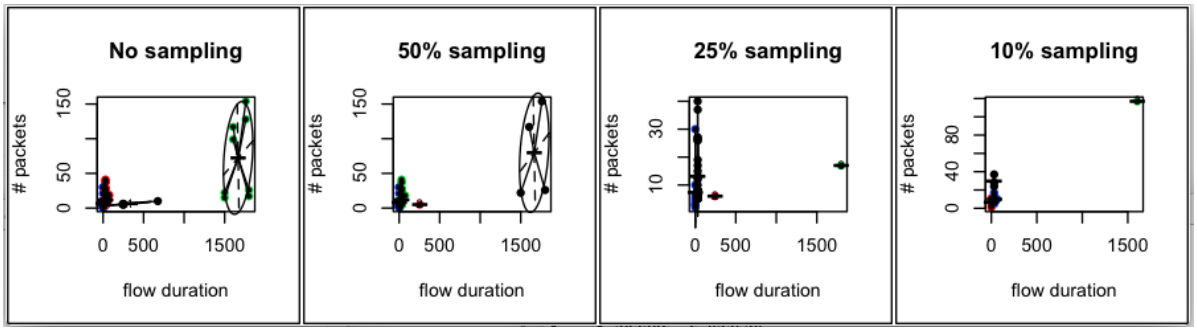


Fig. 4. Clustered patterns with random sampling: a sampling rate from no-sampling (leftmost) to 10% sampling (rightmost).

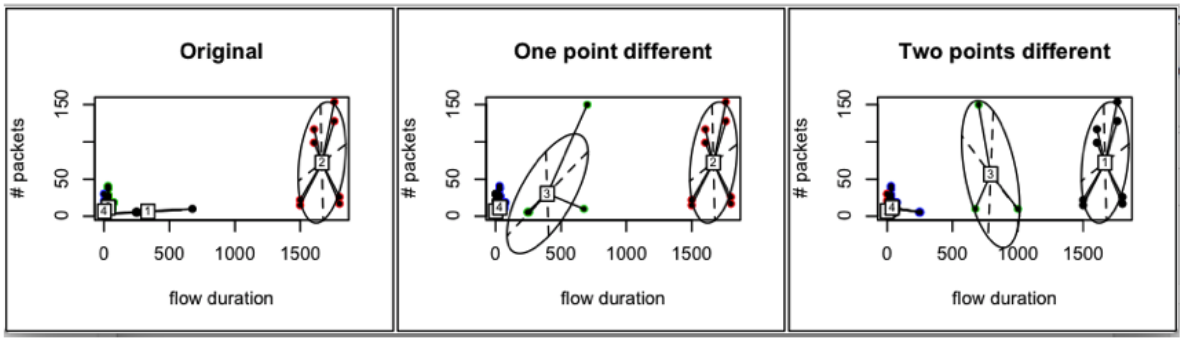


Fig. 5. Robustness to noise: even one or two different data points could impact significantly in the construction of patterns when using a partition-based clustering technique.

Table 1. Traffic composition (10,000 connections per window)

Window	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN	AO	AP
NORMAL	7787	8392	9837	9720	2230	1332	0	5786	9587	1566	4483	0	15	3526	6964	0
DOS	2209	1607	0	278	7531	8174	10000	4079	120	7846	5516	10000	9985	6474	483	10000
U2R	4	0	0	0	1	0	0	0	3	1	0	0	0	0	20	0
R2L	0	1	52	2	6	7	0	33	1	0	0	0	0	0	1023	0
PROBE	0	0	111	0	232	487	0	102	289	587	1	0	0	0	1510	0

5.1 Network State Representation

We consider a grid structure to represent a network state; hence, a network state consists of cells in a d-dimensional space ($\mathbb{R}^d$). The number of cells in a grid space is determined by the resolution. For example, there would be 1,024 cells in a 2D space if the resolution is 32 for both x and y axes. For each data point, there should be a mapping cell that contains an integer counter. The counter is simply incremented, and the density information can be easily inferred from the counters. Thus, it is straightforward to perform this technique in the streaming computation manner (rather than executing it in a batch computation). The complexity of this technique is proportional to the number of cells. Determination of the adequate resolution is essential in this technique, as too small resolutions may lead to losing the specific information while too high resolutions will yield too many empty cells in the space. We will discuss this again in Subsection 7.1.

The following pseudocode in Fig. 6 shows the steps to create a grid representation for a time window under the assumption of two-dimensional variables.

Step 1: Create a $M \times N$ grid structure

1. Grid[0..M-1][0..N-1] $\leftarrow$ 0

Step 2: For every connection record i:

1. Normalize the record
2. Calculate x and y indices
3. Grid[x][y] $\leftarrow$ Grid[x][y] + 1

Fig. 6. Pseudocode for the steps to create a grid representation for a time window under the assumption of two-dimensional variables.

Fig. 7 shows the representation of a single window (with 10,000 data points) in the KDDCup dataset. From Fig. 7, we can see that the cells occupied by the data points with the density level. The number of cells in this representation is (64 $\times$ 64) = 4,096.

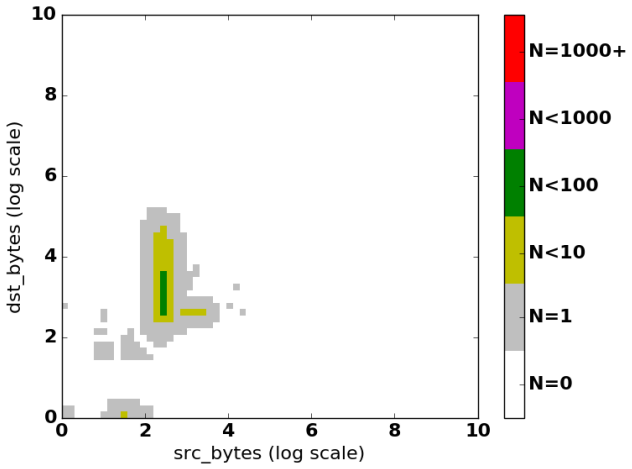


Fig. 7. An example of a grid-based representation of network state with a resolution of 64 $\times$ 64.

To learn more about the grid-based structure, we applied it for classification learning for the traditional anomaly detection [35]. Through the experiments with the original KDDCup dataset and the NSL-KDD dataset [36], we observed 98.5% and 83% of detection accuracy, respectively, which are comparable to the classical learning methods including decision tree and random forest. The learning complexity is very cheap and two orders of magnitude faster than the well-known classification techniques.

5.2 Quantitative Analysis

The proposed framework model includes a tool to estimate the similarity of patterns, which plays a key role to identify changes and anomalies. As an initial experiment, we established a simple measure that compares two grid spaces in questions, using a Jaccard coefficient model. The similarity index for two patterns of P_i and P_j is calculated as follows:

$$S_{i,j} = \frac{|P_i \cap P_j|}{|P_i \cup P_j|}.$$

Thus, $S = 1.0$ indicates that the two windows in question are identical, while $S = 0.0$ means that the windows are completely unrelated each other. We evaluated this simple measure against the 16 windows in Table 1.

Fig. 8 demonstrates the similarity matrix calculated by the Jaccard coefficient model. From the matrix, we can see some windows are highly similar, while some other windows (such as AG, AL, and AP with a full of DOS connections as shown in Table 1) are relatively less similar to others. Interestingly, the matrix shows that $S_{AG,AL} = 1.0$, whereas $S_{AG,AP} = S_{AL,AP} = 0.0$, although the windows contain DOS connections only. From the dataset, we found that the DOS attack in AG and AL is by Neptune, while it is by Smurf in AP, which results in the extreme similarity scores for those windows. As another example, the window of AC contains R2L and PROBE connections. Using the similarity measure, we observed $S_{AC,AI} = 0.83$ and $S_{AC,AH} = 0.79$ as the most similar windows, and both of AH and AI contain R2L and PROBE connections as well.

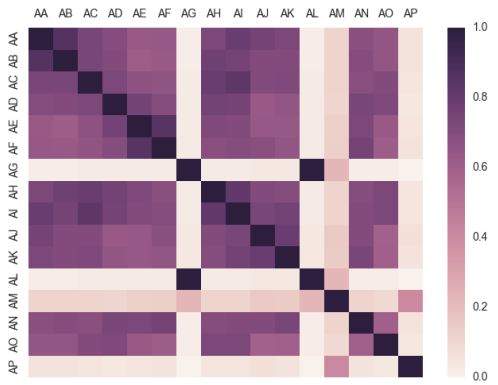


Fig. 8. Similarity matrix using Jaccard Index (AA–AP): a higher (darker) value indicates greater similarity between the two windows.

In our initial experiment, we did not consider the density information to compute similarity among windows. It may be interesting to consider density distributions and distribution comparison methods such as quantiles estimation [37], optimal transport [38, 39], and KS test [3] to establish a sophisticated measure to estimate the similarity.

5.3 Catalog Repository

As described, the traffic data in a time window is summarized into a pattern to represent the network state. The pattern is then stored to CR for future reference and statistics. The formats of the pattern representation may not be identical in NSR and CR. For ease of exposition, we refer to the format of the pattern in NSR as "representative pattern" and the other in CR as "reference pattern".

As discussed, we employed a partitioning-based clustering to obtain patterns in our initial work. A cluster created by the k-means technique is possibly characterized with a set of attributes, such as the centroid coordinate, the information related to the sum of squares, and the number of points in the cluster. And those cluster-related attributes were accounted to establish the similarity measures in our prior work. However, such a limited set of information may not be sufficient to well characterize a cluster. As a result, the network state represented with a set of cluster information would be too abstract to indicate the actual summary of the associated traffic data. Compared with this, the grid-based representation is basically rich with the cell-level information, including the occupancy and density information.

For the reference pattern, there would be two choices. The first choice is to use the identical method that is used for the representative pattern; the other is to implement a new model for the reference pattern. For the first option, there is no additional overhead to develop a new model for the reference pattern. However, the storage complexity could be a concern with the first choice. In detail, the storage requirement for each pattern will be $O(c^d)$ where c is the number of cells and d is the number of dimensions. Since the reference patterns can be referred in the future to search, for example, top-N patterns that are most similar to the current pattern, the storage complexity will be closely connected to the complexity for comparison. In this case, the complexity of $O(c^d)$ might be too expensive for a single pattern.

Fig. 9 demonstrates the use of the reference patterns, showing the two patterns with the greatest index scores compared with the window of AA. The result shows that $S_{AA,AB} = 0.86$ and $S_{AA,AI} = 0.79$. It indicates that 86% of the cells in the two patterns of AA and AB are commonly occupied by the data points, while 79% of the cells are common for AA and AI. The breakdown information shows a high degree of similarity between AA and AB, with a certain number of denial of service records that are roughly 20% of the total. We can also see that AI contains attack connections including DOS attacks; the breakdown shows a high degree of similarity with AA but it is smaller than the one between AA and AB.

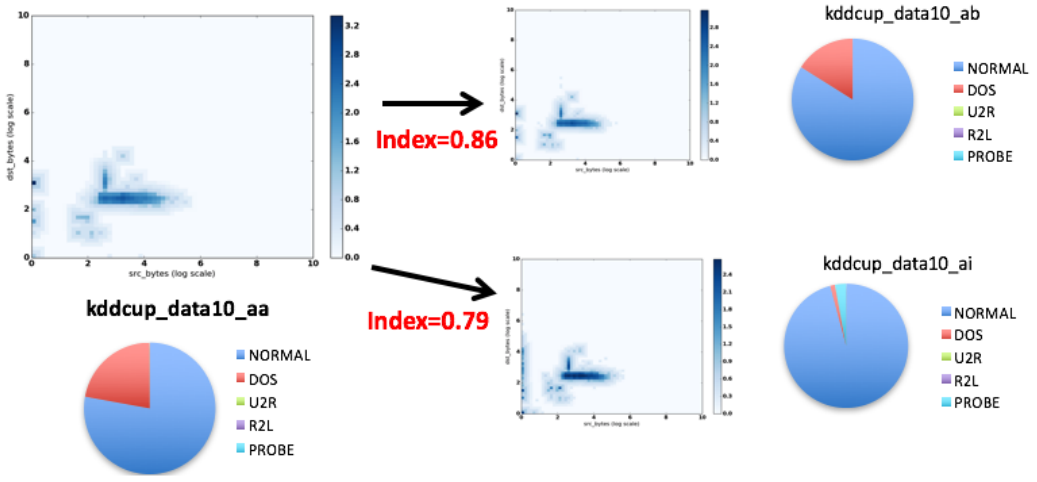


Fig. 9. Similarity estimation using the measure established based on Jaccard index.

6 ESnet Traffic Analysis

In this section, we demonstrate the use of our network traffic analysis techniques with a research network traffic measurement dataset, collected from ESnet that offers the high-bandwidth, reliable network connections among national laboratories in the US, universities and other research institutions. The traffic dataset contains the `tstat` logs, collected to analyze how various network tuning settings impact TCP behavior and network throughput. `tstat` rebuilds each TCP connection by looking at the TCP header in the forward and reverse direction. The details about the `tstat` tool can be found from [40].

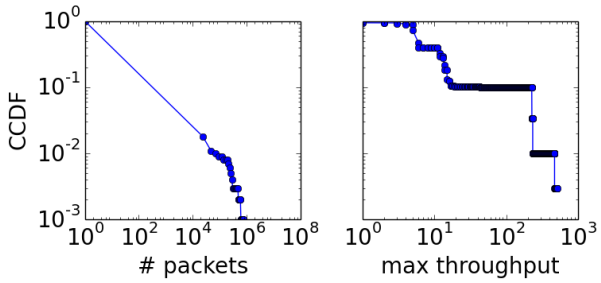


Fig. 10. The complementary cumulative distribution (CCDF) of the variables of the number of packets and the max throughput of connection in the one-day trace in the ESnet data.

In this experiment, we analyze the `tstat` data to monitor the network state change over time. We selected a subset of the measurement collection between 12:00PM and 2:40PM on May 9, 2016, to form 16 10-minute windows (labeled from BA to BP), without any bias in selection. The windows have 367 connections for each on average (min=157 and max=721). We chose two variables of <number of packets, max throughput> to evaluate the changes over time. Fig. 10 shows the complementary cumulative distribution (CCDF) of the two variables in a log-log scale. Due to a high degree of skewness for the above two variables, we performed normalization by applying a log function. We set the number of clusters to 4 ($K = 4$), chosen by the elbow method.

Fig. 11 demonstrates the clustered patterns for the 16 windows, and a group of windows shows somewhat similar patterns, for example, {BC, BD}, {BE, BF} and {BN, BO}. Fig. 12 shows the calculated degree of changes (Δs) for all-pair windows, and a lighter color indicates a smaller change (and thus more similar) between the two windows in comparison. The overall results show that the quantitative measure produces fairly relevant results with the visual representation.

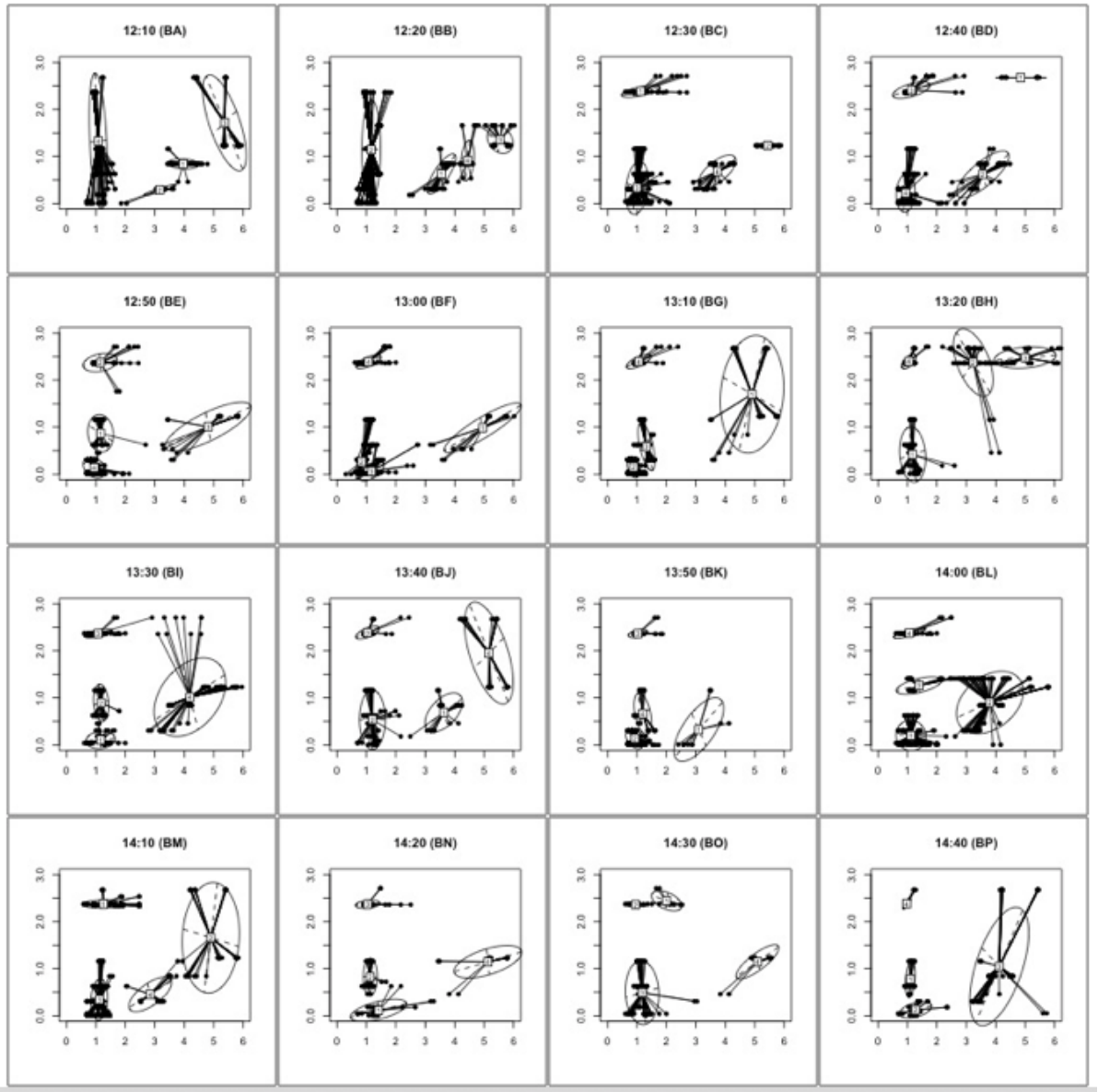
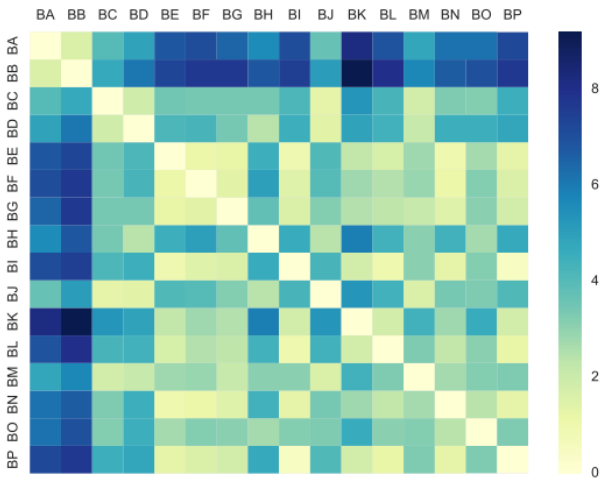
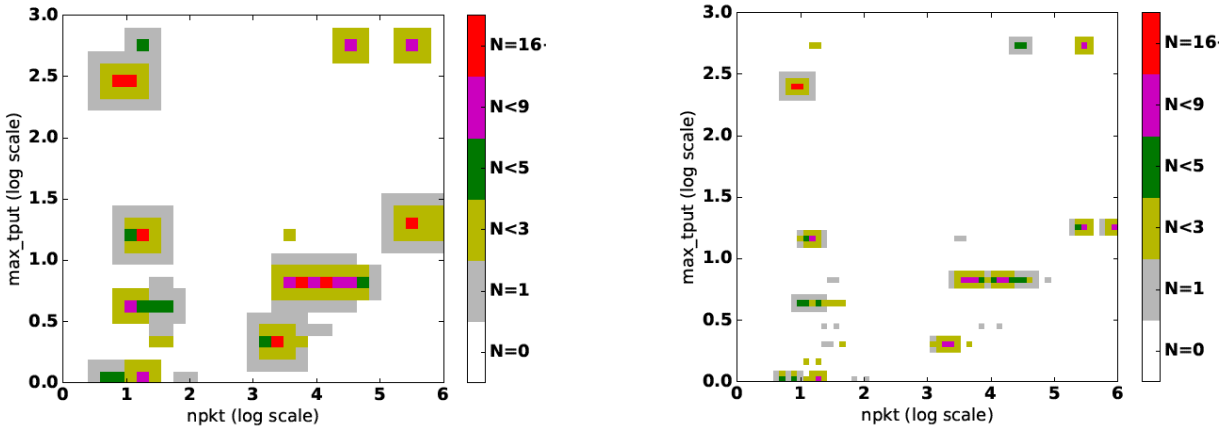


Fig. 11. Clustering results against the ESnet tstat data with two traffic variables of the number of packets on x-axis and max throughput on y-axis (log scaled for both x and y axes). Each pattern shows the summary of a 10-minute collection.

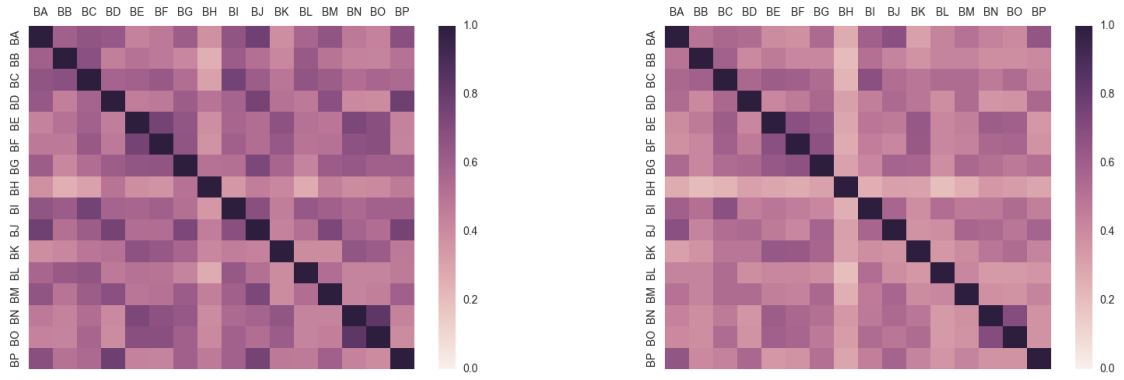


We next applied the grid-based model against the ESnet dataset. Fig. 13 demonstrates the grid-based representation for the dataset BA, with two different resolutions: (32 $\times$ 32) and (64 $\times$ 64) with respect to the number of cells for a window. As in Fig. 7, the exponential decay is applied to consider the impact of



(a) Number of cells = (32 × 32) (b) Number of cells = (64 × 64)

Fig. 13. Grid-based representation of BA with different resolutions.



(a) Number of cells = (32 × 32) (b) Number of cells = (64 × 64)

Fig. 14. Similarity matrix using Jaccard Index (BA–BP) with different resolutions: a higher (darker) value indicates greater similarity between the two windows.

density of each cell. As Fig. 13 indicates, choosing a resolution would be an interesting part in this study, although the two representations look closely similar to each other.

Fig. 14 shows the similarity matrix based on the Jaccard Index for the two grid representations in Fig. 13. We can see that the overall patterns are almost identical regardless of the resolutions. It would be interesting to take a look at the similarity matrix with the degree of changes (Δ's). We also observe that a high degree of correlation between Fig. 14 and Fig. 12. For example, three windows of BE, BF, BG made relatively small changes, and the similarity matrix shows a relatively high degree of similarity. Another example would be the windows of BN and BO, with a relatively small change and a relatively high degree of similarity. BA also shows the high Δ values with BE, BF, BK, BL, and BP, and the similarity matrix shows that BA is not much similar to BE, BF, BK and BL, but not to BP. Since the ESnet data does not contains annotation information, we leave further examination of accuracy for the grid-based model as a future task. As we discussed, however, the grid-based method is reliable to noises and straightforward for streaming-based computation, enabling scalable analysis of network traffic measurements.

7 Discussion

In this section, we compare the two presented models for traffic summarization, and then discuss the dimension reduction issue for pattern representation in a visual way.

7.1 Comparison of the Summarization Models

Table 2 provides a summary of the comparison between the clustered patterns and grid representation models. The main benefit of the grid structure model is that it is straightforward to create patterns in a data stream computation manner; comparatively clustering relies basically on the batch processing to expect accurate results. The clustered pattern model is cheap with respect to the storage complexity since a pattern is represented with a vector of clusters (including centroid coordinates, sum of squared errors, etc). On the other hand, the grid-based model is more expensive because it needs to maintain the cell information in a two-dimensional space. In addition, determining the resolution would be an open question. For example, if the resolution is (32×32), the number of cells is 1,024, while it is 4,096 with the resolution of (64×64), as in Fig. 13. Our future research tasks include the investigation of the impact of the resolution on the summarization and complexity.

7.2 Visualization and Dimension Reduction

Visualizing network states would be desired and beneficial for operators to recognize the state of the network in an intuitive way. Although online monitoring often limits the number of variables in analysis, there would be a need to keep track of more than two variables, which makes it complicated to visualize. One simple option is to create multiple plots in independent 2D spaces for each combination of variables. In that case, it needs $\binom{|V|}{2}$ plots where $|V|$ is the number of variables. This option would be neither intuitive nor scalable.

Another option is to use a dimension reduction technique such as PCA, t-SNE, autoencoder, and other reduction tools. Fig. 15 shows the result of clustering with PCA against the 16-hour trace used in Fig. 2. In this experiment, the data points were converted to create z-scores for standardizing. We observed that the results largely consent to ones in Fig. 2 with respect to similarity. Grid-based patterns can also be created by using a dimension reduction tool when we have more than two traffic variables in the analysis. Examining dimension reduction methods to see their efficiency for our traffic summarization is also one of the interesting research tasks.

Table 2. Comparison of clustered patterns and grid representation

	Clustered patterns	Grid representation
Robust to sampling	Weak	Moderate
Stream processing	Hard	Easy
Robust to noise	Weak	Robust
Representation complexity	Relatively cheap	Relatively expensive
	($O(k)$, where k is the number of cluster)	($O(c)$, where c is the number of cells)

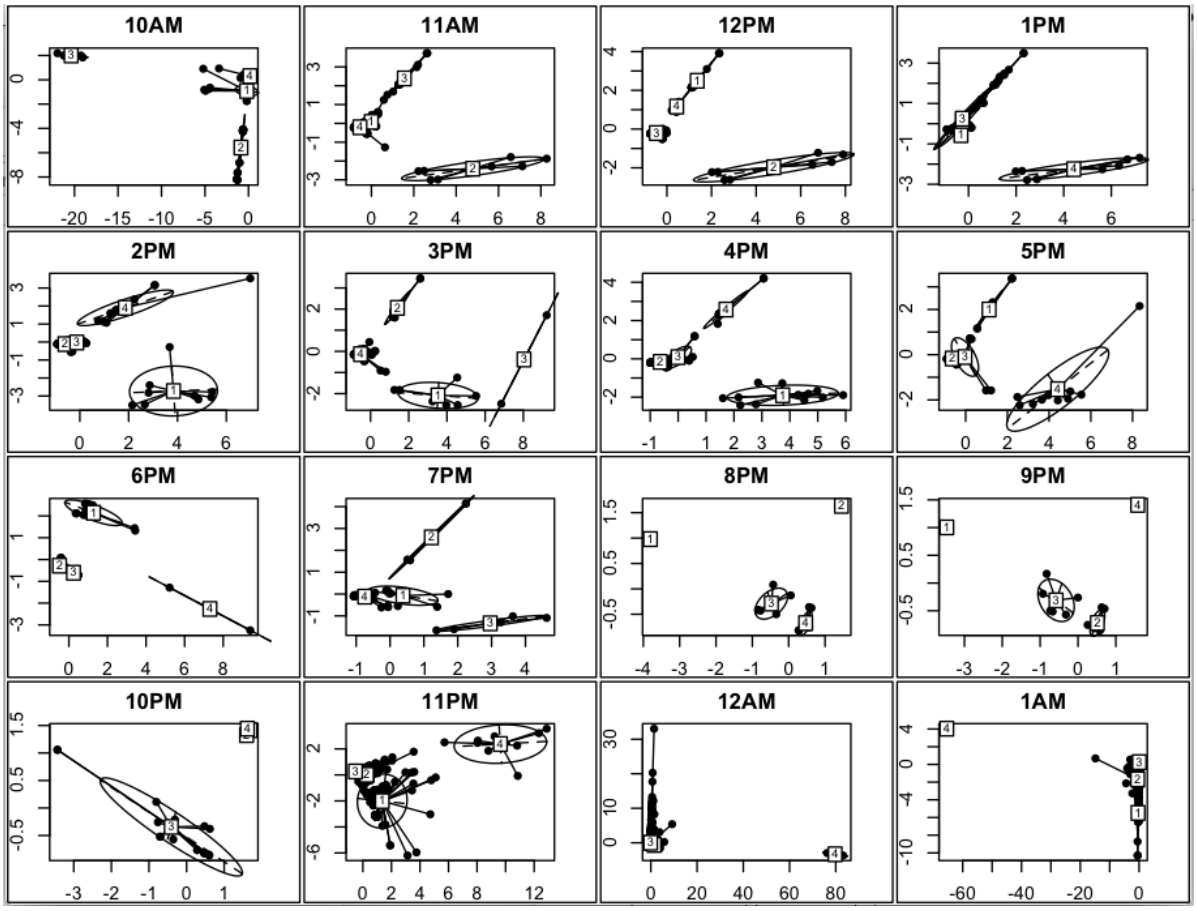


Fig. 15. Clustering with PCA with three attributes (the same 16-hour trace used in Fig. 2). The data points were transformed using standardization for effective PCA analysis.

8 Conclusions

This paper presents a new approach to the high-level online network traffic analysis using clustered patterns and grid patterns. The main goal of this study is to enable intuitive analysis of multivariate network traffic attributes at high level. We first demonstrated the use of clustered patterns with the observed challenges, and next presented a grid-based model to overcome the limitations of the clustered pattern-based technique, with particular respect to the streaming computation and robustness to noises. Finally, we demonstrated network traffic analysis using the presented techniques against the ESnet `tstat` trace.

The proposed approach has several important impacts. First, the multivariate approach for network traffic analysis has not been explored well, and the proposed method is new in the study area. Second, our work enables data streaming processing for effective online monitoring. Third, one of the core elements for scalability in this work is an approximation model that minimizes computational and storage complexity for the pattern-based network state representation and the catalog repository.

As this work is still ongoing, there would be many research tasks to be explored in the future. We are currently examining several possible methods for the representation of network states based on the grid structure and distribution models. For quantitative analysis, new measures are also needed to be defined for the newly established representation model. In addition, visualization and dimensionality reduction need to be investigated to efficiently support the high-dimensional multivariate analysis with the defined representation model, which will be helpful to enable an intuitive monitoring.

Acknowledgment

The authors would like to thank Brian Tierney at ESnet for the helpful discussion and support with the network traffic trace data.

References

[1] Dapeng Liu, Youjian Zhao, Haowen Xu, Yongqian Sun, Dan Pei, Jiao Luo, Xiaowei Jing, and Mei Feng. Opprentice: Towards practical and automatic anomaly detection through machine learning. In *Proceedings of the 2015 Internet Measurement Conference*, IMC '15, pages 211–224, 2015.

[2] Balachander Krishnamurthy, Subhabrata Sen, Yin Zhang, and Yan Chen. Sketch-based change detection: Methods, evaluation, and applications. In *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, IMC '03, pages 234–247, 2003.

[3] Jaesik Choi, Kejia Hu, and Alex Sim. Relational dynamic bayesian networks with locally exchangeable measures. Technical Report LBNL-6341E, Lawrence Berkeley National Laboratory, 2013.

[4] Minlan Yu, Lavanya Jose, and Rui Miao. Software defined traffic measurement with opensketch. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, nsdi'13, pages 29–42, 2013.

[5] Kenjiro Cho, Kensuke Fukuda, Hiroshi Esaki, and Akira Kato. Observing slow crustal movement in residential user traffic. In *Proceedings of the 2008 ACM Conference on Emerging Network Experiment and Technology, CoNEXT 2008, Madrid, Spain, December 9-12, 2008*, page 12, 2008.

[6] Robert Schweller, Ashish Gupta, Elliot Parsons, and Yan Chen. Reversible sketches for efficient and accurate change detection over network data streams. In *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, IMC '04, pages 207–212, 2004.

[7] Zaoxing Liu, Antonis Manousis, Gregory Vorsanger, Vyas Sekar, and Vladimir Braverman. One sketch to rule them all: Rethinking network flow monitoring with univmon. In *Proceedings of the 2016 conference on ACM SIGCOMM 2016 Conference, Florianopolis, Brazil, August 22-26, 2016*, pages 101–114, 2016.

[8] Jinoh Kim and Alex Sim. A new approach to online, multivariate network traffic analysis. In *Proceedings of the International Conference on Computer Communications and Networks (ICCCN'17), Vancouver, Canada*, 2017.

[9] G. S. Manku and R. Motwani. Approximate frequency counts over data streams. In *Proceedings of the International conference on Very Large Data Bases (VLDB)*, pages 346–357, 2002.

[10] S. Das, S. Antony, D. Agrawal, , and A. E. Abbadi. Cots: A scalable framework for parallelizing frequency counting over data streams. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, pages 1323–1326, 2009.

[11] S. Das, S. Antony, D. Agrawal, , and A. E. Abbadi. Thread cooperation in multicore architectures for frequency counting over multiple data streams. *Proceedings of the VLDB Endowment*, 2(1):217–228, 2009.

[12] S. Guha, N. Koudas, and K. Shim. Data-streams and histograms. In *Proceedings of the ACM symposium on Theory of computing*, pages 471–475, 2001.

[13] C. Aggarwal, J. Han, J. Wang, and P. Yu. A framework for clustering evolving data streams. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, pages 81–92, 2003.

[14] P. Domingos and G. Hulten. A general method for scaling up machine learning algorithms and its application to clustering. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 106–113, 2001.

[15] S. Guha, N. Mishra, R. Motwani, , and L. O'Callaghan. Clustering data streams. In *Proceedings of the the 41st Annual Symposium on Foundations of Computer Science*, pages 356–366, 2000.

[16] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O' Callaghan. Clustering data streams: Theory and practice. *IEEE Transactions On Knowledge and Data Engineering*, 15(3):515–528, 2003.

[17] M. Datar, A. Gionis, P. Indyk, and R. Motwani. Maintaining stream statistics over sliding windows. In *Proceedings of the ACM-SIAM symposium on discrete algorithms*, pages 635–644, 2002.

[18] B. Babcock, M. Datar, and R. Motwani. Maintaining stream statistics over sliding windows. In *Proceedings of the ACM-SIAM symposium on discrete algorithms (SODA)*, pages 635–644, 2002.

[19] Y. Matias, J. S. Vitter, and M. Wang. Wavelet-based histograms for selectivity estimation. In *Proceedings of the ACM SIGMOD international conference on Management of data (SIGMOD)*, pages 448–459, 1998.

[20] J. S. Vitter and M. Wang. Approximate computation of multidimensional aggregates of sparse data using wavelets. In *Proceedings of the ACM SIGMOD international conference on Management of data (SIGMOD)*, pages 193–204, 1999.

[21] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Locally adaptive dimensionality reduction for indexing large time series databases. In *Proceedings of the ACM SIGMOD international conference on Management of data (SIGMOD)*, pages 151–162, 2001.

[22] S. Papadimitriou, J. Sun, and C. Faloutsos. Dimensionality reduction and forecasting on streams. *Data Streams, Models and Algorithms*, 31:261–288, 2007.

[23] Suchul Lee, Hyunchul Kim, Dhiman Barman, Sungryoul Lee, Chong-kwon Kim, Ted Kwon, and Yanghee Choi. Netramark: A network traffic classification benchmark. *SIGCOMM Comput. Commun. Rev.*, 41(1):22–30, January 2011.

[24] Thomas Karagiannis, Konstantina Papagiannaki, and Michalis Faloutsos. Blinc: Multilevel traffic classification in the dark. *SIGCOMM Comput. Commun. Rev.*, 35(4):229–240, August 2005.

[25] Marios Iliofotou, Prashanth Pappu, Michalis Faloutsos, Michael Mitzenmacher, Sumeet Singh, and George Varghese. Network monitoring using traffic dispersion graphs (tdgs). IMC '07, pages 315–320, 2007.

[26] Jinoh Kim, Alex Sim, Sang Suh, and Ikkyun Kim. An approach to online network monitoring using clustered patterns. In *Proceedings of the International Conference on Computing, Networking and Communications (ICNC'17), Silicon Valley, CA, January 26-29*. IEEE, 2017.

[27] Bahman Bahmani, Benjamin Moseley, Andrea Vattani, Ravi Kumar, and Sergei Vassilvitskii. Scalable k-means++. *VLDB Endow.*, 5(7):622–633, March 2012.

[28] G. Ayorkor Mills-Tettey, Anthony Stentz, and SM. Bernardine Dias. The dynamic hungarian algorithm for the assignment problem with changing costs. Technical report, Carnegie Mellon University, East Lansing, Michigan, July 2007.

[29] Maurizio Dusi, Alice Este, Francesco Gringoli, and Luca Salgarelli. Using GMM and svm-based techniques for the classification of ssh-encrypted traffic. In *Proceedings of IEEE International Conference on Communications, ICC*, pages 1–6, 2009.

[30] F. Rgringoli, L. Salgarelli, M. Dusa, N. Cascarano, F. Risso, and k claffy. Gt: picking up the truth from the ground for internet traffic. *ACM SIGCOMM Computer Communication Review*, 39(5), October 2009.

[31] Romain Fontugne, Pierre Borgnat, Patrice Abry, and Kensuke Fukuda. Mawilab: Combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking. In *Proceedings of Co-NEXT'10*, pages 8:1–8:12, 2010.

[32] C. Estan, K. Keys, D. Moore, and G. Varghese. Building a Better NetFlow. In *SIGCOMM 2004*, pages 245–256, Portland, OR, Sep 2004. SIGCOMM 2004.

[33] Mea Wang, Baochun Li, and Zongpeng Li. sflow: Towards resource-efficient and agile service federation in service overlay networks. In *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, pages 628–635, 2004.

[34] Erich Schikuta. Grid-clustering: A fast hierarchical clustering method for very large data sets. Technical Report CRPC-TR93358, Rice University, 1993.

[35] Jinoh Kim, Wucherl Yoo, Alex Sim, Sang Suh, and Ikkyun Kim. A lightweight network anomaly detection technique. In *Proceedings of the International Workshop on Computing, Networking and Communications (in conjunction with ICNC'17), Silicon Valley, CA, January 26-29*, 2017.

[36] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of the kdd cup 99 data set. In *Proceedings of the Second IEEE International Conference on Computational Intelligence for Security and Defense Applications*, pages 53–58, 2009.

[37] Assaf Glazer, Michael Lindenbaum, and Shaul Markovitch. q-ocsvm: A q-quantile estimator for high-dimensional distributions. In *Proceedings of the Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013, Lake Tahoe, NV*, pages 503–511, 2013.

[38] Justin Solomon, Fernando de Goes, Gabriel Peyré, Marco Cuturi, Adrian Butscher, Andy Nguyen, Tao Du, and Leonidas Guibas. Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Trans. Graph.*, 34(4):66:1–66:11, July 2015.

[39] Vivien Seguy and Marco Cuturi. Principal geodesic analysis for probability measures under the optimal transport metric. In *Proceedings of the Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, Montreal, Quebec, Canada*, pages 3312–3320, 2015.

[40] M. Mellia, R. Lo Cigno, and F. Neri. Measuring ip and tcp behavior on edge nodes with tstat. *Comput. Netw.*, 47(1):1–21, January 2005.

Jinoh Kim received his Ph.D. degree in Computer Science from University of Minnesota, Twin Cities. He is currently an Assistant Professor of Computer Science at Texas A&M University-Commerce. The areas of research interests span from systems and networks, including large-scale distributed systems, big-data computing, network security and network traffic analysis. Prior to that, he was a researcher at the Lawrence Berkeley National Laboratory for 2010-2011 and an Assistant Professor of Computer Science at Lock Haven University of Pennsylvania for 2011-2012. From 1991 to 2005, he was a researcher and a senior researcher at ETRI (a national lab in Korea) participating in various research projects in system/network management and security.

Ales Sim is a Senior Computing Engineer at the Lawrence Berkeley National Laboratory. His current research interests are in data modeling and analysis methods, machine learning for large scale streaming data, and I/O optimization for exascale HPC applications. He has been actively involved in applications, such as accelerator simulation, astronomy, climate modeling, combustion modeling, fusion science, genomics, high energy physics, nuclear science, power systems, smart networking infrastructure, and others. He has led several projects from DOE and NSF as a PI or Co-PI. He has authored and co-authored over 130 technical publications, demonstrated software products in conferences, and released a few software packages under open source license.