A New Approach to Online, Multivariate Network Traffic Analysis

Jinoh Kim*†, Alex Sim†
Texas A&M University, Commerce, TX 75428 *
Lawrence Berkeley National Laboratory, Berkeley, CA 94720 †
Email: jinoh.kim@tamuc.edu, {jinohkim,asim}@lbl.gov

***Abstract*—Network traffic analysis has long been a core element for effective network operations and management. While online monitoring has been studied for a while, 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. Another important challenge to enable online monitoring is to support multivariate analysis of traffic variables to help administrators identify unexpected network events intuitively. To this end, we propose a new approach that offers a high-level summary of the network traffic with the multivariate analysis. With this approach, the current state of the network will display an abstract pattern compiled from a set of traffic variables, and the detection problems in traffic analysis (e.g., change detection and anomaly detection) can be reduced to a straightforward pattern identification problem. In this paper, we introduce our preliminary work with clustered patterns for online, multivariate traffic analysis with the challenges and limitations. 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 streaming-based computation and robustness to outliers.**

I. Introduction

Monitoring 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 monitoring is essential to identify any unexpected event in a real-time manner, including network anomalies, sudden changes, heavy hitters, etc, which would be the indication of cyber-attacks, misconfiguration of network devices, or network fault [7], [21], [22], [36]. 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 [26]. Ideally, online network monitoring should be able to detect such indicative events in a timely manner to minimize the potential malignant impacts.

While online monitoring has been studied for a while, 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. 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 [6]. 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 traffic. For instance, recent DoS attacks were conducted by a botnet comprising hundreds of thousands of IoT devices [2]. To enable online monitoring against the large-scale data, streaming computation techniques have been widely studied and the sketch [21], [23], [30], [36] 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 to enable online monitoring is to support multivariate analysis of traffic variables to help administrators to identify unexpected network events in an intuitive way. Traditionally traffic variables were independently analyzed, and combining the individual results is left to the administrator. For example, Opprentice [22] 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 above is 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 also 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 [7].

To address the above critical obstacles to achieve effective online network monitoring, we propose a new approach that offers a high-level summary of the network traffic from the multivariate variables under

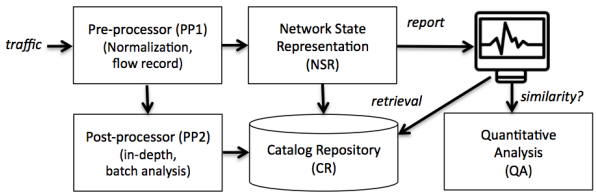

Fig. 1. Proposed framework for online network monitoring

consideration. With this approach, the current state of the network will display an abstract *pattern* 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 to another with the previously observed patterns. The detection problems in traffic analysis (e.g., change detection and anomaly detection) can thus be reduced to one of the pattern identification and classification problems.

In this paper, we first present our framework model for online network traffic analysis in the next section. In Section III, we briefly introduce our preliminary approach based on clustered patterns with its potential and limitations. We then discuss a new technique based on a grid approximation model for scalable, streaming-based analysis with our initial results in Section IV. Section V summarizes the closely related studies and we conclude our presentation in Section VI.

II. Proposed Framework

The proposed online network monitoring model is shown in Figure 1. The overall scenario is as follows. The raw traffic data comes into the first module ("Pre-processor" or PP1) that performs the first-line of data processing including normalization and flow record construction. The output of PP1 is forwarded to (i) "Post-processor' (or PP2) ' that performs in-depth analysis in a batch manner and (ii) "Network State Representation" (NSR) that creates a pattern for the time window. NSR reports the pattern to the administrator, and passes it to "Catalog Repository" (CR) that maintains the historic patterns 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 monitoring. The administrator can access CR to retrive the patterns created in the past. For example, she may want to search similar patterns with the current one to get an idea to interpret it. The component of "Quantitative Analysis" provides a tool to estimate the similarity of paterns in question.

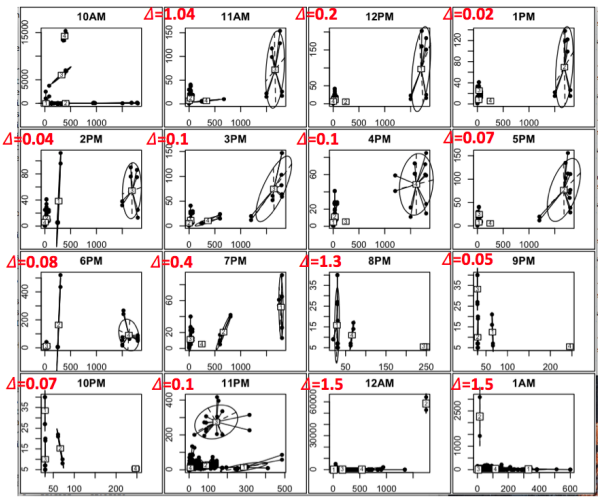

Fig. 2. Clustering results against a UNIBS data trace [12] for flow duration on x-axis and average number of packets in flow on y-axis. Note that cluster IDs were randomly selected by the clustering tool (R).

In the framework model, this paper gives an attention to the components of NSR and QA. In the next section, we introduce our initial observations with clustered patterns for network state representation and quantitative analysis, and discuss the challenges and limitations.

III. Using Clustered Patterns

We initially considered *clustering* to represent network states with its simplicity and the benefit of the aggregation of multidimensional attributes. In this manner, the clustered result is taken as a pattern that tells the network state in question. We briefly introduce the basic concept of the clustered patterns here, and the details of this technique with two use cases of *change detection* and *anomaly detection* can be found in [19].

To make clustered patterns, we employed a simple k-means clustering against a 16-hour trace excerpted from the UNIBS traffic trace, between 10AM on September 30, 2009 and 2AM on October 1, 2009 [12]. The data set contains the information for network flows[1] with timing, and the ground-truth data with the associated application for each connection is provided [29]. As statistics, the average number of flows is 789 flows/hour with a high degree of variance (min=20, max=7052).

Figure 2 demonstrates the clustering results over 16 time windows (over 16 hours). We would like to mention that the cluster IDs in the plots were randomly assigned by the clustering tool (R). From the figure, we can see somewhat similar and dissimilar patterns over time. For example, the pattern for 10AM time window is

[1]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

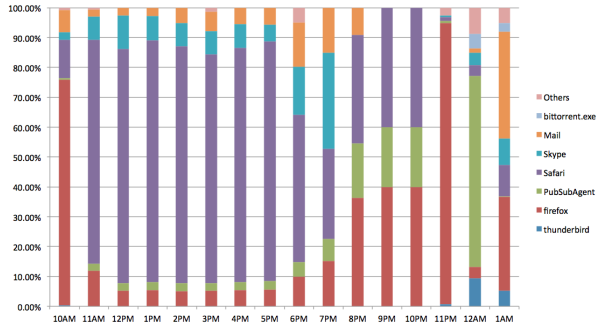

Fig. 3. Breakdown of applications for time windows (10AM–1AM), compiled from the ground-truth data in the UNIBS data trace

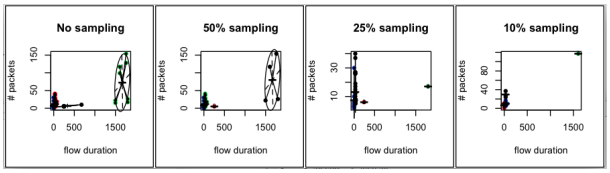

Fig. 4. Clustered patterns with random sampling: a sampling rate from no-sampling (leftmost) to 10% sampling (rightmost)

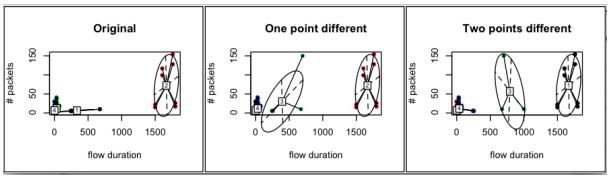

Fig. 5. Robustness to noise: even one or two different data points could impact significantly in the construction of patterns when using a partition-based clustering technique.

quite different from the one for 11AM time window. In contrast, the clustered patterns from 11AM to 5PM 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 the figure, Δ is a quantitative measure to estimate the similarity, calculated based on the movement of the centroid positions between two time windows. The quantitative measure also shows strong correlations with the visual patterns.

Figure 3 shows the composition of applications for each window using the ground-truth information provided with the data set. For example, the breakdown graph (Figure 3) shows a high degree of similarity from 11AM to 5PM and from 8PM to 10PM, respectively, which agrees with similarity of the clustered patterns in Figure 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 Figure 2.

Our preliminary experiments show that the clustered patterns would be helpful to summarize multivariate variables in analysis to represent the associated network states. At the same time, we observed several limitations with this method and we will next discuss two primary challenges.

a) **Data streaming processing:** 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 [14] contains 10,000 flows per second. To relax this concern, sampling could be considered like NetFlow [13] and sFlow [35]. However, we observed that sampling is not viable for clustered patterns, as can be seen from Figure 4 that demonstrates the result of sampling. From the figure, we can see that the random sampling results in a high degree of discrepancies, suggesting ineffectiveness for the pattern-based analysis. Although not shown, we also computed Δs between sampled and non-sampled results and observed non-trivial variations.

Another problem with the clustered pattern method is its nature of the batch-style processing. That is, clustering can be executed when all the data points are available. However, data streaming processing is a desired property for online monitoring with much greater scalability. One well-known streaming computation technique is the sketch [21], [23], [30], [36] that provides a probabilistic summary of a variable for analyzing network traffic data.

b) **Robustness to noise:** Another problem using a partition-based clustering for making patterns is a high degree of sensitivity to outliers. Figure 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.

IV. Grid-based Representation

To overcome the limitations of the clustered pattern-based technique, we investigated a grid-based structure. In this section, we introduce the network state representation using a grid structure and a quantitative measure to estimate the similarity of grid patterns.

A. Network State Representation

To examine the feasibility of the grid-based structure, we conducted a set of experiments with the KDDCup 1999 data ("kddcup.data_10_percent_corrected") [3]

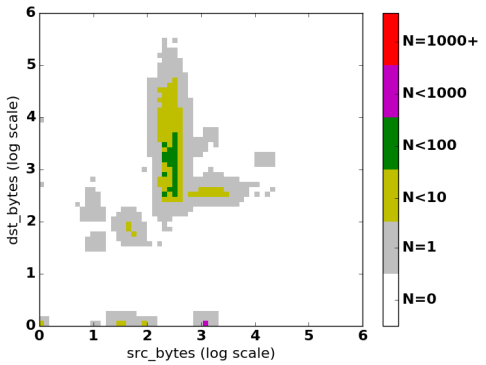

Fig. 6. An example of a grid-based representation of network state with a resolution of 64 × 64.

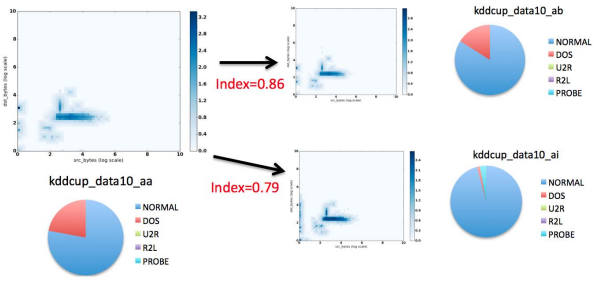

Fig. 7. Similarity estimation using the measure established based on Jaccard index

that has been widely used in the anomaly detection study. The dataset contains a set of connection information with the associated labels, which can be classified one of: denial of service ("DOS"), unauthorized access from a remote host ("R2L"), unauthorized access to root functions ("'U2R"), surveillance and other probing for vulnerabilities ("Probe"), and "Normal". We formed 16 windows from the dataset, each of which contains 10,000 connections excerted from the beginning of the data file in order. Table I shows the summary of the 16 windows with respect to traffic composition.

Figure 6 shows the representation of a single window (with 10,000 data points) in the KDDCup data set. The figure shows the cells occupied by the data points with the density level. The number of cells in the figure is (64 × 64) = 4,096. It is straightforward to perform this in the streaming fashion. The counter for the associated cell with the incoming data point is simply incremented and the density information can be easily calculated using the counters. In addition, the storage complexity is not expensive and proportional to the number of cells in the structure.

To learn more about the grid-based structure, we applied it for classification learning for the traditional anomaly detection [20]. Through the experiments with the KDDCup data sets [3], [33], 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.

B. 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 I. Figure 7 shows the two patterns with the greatest index scores compared to the window of AA. The result shows that $S_{AA,AB} = 0.86$ and $S_{AA,AI} = 0.79$. Note that AA and AB are the adjacent windows as the names indicate. The figure also shows the grid patterns and the traffic breakdown information for the windows. From the breakdown information, we observed 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.

Figure 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 I) are relatively less similar with 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 data set, 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.

While interesting, it may be enhanced by considering some other factors to define the similarity measure. In our initial experiment, for example, we did not consider the density information to compute similarity among windows. It may be interesting to consider density

TABLE I
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

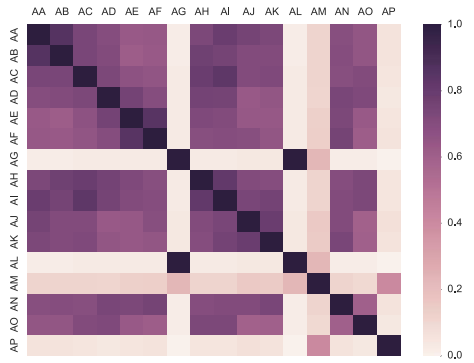

Fig. 8. Similarity matrix: the higher, the more similar the two windows are.

distributions and distribution comparison methods such as optimal transport [31], [32] and K-S test [7] to establish a sophisticated measure to estimate similarity.

V. RELATED WORK

The technique of sketch has been extensively studied for data streaming computation-based heavy-hitter detection [21], [23], [36]. The sketch basically makes use of a set of hash functions, and the incoming data points are counted using the key in question and the hash functions. For example, the key would be 64 bits with source and destination IP addresses. The statistics of the hashed results, such as *min, mean, quartile*, can then be referenced for the detection. Probabilistic models and samplings have also been considered to summarize network traffic. The work in [7] proposed a dynamic sampling technique to reduce the size of streaming data based on the difference of the probabilistic density. The authors employed the Kolmogorov-Smirnov test to measure the distance of two summaries observed in different time intervals. The past techniques are limited to capture a single traffic variable only, and individual variables should be analyzed separately. The key difference of the proposed approach is the ability to capture the multivariate traffic variables to provide a comprehensive view of the network state.

Probabilistic models and samplings have been considered for network traffic monitoring, especially for high volume traffic, and changing data patterns has been studied in streaming network traffic measurements [7]. Network monitoring could use streaming data mining techniques, and sampling methods and data reduction techniques were studied by frequency counting [8], [9], [24], histogram [15], clustering [4], [11], [16], [17], sliding windows [5], [10], random sampling [27], wavelets [24], [25], [34] and dimensionality reduction [18], [28]. Many of these sampling methods provide a quick understanding of the monitored data stream, but characterizing accurate data distribution from the streaming data is still a challenge, especially with the recent hardware advances, which produces data records at a much higher rate. In addition, the critical hurdle is how to combine multiple attributes for comprehensive analysis rather than single dimensional streaming data analysis as discussed earlier.

VI. CONCLUSIONS

This paper presents a new approach to the high-level online network monitoring 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.

The proposed approach has several important impacts. First, the idea of multivariate analysis for network monitoring is new and has not been explored well. 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 it is in the initial stage, there are many research items to be explored. 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 based on a new representation model.

VII. ACKNOWLEDGMENT

This work was supported in part 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), and by the Office of Advanced Scientific Computing Research, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

REFERENCES

[1] Cisco white paper: Cisco vni forecast and methodology, 2015-2020, 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] KDD Cup 1999 Data. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

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

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

[6] K. Cho, K. Fukuda, H. Esaki, and A. 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.

[7] J. Choi, K. Hu, and A. Sim. Relational dynamic bayesian networks with locally exchangeable measures. *LBNL Technical Report, LBNL-6341E*, 2013.

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

[9] 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.

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

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

[12] M. Dusi, A. Este, F. Gringoli, and L. 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.

[13] 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.

[14] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda. Mawilab: Combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking. In *Proceedings of CoNEXT'10*, pages 8:1–8:12, 2010.

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

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

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

[18] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Locally adaptive dimensionality reduction for indexing large time series databases. In *ACM SIGMOD*, pages 151–162, 2001.

[19] J. Kim, A. Sim, S. Suh, and I. Kim. An approach to online network monitoring using clustered patterns. In *Proceedings of the International Conference on Computing, Networking and Communications (ICNC 2017), Silicon Valley, CA, USA, January 26-29, 2017*. IEEE, 2017.

[20] J. Kim, W. Yoo, A. Sim, S. Suh, and I. Kim. A lightweight network anomaly detection technique. In *Proceedings of the International Workshop on Computing, Networking and Communications (in conjunction with ICNC 2017), Silicon Valley, CA, USA, January 26-29, 2017*. IEEE, 2017.

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

[22] D. Liu, Y. Zhao, H. Xu, Y. Sun, D. Pei, J. Luo, X. Jing, and M. Feng. Opprentice: Towards practical and automatic anomaly detection through machine learning. In K. Cho, K. Fukuda, V. S. Pai, and N. Spring, editors, *Proceedings of the 2015 ACM Internet Measurement Conference, IMC 2015, Tokyo, Japan, October 28-30, 2015*, pages 211–224, 2015.

[23] Z. Liu, A. Manousis, G. Vorsanger, V. Sekar, and V. 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.

[24] G. S. Manku and R. Motwani. Approximate frequency counts over data streams. In *VLDB*, pages 346–357, 2002.

[25] Y. Matias, J. S. Vitter, and M. Wang. Wavelet-based histograms for selectivity estimation. In *ACM SIGMOD*, pages 448–459, 1998.

[26] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver. The spread of the sapphire/slammer worm, 2003.

[27] R. Motwani and P. Raghavan. Randomized algorithms. In *Cambridge University Press*, 1995.

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

[29] 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.

[30] R. Schweller, A. Gupta, E. Parsons, and Y. 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.

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

[32] J. Solomon, F. de Goes, G. Peyré, M. Cuturi, A. Butscher, A. Nguyen, T. Du, and L. Guibas. Convolutional wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Trans. Graph.*, 34(4):66:1–66:11, July 2015.

[33] M. Tavallaee, E. Bagheri, W. Lu, and A. 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.

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

[35] M. Wang, B. Li, and Z. 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.

[36] M. Yu, L. Jose, and R. 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.