Classifying Elephant and Mice Flows in High-Speed Scientific Networks

Anshuman Chhabra[1,2] and Mariam Kiran[2]

[1]Division of Electronics and Communication Engineering,
Netaji Subhas Institute of Technology,
New Delhi, India
Email: achhabra@es.net

[2]Energy Sciences Network,
Lawrence Berkeley National Laboratory,
California, USA
Email : mkiran@es.net

Abstract

Complex science workflows usually involve very large data demands and resource-intensive computations. These demands need a reliable high-speed network to continually optimize its performance for the application data flows. Characterizing these flows into large flows (elephant) versus small flows (mice) can allow networks to optimize their performance by detecting and handling these demands in real-time. However, predicting elephant versus mice flows is extremely difficult as their definition varies based on networks.

Machine learning techniques can help classify flows into two distinct clusters to identify characteristics of transfers. In this paper, we investigate unsupervised and semi-supervised machine learning approaches to classify flows in real time. We develop a Gaussian Mixture Model combined with an initialization algorithm, to develop a novel general-purpose method to help classification based on network sites (in terms of data transfers, flow rates and durations). Our results show that despite of variable flows at each site, the proposed algorithm is able to cluster elephants and mice with accuracy rate of 90%. We analyzed NetFlow reports of 1 month from 3 ESnet site routers to train the model and predict clusters.

Keywords: Elephant and Mice Flows, Wide Area Networks, Machine Learning, Gaussian Mixture Models

1. Introduction

Scientific networks host applications which process complex climate models, nuclear research and genomics data. These applications require high network capacity, high speed data delivery and guaranteed connectivity, to ensure all scientists communicate as efficiently as possible. For example, the Large Hadron Collider produces data at the order of petabytes per second, and usually requires high-speed transfers to multiple research sites and storage facilities across the world [1]. In another example, genomics use cases require advanced bandwidth reservation across network links for dedicated quality for their data delivery. As networks and number of applications grow and become more complex, managing the diverse demands across distributed networking sites in becoming increasingly cumbersome. Researchers are investigating software defined networking (SDN) to provide flexible and agile networks being controlled via centralized software. However, SDN uses specific protocols (e.g. OpenFlow) that can configure only some devices via special forwarding rules [2]. SDN capabilities can be leveraged with active monitoring, prediction and informed decision-making to pre-compute and anticipate network service demands to intelligently support complex distributed science applications.

To understand application demands, an additional precursor is required to study and analyze traffic flow patterns across various network links and sites. Researchers have been trying to understand flows by exploring network statistics, combined with machine learning algorithms to predict traffic patterns at various times of the day [3]. This analysis is fundamental to developing self-optimizing network architectures that can recognize and predict traffic demands for diverse applications. For example, simple calls send short (mice) flows in a data center, that are bursty and latency-sensitive applications, whereas big data file transfers tend to have long-lived (elephant) flows where transfer throughput in more important than latency. If not managed efficiently, elephant flows can fill network buffers, causing queuing delays and packet drops. On the other hand, mice flows are more difficult to predict and more dynamic in nature.

Network routing protocols such as shortest path first or multi-path routing are being used to optimize path usage based on arriving flow sizes. Researchers usually identify elephant versus mice flows, to apply different routing approaches to manage their transfers. These include assigning different flows to different queues [4], splitting flows across links [5] or applying policy-based routing which is provided as rules [6]. However, accurately predicting what is an elephant versus a mice flow, as they arrive is nearly impossible due to the different kinds of experiments sending data across the networks.

In the past, machine learning techniques have been used to automate and classify traffic for intrusion detection and traffic profiling. Researchers classify traffic based on number of packets transfered, IP addresses, TCP traces, file sizes and flow durations [7]. These techniques include Naive Bayes Theorem, decision trees, support vector machines and random forest. Classifying host traffic and flows can be coupled with tools to perform real-time path optimization [8] [9]. These results can statistically characterize small and large flows, but most current research is based on simplistic file transfer experiments or simulation results. It is increasingly difficult to obtain real traffic logs from network administrators due to security and industry policies.

Apart for commercial networks, in research networks, such as ESnet (Energy Sciences Network) and Internet2, the challenge goes further where large data transfers across multiple sites are more common and can throttle the network if not managed. Scientists located at different sites send variable flows across the network topology. This topology is usually pre-decided which is difficult to dynamically change depending on the traffic demands during the day.

The aim of this work is to develop a flow characterization technique that can classify traffic patterns based on site behaviors. We predict that different sites receive different kinds of traffic during the day, month and year. Where other approaches have used a pre-decided threshold to predict flows [10], these thresholds will not be able to hold for all sites in general. Because each site is different, we propose to investigate unsupervised machine learning techniques to help classify elephant and mice flows per site. Unsupervised classification can create traffic clusters without any predefined notions on how the traffic should look. These results can lead to unique patterns observed across each site. This work lays the foundation to design automated tools that can be applied across all network sites and routers.

Elephant flows can pose a serious problem for networks where they can slow down transfers and impede service quality for real-time mice flows. However, most research uses predefined rules to classify these flows. Table 1 shows the variability among flows across three different sites. Having one algorithm that can classify flows, unbiased based on the individual sites is a challenge. Machine learning algorithms have been successful in vision and text recognition by deducing common images and patterns across multiple pictures. These techniques can be employed in network traffic analysis to understand how flows differ across sites and whether commonalities can be found. Our specific contributions in this paper are as follows:

- We design and implement a machine learning algorithm which can perform traffic cluster analysis based on Netflow records from three different site routers over the ESnet network. The algorithm uses the Gaussian Mixture Model coupled with a novel initialization algorithm to identify cluster patterns across WANs. The proposed algorithm is a generalized algorithm that can be applied across different sites with varying flows. It accounts for the flow distribution and trains itself uniquely per site behavior.

- We investigate unsupervised and semi-supervised machine learning techniques in terms of their suitability to traffic cluster analysis. Our results are able to show that unsupervised learning is beneficial for traffic analysis where little prior knowledge is known. This can help find patterns which network administrators are unaware of, for how their networks are being used.

Our results have shown that this work is imperative to lay the foundational research for develop self-autonomous networks. Allowing networks to self-learn and understand their own traffic can help optimize their behavior for future flows. This paper has been organized as follows: Section 2 presents the motivation of this work and the impact these methods will have on flows. Section 3 discusses the background and related work in the area of flow classification. This section also discusses the different kinds of machine learning clustering algorithms and their suitability to different problems. Section 4 presents the proposed methodology, explaining the algorithm developed and how it is applied on Netflow records. Section 5 presents the results of the clustering analysis along with a comparison to K-means clustering analysis to show the suitability of the algorithm. Finally, Section 6 presents a discussion and conclusions with how this work is applicable to future research in developing self-autonomous networks.

Site	Mean (Sz)	Max (Sz)	Mean (Dur)
Router1	0.1547	25.6	23.1965
Router2	0.0313	36.4	4.1433
Router3	0.0238	72.5	6.6344

Table 1: Variability of flows across sites. Size (Sz) is in GB, Duration (Dur) is in seconds. The router names have been anonymized.

2. Motivation

Research-based networks such as Internet2, GEANT and ESNet host users that regularly move exceptionally large datasets to local supercomputing clusters for further analysis or simply storing data. These experiments are computationally expensive and highly dependent on the time for processing and data arrival speeds. Any discrepancies across the network can compromise the experiments hindering research. Characterizing elephant and mice flows in these WANs can lead to following benefits:

- In cases of substandard network performance, managing elephant flows could lead to faster and more efficient diagnosis of source(s) of the problem. This allows network engineers to figure out the problem causing elephant flows and rectify any buffering or slow quality at the client-end.
- Smaller size flows are bursty and cause inadvertent jitter leading to delays in the network. If these flows are identified, they can be subjected to bandwidth throttling or be isolated from other real-time flows to allow for higher network utilization and shorter delays.

Currently, flow classification is either done based on human experience or using thresholds based on file size and bytes transfered, to identify what is a big versus a small flow. This leads to knowledge and architecture dependency and sometimes unoptimized networks that are difficult to grow as number of users and devices increase. Automating the classification can allow for more advanced tools to be developed which can train and learn based on their own unique traffic patterns on how to manage networks efficiently. This requires a general model that can learn and train efficiently, requiring less processing time and improved accuracy results.

3. Background and Related Work

Elephant flows or long-lived TCP flows, even if less in number can potentially fill network buffers end-to-end [11]. These can lead to queuing delays, affect latency-sensitive flows and smaller bursty flows (known as mice). Strategies in traffic engineering often propose to optimize network links by identifying and handling elephant and mice flows differently [12]. Some of these approaches include sorting flows into different queues, applying different routing approaches or even splitting longer flows into smaller ones [11, 13].

Zhang et al. [14] showed a correlation between flow sizes and rates. Therefore, the idea that Elephant or high-rate flows could be identified using flow sizes is well founded and theoretically correct [11].

Researchers have argued modifying hardware configurations to dynamically route flows as they arrive. However, large and small size flows are difficult to predict in advance and can lead to difficulties in updating hardware configurations [15, 16, 17, 18, 3]. Other researchers have proposed using sampling methods to identify flow characteristics [17, 19, 20, 21]. However, these sampling algorithms reduce the available memory and the accuracy achieved is restricted.

To classify elephant flows in particular, a hybrid network traffic engineering system (HNTES) was used to explore Netflow data to study flow source, destination addresses and reconfigure firewall filters to redirect flows over intra-domain virtual circuits. Deployed over four ESnet routers, HNTES was able to show a 91% improvement [22] in redirecting large flows. However, this approach used an offline training approach to analyze and make decisions [10]. Liu et al. [23] validated their results using scientific data transfer logs but showed a significant throughput variance in flows. These results also highly depended on the source and destination involved [24]. But, as highlighted in [25], applications may use different port numbers and protocols depending on compute resources involved. This makes it unreliable to just base the results on the ip addresses involved.

Predicting flow sizes is a difficult problem, especially as flows vary from site to site. Using thresholds or differentiating characteristics can help understand the various arriving flows, but can lead to extremely biased results which are not applicable across all sites. Machine learning methods have been known to learn characteristics through labeled and unlabeled data sets in image and text recognition exercises. Similar principles can be used in network data to identify what are the features of flows when divided based on flow rates and sizes. Researchers have been exploring these techniques in the past. Nguyen et al. [26] discussed a survey of statistical classification techniques in traffic patterns by combining IP networking and data mining techniques. They highlighted the need for reliable data sets and key elements for accurate classifiers to be developed. Additionally, Ibrahim et al. [27] discussed the problem of dataset validation and training issues in traffic classification. The authors discussed the problem of training based on real online traffic and dependency of where the data is captured can affect the reliability of WAN results. In terms of techniques used decision tree classification based on latency and throughput [28], Naive-Bayes Tree based on packet-length and payload classification [29], Naive-Bayes based on applications [30] or using support vector machines based on packet headers [31] have demonstrated good results.

In this paper, we particularly explore unsupervised methods to study traffic patterns and aim to produce a generalized algorithm to accurate predict classes based on per site traffic. Our technique is based on bytes sent and file size, which in comparison to other techniques is a new method and aims to develop tools that can perform real-time classification in the future.

4. Proposed Solution and Contributions

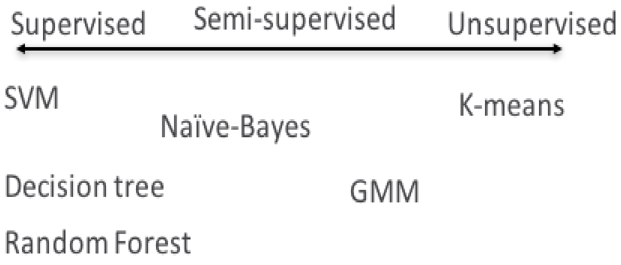

Figure 1: Machine learning techniques ranging between unsupervised and supervised techniques.

Figure 1 summarizes the various classification techniques used which belong between supervised to unsupervised areas. Supervised techniques use some knowledge about datasets (such as labeled data) to group data into cluster. Unsupervised techniques start without any knowledge of datasets, identify features and cluster similar records into unique sets. Algorithms which use a mix of techniques fall under semi-supervised area.

For flow classification, we assumed to start with unsupervised techniques to help cluster flows into similar kinds of two clusters. The aim here is to produce two distinct cluster, by which we can analyze the difference between elephant and mice flows. K-means algorithm is easy to understand and sensitive to multiple data dimensions clustering data sets based on the information available in the record. Initial experiments with k-means showed that this algorithm was not able to produce two distinct clusters (Figure 2). This is because the flows contained some information on file sizes which were difficult to group.

Based on the results, we modified the clustering algorithm with an initialization step to train the data based on local datasets available. This is shown in Figure 3. Details of the technique are as follows.

4.1. Terms and Definitions

1. Flows: We define flows as 5-tuple identifier of Source IP Address, Destination IP Address, Source Port Number, Destination Port Number and Communication Protocol (TCP, UDP, ICMP). A flow with a particular set of these parameters is considered unique and no two flows will possess the same parameters at any given point of time.
2. NetFlow Record (N_r): NetFlow data reports captured (with the extended option) at the routers possess the following entries relevant to our problem,

$$N_{\mathrm{r}} = \{t, d, IP_{\mathrm{s}}, P_{\mathrm{s}}, IP_{\mathrm{d}}, P_{\mathrm{d}}, C_{\mathrm{p}}, p, b, bps\} \quad (1)$$

where t is the Unix timestamp signifying when the flow was first seen, d is the duration for which the flow persisted on the link, $\{IP_s, P_s, IP_d, P_d, C_p\}$ is the 5-tuple flow ID (source IP, source port number, destination IP, destination port number, protocol), p represents the number of packets, b is the number of bytes of the transfer and bps represents the bytes per second. For our work, we use only size (bytes) of the data transfer and durations as the feature set.

4.2. Gaussian Mixture Model (GMM) Description

Gaussian Mixture Models (or Expectation Maximization) algorithms are a class of probabilistic unsupervised clustering algorithms. These assume clusters to belong to normal distributions.

We assume that elephant and mice flows together form a Gaussian mixture and can be represented as

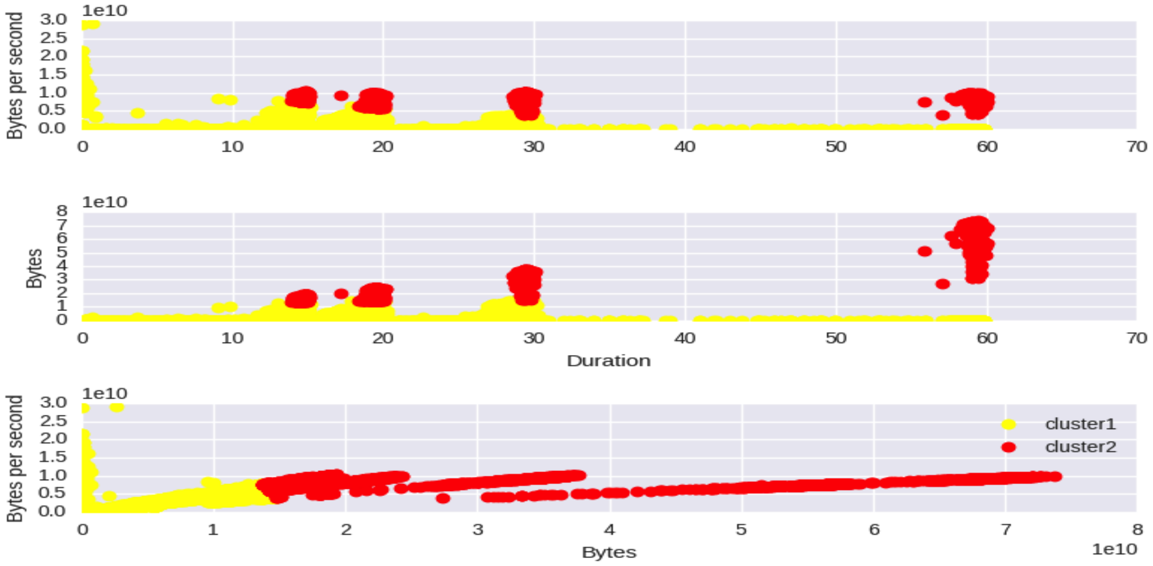

Figure 2: k-means clustering of flows on Router3.

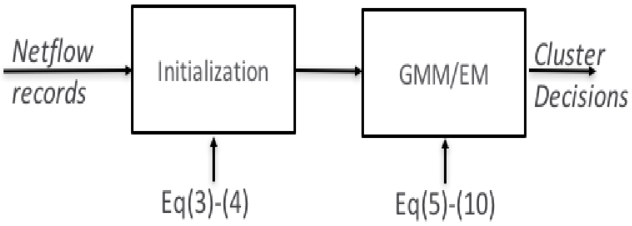

Figure 3: Method description.

Gaussian distributions. The probability distribution can thus be as follows,

$$p(X) = \pi_e \mathcal{N}(X|\mu_e, \Sigma) + \pi_m \mathcal{N}(X|\mu_m, \Sigma) \tag{2}$$

where π_e and π_m are the mixture coefficients for elephant and mice flows respectively. μ_e and μ_m are the means of the elephant and mice normal distributions.

The next steps is to perform the classification by implementing the initialization step. The reasons for GMM initializations are as follows,

- Since the GMM can be understood as a 'soft' K-Means algorithm for clustering, initializations can be similar to the way K-means initializes such as using a distance metric. Here the initialized cluster centers are chosen to be the samples that are closest to the randomly chosen centers.
- Cluster centers are also randomly chosen without any prior distance metrics.

Both reasons for initializing cluster centers prove to be unsuitable for flow classification. Firstly, if points are chosen on random, it will cause the clustering to be random. Secondly, if initialized as K-means using a distance metric, it does not apply to varying data sets which lead to different cluster properties across sites. Therefore we implemented an initialization step which uses the data records to generate cluster centers.

4.3. Proposed Algorithm

Here X represents the set of all relevant parameters (sizes and rates) of the flows that are seen by the system at a given time.

1. Initialization Step: The mixture coefficients initializations are predefined. The covariance are initialized as in regular EM. However, cluster means are initialized according to the formula,

$$\mu_e = \mu + \pi_e(max(X)) \tag{3}$$

$$\mu_m = \mu - \pi_m(min(X)) \tag{4}$$

where μ is just the mean of all samples and π_e and π_m are the mixture coefficient initializations.

2. Expectation Step: Responsibility values (Φ_{nk}) are then computed for all N samples in X and for each cluster. These signify how strongly the *nth* sample belongs to either k cluster.

$$\Phi_{nk} = \frac{\pi_k \mathcal{N}(X(n)|\mu_k, \Sigma)}{\pi_e \mathcal{N}(X(n)|\mu_e, \Sigma) + \pi_m \mathcal{N}(X(n)|\mu_m, \Sigma)} \tag{5}$$

$$k = \{e, m\} \tag{6}$$

3. Maximization Step: Re-compute values of means, covariance and mixture coefficients for both clusters through updated values of responsibility using equations (7-9). Then the log likelihood is computed for all N samples using equation (10). The likelihood is the probability that the sample belongs to the cluster is assigned. Therefore, if the likelihood converges, the algorithm has finished running. If not, then it goes back to the Expectation Step.

$$\mu_{\mathrm{k}}{}^{*} = \frac{\sum_{n=1}^{N} \Phi_{\mathrm{nk}} X(n)}{\sum_{n=1}^{N} \Phi_{\mathrm{nk}}} \tag{7}$$

$$\Sigma^{*}{}_{\mathrm{k}} = \frac{\sum_{n=1}^{N} \Phi_{\mathrm{nk}} (X(n) - \mu_{\mathrm{k}}{}^{*})(X(n) - \mu_{\mathrm{k}}{}^{*})^{\mathrm{T}}}{\sum_{n=1}^{N} \Phi_{\mathrm{nk}}} \tag{8}$$

$$\pi_{\mathrm{k}}{}^{*} = \frac{\sum_{n=1}^{N} \Phi_{\mathrm{nk}}}{N} \tag{9}$$

$$ln(p(X|\mu, \Sigma, \pi)) = \sum_{n=1}^{N} \{\pi_{\mathrm{e}} \mathcal{N}(X(n)|\mu_{\mathrm{e}}, \Sigma) + \pi_{\mathrm{m}} \mathcal{N}(X(n)|\mu_{\mathrm{m}}, \Sigma)\} (10)$$

5. Results

We trained the algorithm on 1 month of NetFlow records collected across 3 site routers. The results are shown in Figure 4-6.

Here it is important to note that the mixture coefficients are processed at runtime. We use $\pi_{\mathrm{e}} = 0.9$ and $\pi_{\mathrm{m}} = 0.1$. This definition is based on the assumption that approximately 90% of all network traffic is a small number of large-sized elephant flows and the rest is 10% which are many in number of small-sized mice flows.

This assumption is also used to validate the algorithm, to prove experimentally that elephant flows always lie in the top 10% of flows. The GMM algorithm trains on the size of data transfers and duration. The results are able to show two distinct clusters of flows such as an elephant flow being large in size and have long durations.

Table 2 statistically verifies whether or not the identified flows are actually in the top 10% of flows that lasted the longest in the network while having the largest rates.

Router Site	Lie in top 10% (size)?	Lie in top 10% (rate)?
Router1	Yes	Yes
Router2	Yes	Yes
Router3	Yes	Yes

Table 2: Experimental results on identified Elephant flows.

6. Conclusion

In this paper, we presented a novel machine learning approach for classifying elephant flows that is agnostic to the variance in flow data. These are preliminary results on studying the kind of flows observed at different sites. Further analysis is needed to study the cluster properties and development of the definition of what are elephants versus mice flows.

Based on GMM/EM for unsupervised classification, it prevents randomness and produces a classifier that can predict high-rate, large-size elephant flows. The algorithm was validated by statistically checking if identified flows possess the desired properties of large-rate and long durations.

Once clusters have been identified, properties of these clusters can be identified to perform online classification as a flow arrives (Figure 7). Traditionally, engineers would try to recognize flows based on their experience to find problems. The GMM model developed is able to automatically figure out elephant and mice flows specific to particular sites. This can then be related to the kind of applications running at the sites which can be responsible for throttling and avoid congestion over links.

7. References

[1] LBNL, Hep (high energy physics) network requirements workshop, in: ESnet Network Requirements Workshop, LBNL.

[2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, Openflow: enabling innovation in campus networks, in: ACM SIGCOMM Computer Communication Review, IEEE, p. 38(2).

[3] J. Paisley, J. Sventek, Real-time detection of grid bulk transfer traffic, in: 2006 IEEE/IFIP Network Operations and Management Symposium NOMS 2006, pp. 66–72.

[4] B. Suter, T. V. Lakshman, D. Stiliadis, A. K. Choudhury, Design considerations for supporting tcp with per-flow queueing, in: INFOCOM '98. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, volume 1, pp. 299–306 vol.1.

[5] A. Curtis, J. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, S. Banerjee, Devoflow: Scaling flow management for high-performance networks, in: Proceedings of the ACM SIGCOMM 2011 Conference, ACM, 2011, pp. 254–265.

[6] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, S. Shenker, Nox: Towards an operating system for networks, SIGCOMM Comput. Commun. Rev. 38 (2008) 105–110.

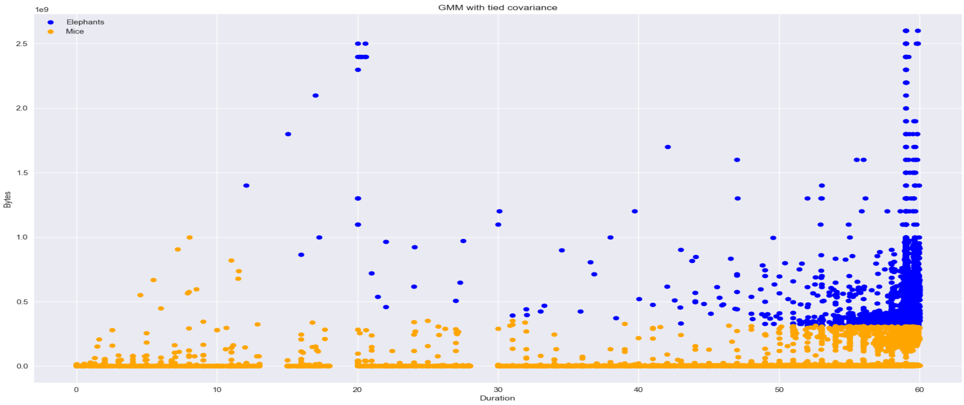

Figure 4: Elephants identified in blue and mice in orange for Router1.

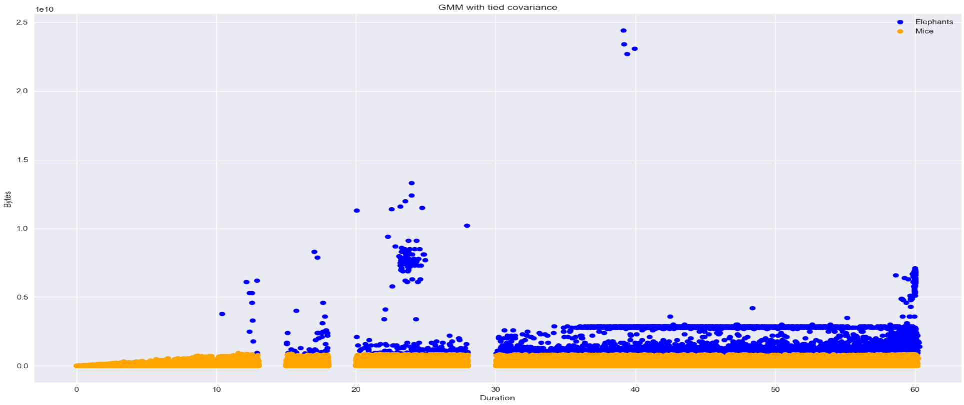

Figure 5: Elephants in blue and mice in orange for Router2.

[7] M. Mirza, J. Sommers, P. Barford, X. Zhu, A machine learning approach to tcp throughput prediction, in: IEEE/ACM Transactions on Networking.

[8] S. Shirali-Shahreza, Y. Ganjali, Traffic statistics collection with flexam, in: Proceedings of the 2014 ACM conference on SIGCOMM, ACM.

[9] T. V. L. Zizhong Cao, Murali Kodialam, Traffic steering in software defined networks: planning and online routing, in: DCC '14: Proceedings of the 2014 ACM SIGCOMM workshop on Distributed cloud computing, ACM.

[10] Z. Yan, C. Tracy, M. Veeraraghavan, T. Jin, Z. Liu, A network management system for handling scientific data flows, Journal of Network and Systems Management 24 (2016) 1–33.

[11] T. Mori, M. Uchida, R. Kawahara, J. Pan, S. Goto, Identifying elephant flows through periodically sampled packets, in: Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement, ACM, 2004, pp. 115–120.

[12] J. W. Jiang, T. Lan, S. Ha, M. Chen, M. Chiang, Joint vm placement and routing for data center traffic engineering, in: INFOCOM, 2012 Proceedings IEEE.

[13] R. Trestian, G. M. Muntean, K. Katrinis, Micetrap: Scalable traffic engineering of datacenter mice flows using openflow, in: 2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013), pp. 904–907.

[14] Y. Zhang, L. Breslau, V. Paxson, S. Shenker, On the characteristics and origins of internet flow rates, in: Proceedings of the 2002 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, ACM, 2002, pp. 309–322.

[15] Y. Lu, B. Prabhakar, F. Bonomi, Elephanttrap: A low cost device for identifying large flows, in: High-Performance Interconnects, IEEE, pp. 99–108.

[16] M. Kodialam, T. V. Lakshman, S. Mohanty, Runs based traffic estimator (rate): a simple, memory efficient scheme for per-flow rate estimation, in: IEEE INFOCOM 2004, volume 3, pp. 1808–1818 vol.3.

[17] F. Hao, M. Kodialam, T. V. Lakshman, H. Zhang, Fast, memory-efficient traffic estimation by coincidence counting, in: Proceed-

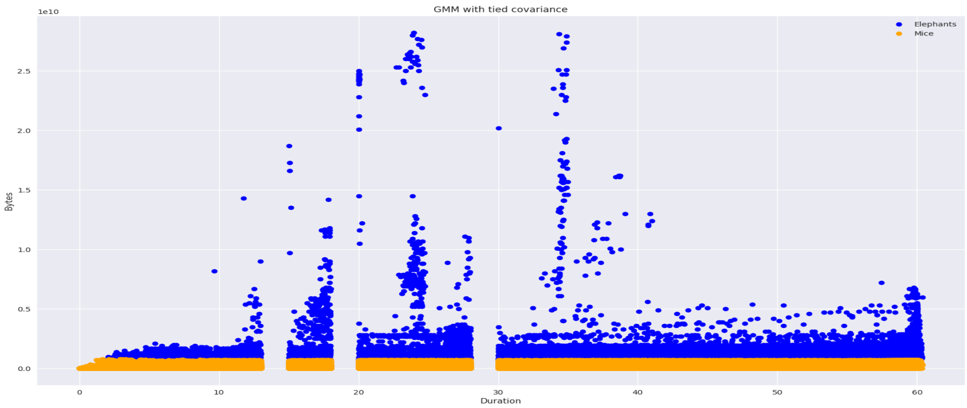

Figure 6: Elephants identified in blue and mice in orange for Router3.

Netflow records
Cluster decisions
Elephant versus Mice

Figure 7: Using the GMM classifier for online flow clustering.

ings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies, volume 3, pp. 2080–2090 vol. 3.

[18] M. Zadnik, M. Canini, A. W. Moore, D. J. Miller, W. Li, Tracking elephant flows in internet backbone traffic with an fpga-based cache, in: 2009 International Conference on Field Programmable Logic and Applications, pp. 640–644.

[19] N. Kamiyama, T. Mori, Simple and accurate identification of high-rate flows by packet sampling, in: Proceedings 25TH IEEE International Conference on Computer Communications, pp. 1–13.

[20] N. Duffield, C. Lund, M. Thorup, Flow sampling under hard resource constraints, in: Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems, SIGMETRICS '04/Performance '04, ACM, 2004, pp. 85–96.

[21] C. Barakat, G. Iannaccone, C. Diot, Ranking flows from sampled traffic, in: Proceedings of the 2005 ACM Conference on Emerging Network Experiment and Technology, CoNEXT '05, ACM, New York, NY, USA, 2005, pp. 188–199.

[22] T. Jin, C. Tracy, M. Veeraraghavan, Z. Yan, Traffic engineering of high-rate large-sized flows, in: 2013 IEEE 14th International Conference on High Performance Switching and Routing (HPSR), pp. 128–135.

[23] Z. Liu, M. Veeraraghavan, Z. Yan, C. Tracy, J. Tie, I. Foster, J. Dennis, J. Hick, Y. Li, W. Yang, On using virtual circuits for gridftp transfers, in: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, SC '12, IEEE Computer Society Press, Los Alamitos, CA, USA, 2012, pp. 81:1–81:11.

[24] T. Jin, C. Tracy, M. Veeraraghavan, Characterization of high-rate large-sized flows, in: 2014 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), pp. 73–76.

[25] S. Zander, T. Nguyen, G. Armitage, Automated traffic classification and application identification using machine learning, in: The IEEE Conference on Local Computer Networks 30th Anniversary (LCN'05)l, pp. 250–257.

[26] T. T. T. Nguyen, G. Armitage, A survey of techniques for internet traffic classification using machine learning, IEEE Communications Surveys Tutorials 10 (2008) 56–76.

[27] H. Ibrahim, O. Zuobi, M. Al-Namari, G. MohamedAli, A. Abdalla, Internet traffic classification using machine learning approach: Datasets validation issues, in: 2016 Conference of Basic Sciences and Engineering Studies (SGCAC), pp. 158–166.

[28] W. Li, A. W. Moore, A machine learning approach for efficient traffic classification, in: 2007 15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, pp. 310–317.

[29] N. Williams, S. Zander, G. Armitage, A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification, SIGCOMM Comput. Commun. Rev. 36 (2006) 5–16.

[30] A. W. Moore, D. Zuev, Internet traffic classification using bayesian analysis techniques, SIGMETRICS Perform. Eval. Rev. 33 (2005) 50–60.

[31] R. Yuan, Z. Li, X. Guan, L. Xu, An svm-based machine learning method for accurate internet traffic classification, in: Information Systems Frontiers, p. 149156.