A Clustering-based Analysis of DPI-labeled Video Flow Characteristics in Cellular Networks

Johan Garcia
Department of Mathematics and Computer Science
Karlstad University, Sweden
Email: {johan.garcia@kau.se

***Abstract*—Using a specially instrumented deep packet inspection (DPI) appliance placed inside the core network of a commercial cellular operator we collect data from almost four million flows produced by a 'heavy-hitter' subset of the customer base. The data contains per packet information for the first 100 packets in each flow, along with the classification done by the DPI engine. The data is used with unsupervised learning to obtain clusters of typical video flow behaviors, with the intent to quantify the number of such clusters and examine their characteristics. Among the flows identified as belonging to video applications by the DPI engine, a subset are actually video application signaling flows or other flows not carrying actual transfers of video data. Given that DPI-labeled data can be used to train supervised machine learning models to identify flows carrying video transfers in encrypted traffic, the potential presence and structure of such 'noise' flows in the ground truth is important to examine. In this study K-means and DBSCAN is used to cluster the flows marked by the DPI engine as being from a video application. The clustering techniques identify a set of 4 to 6 clusters with archetypal flow behaviors, and a subset of these clusters are found to represent flows that are not actually transferring video data.**

I. Introduction

An increasing fraction of all traffic flowing through cellular networks is transporting video. As video traffic becomes a major traffic type, and as the performance of such traffic has a large importance on user satisfaction, network operators may choose to do video traffic management. This is done to strike an appropriate trade off between optimizing the perceived Quality of Experience for the end user, and the cost of consumed network resources. Such traffic management is only possible if flows containing video data transfers can be separated from other traffic flows. A common approach is to use Deep Packet Inspection (DPI) to examine various aspects of the traffic to infer the application type to which the flow belongs to.

DPI approaches are severely hampered for encrypted traffic, making alternate approaches necessary. Regardless of encryption the traffic characteristics of flows in terms of packet sizes, packet direction and packet timing aspects are available. Such features can form the base of a Machine Learning (ML) approach using supervised learning with the goal of classifying flows into video or non-video flows. However, this requires training of the ML model that will do such classification. Training can be done using currently available DPI-classified flows as ground truth. For flows classified by the DPI as being from a video application, there is a noteworthy distinction between a flow being classified as coming from a video application, and the flow actually transferring video data. This distinction is at the focus of this investigation. Here we explore the separation of all flows classified by the DPI as video-related into those flows who carry actual video data, and those who do not. We use unsupervised learning to identify archetypal video flow types, some of which are clearly not representing flows transferring video data.

A considerable amount of related work exists, and Finsterbush et al. [4] provide a survey of payload-based classification approaches. An early survey of machine-learning based classification is provided by Nguyen et al [8], while Erman et al. [2] examines traffic classification using K-means and DBSCAN which are the two approaches also used here. Regarding classifying encrypted traffic, Shbair et al. [10] proposed a multi-level approach to identify application classes in HTTPS traffic. Fu et al. [5] examines encrypted cellular traffic, with a particular interest in classifying mobile messaging apps. By analyzing a recent cellular data set we in this study extend the knowledge regarding video flow behavior as observable inside the core of the cellular network, with a focus on features also available for encrypted traffic.

II. Experimental overview

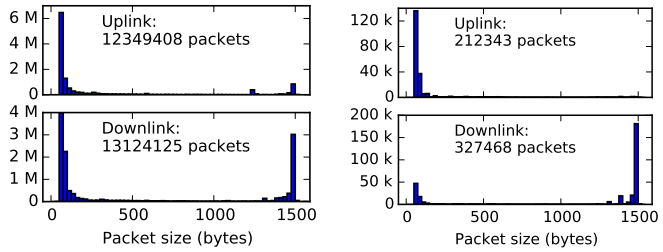
A. Measurement collection

The data set was collected from inside the cellular network backbone of a commercial cellular operator. An operational DPI box was modified to collect anonymized information for each of the first 100 packets in a flow. For each packet, the data also includes the flow application label as inferred by the DPI engine at that point in time. The DPI engine has over 1000 applications that it differentiates between, and it can update the classification of a flow as more packets are observed. A list of applications considered to be video-related was provided by the DPI vendor.

The collected data comes from a subset of the most active cellular subscribers, the so-called 'heavy-hitters'. Data collection was performed during May 2016. The data set consists of 3909197 unique flows, which in the first 100 packets had a total of 33867394 packets and 11.8 gigabytes.

Data sanitization was employed to remove flows which were started during the last 30 seconds of the capture window, or were initiated before the start of the capture window. Flows

978-3-901882-89-0 @2017 IFIP



(a) For all flows (b) For video flows

Figure 1: Packet size histograms

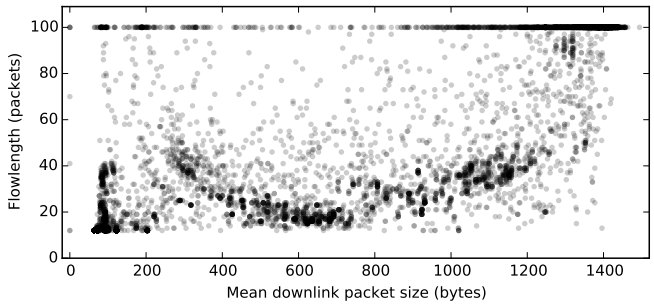


Figure 2: Scatter plot between mean downlink packet size and flow length for 12019 video flows

	Application	Nr of flows	Mean DL Packetsize	Observed packets	Transport protocol
1	SSLv3	188429	650	5613758	TCP
2	DNS	173649	203	3107711	UDP/TCP
3	HTTP	142748	828	5003460	TCP
4	Google	39016	591	1514997	TCP/UDP
5	Facebook	15604	744	754243	TCP

Table I: Most frequent application classes overall

	Application	Nr of flows	Mean DL packetsize	Observed Packets	Transport protocol
1	YouTube	5293	1128	313084	UDP/TCP
2	Newcamd	4001	78	49572	TCP
3	HTTP Media stream	1898	1293	109948	TCP
4	Netflix	733	1316	58575	TCP
5	Flash video over HTTP	57	1417	5567	TCP

Table II: Most frequent video applications

which had a length of 10 or fewer packets were not considered for the further analysis as these short flows are dominated by DNS traffic, BitTorrent KRPC exchanges, and ICMP traffic.

B. Data set characterization

One aspect of the data that can be characterized is the packet size distribution. Figure 1a shows the distribution for the complete data set, and also indicates the number of packets in each direction. It can be observed that the majority of the packets in both the uplink and downlink directions are actually less than a few hundred bytes. Figure 1b shows the packet size distribution for the flows which the DPI engine has classified as video-related. In the context of this examination, the downlink direction is the major interest and as can be seen the distribution for video flows is markedly different to what is seen in Figure 1a. For video flows a much larger fraction of the packets have a large packet size. This would be consistent with the intuition that video flows need to transfer large amounts of data causing the majority of packets to be full-size packets.

The data can also be characterized according to the distinct applications which generated the transferred packets. Tables I and II show the top five applications, according to the number of flows observed.

The second video-related application stands out with regards to its average downlink packet size, which is only 78 bytes. The Newcamd application is not transferring video, but rather is an application for card sharing to circumvent encrypted broadcasts.

It is also possible to characterize the data graphically, and Figure 2 shows a scatter plot of two features potentially useful for clustering. A large fraction of the flows are located towards the upper right corner, which would be appropriate for flows transferring actual video data. The collection limit of 100 packets creates the distinct break at the top of the figure.

III. Video flow clustering

A large number of clustering algorithms exists, and K-means is one commonly used approach. K-means requires the number of clusters to be specified when the algorithm is run. In this evaluation, the correct number of clusters to use in the clustering algorithm is not known, since that is one of the outputs of the analysis.

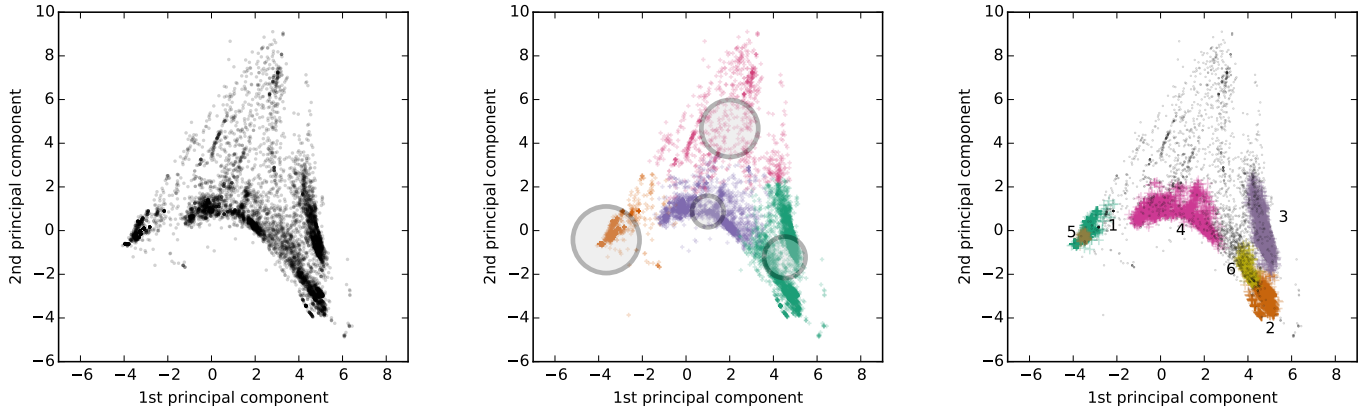
Besides K-means, the DBSCAN density based method is also used here. This method does not require the number of cluster to provided, but instead has two tunable parameters which in turn impacts the number of clusters the algorithm detects.

A. Employed features

For the clustering task, a straightforward set of 21 statistics-based features were computed for each flow. Seven different metrics were used, and these metrics were computed separately for the complete set of all observed packets in the flow, packets in the downlink direction, and packets in the uplink direction. The utilized metrics were number of packets, mean packet size, maximum packet size, standard deviation of the packet sizes, variance of the packet sizes, skew of the packet sizes, and kurtosis of the packet sizes. The features were scaled to zero mean and unit variance.

B. Initial clustering examination

Unsupervised techniques for reducing the number of dimensions are useful for providing an intuition of the clustering behavior in ways which cannot be seen in scatter plots such as Figure 2, which only captures variation in two features. Principal component analysis (PCA) is a technique that performs linear transformations to maximize the variance for the resulting primary components. A PCA transformation



(a) Plot of all video flows along the two first principal components.

(b) K-means clustering with cluster centers, scaled for clarity.

(c) DBSCAN clustering. Black dots signify samples not belonging to any cluster.

Figure 3: Scatter plots of all video flows along the first two principal components

was performed on the 21-dimensional data set created by the employed features, and the flows are shown along the two primary components in Figure 3a. The figure thus shows the projection of the 21-dimensional feature space onto a two-dimensional plane giving maximum variation, which in this case explains 77 percent of the total variance. It can be seen that the PCA creates more spread for the cramped corner part of Figure 2, and there are visual indications of clusters. However, Figure 3a contains no information regarding the spread in the remaining 19 PCA dimensions.

To find the appropriate number of clusters, an internal clustering validation metric is needed. The properties of several such metrics are discussed by Liu et al. [7], and in this study the silhouette coefficient [9] is used. For each sample, the silhouette coefficient is composed from two scores:

a: The mean distance between a sample and all other points in the same class.

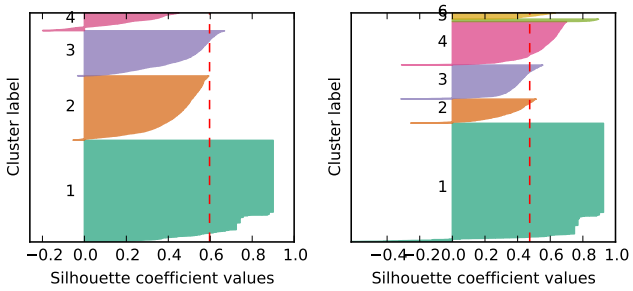
b: The mean distance between a sample and all other points in the next nearest cluster.

The silhouette coefficient for a single sample is then given as: $s = \frac{b-a}{max(a,b)}$. The mean silhouette score for all data points can then be used as an overall metric of clustering performance.

C. K-means clustering

Here we use the K-means++ variant [1] for clustering in the PCA space using Euclidean distance. The specified number of clusters to be formed is varied, and the average silhouette score for each resulting clustering is computed. Utilizing this approach the best silhouette score (0.597) was obtained for four clusters. The distribution of silhouette scores for each flow sample is shown in Figure 4a. As is visible in the figure, cluster one contains most of the samples, and has a high average silhouette score.

The resulting clusters are shown in Figure 3b. As can be seen, K-means clustering employs hard clustering, signifying



(a) K-means clustering

(b) DBSCAN clustering

Figure 4: Silhouette coefficient values

that all samples must belong to a cluster. Also shown are the cluster centers, and a relative indication of the cluster size.

D. DBSCAN clustering

DBSCAN clustering [3] does not require the input of the number of clusters to be formed, but the algorithm uses input parameters that affects the numbers of clusters formed. DBSCAN works by separating areas of high density from areas of low density. High density areas are considered clusters. Clusters are formed by core samples, which needs to have a min_samples number of samples within a distance of eps. In this evaluation we examined a range of settings using grid search in the ranges [10,460, step=50] for min_samples and [0.05,0.95, step=0.05] for eps, and used Euclidean distance as distance metric. A consequence of the DBSCAN approach is that the clusters do not need to be convex shaped as is the case for K-means, and that not all samples need to be assigned to a cluster.

The DBSCAN clustering had the highest silhouette coefficient (0.474) for eps=0.900 and min_samples=60. These settings resulted in six clusters, with a silhouette score distribution as shown in Figure 4b. In related work [2] DBSCAN

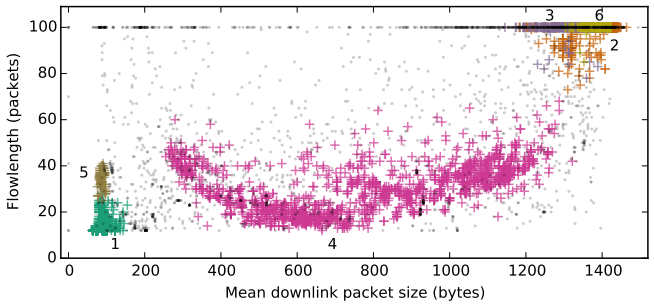


Figure 5: DBSCAN clustering results transformed back to original feature space and shown over two of the 21 features.

Cluster label:	1	2	3	4	5	6	N
Application (DPI-label)	%	%	%	%	%	%	%
YouTube	5.4	0.1	26	30	1.9	0.5	36
Newcamd	99	0.0	0.0	0.9	0.0	0.0	0.4
HTTP Media stream	41	31	2.6	2.8	0.0	6.6	16
Netflix	2.7	51	0.3	20	0.0	13	14
Flash video over HTTP	0.0	93	0.0	1.8	0.0	0.0	5.3
Number of flows in cluster	5029	1017	1437	1832	100	244	2360

Table III: Per application flow distribution over clusters

also achieved a lower clustering metric score, but its ability to have unclustered samples was found to create more accurate cluster assignments as compared to K-means.

The resulting clusters in PCA dimensions are shown in Figure 3c. It can be seen that a number of samples are not belonging to any cluster. For some of the these samples the separation from a cluster is located in other dimensions than the two shown in the figure. Such point are seen in the figure as unclustered samples visually overlapping clustered regions.

By inverting the PCA transform and feature scaling it is possible to map the obtained DBSCAN clusters back to the original feature space. An illustration of the resulting clusters for the same two features as in Figure 2 are shown in Figure 5. Here it can be observed that three of the clusters (2,3,6) appears to capture flows with characteristics consistent with transporting actual video data. The three remaining clusters (1,4,5) captures flows which are generated by video applications, but are unlikely to be transferring actual video data.

The video application is available from the DPI-labeling, and Table III presents an overview of what fraction of flows for each video application that maps to which cluster (1-6) or do not belong to any cluster (N). From the table it can be observed that for several video applications the majority of flows are actually not transporting video information. It is also clear that the Newcamd and Flash video over HTTP applications has a fairly consistent behavior, with a large majority of their flows belonging to a single cluster. The other three DPI-labeled video applications are mixing actual video transfer flows with other flow types. Both inherent application characteristics and DPI rule set construction aspects [6] likely influence this observed behavior.

IV. Conclusions

Using packet characteristics captured from live traffic in a cellular network, we perform a clustering analysis using K-means and DBSCAN to explore the number of archetypal video flow characteristics that are present. The examined clustering mechanisms result in 4 to 6 identified clusters. Several of the clusters represent flows that are not actual video data transfers. While K-means achieved a better silhouette score, DBSCAN allows for flow instances to be considered as background, and not belonging to any cluster. The initial visualization performed over basic features of the data, as well as a principal component analysis indicated that there was a noticeable presence of such background flows.

While more detailed studies are necessary to conclusively characterize the typical flow behavior in each identified cluster, it is nevertheless possible to conclude that several of the identified clusters captures flows that are not actually transferring video data. When DPI labeled data such as captured in this experiment is to be used to train machine learning models that aim to detect flows transferring video data, using unsupervised methods such as described here to identify and remove application signaling and other flows not actually carrying video data can be of considerable benefit.

Acknowledgments

The authors wish to thank Procera Networks for support and for providing the data set. This study was partly funded by the HITS project grant from the Swedish Knowledge Foundation.

References

[1] David Arthur and Sergei Vassilvitskii. K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 1027–1035, 2007.

[2] J. Erman, M. Arlitt, and A. Mahanti. Traffic classification using clustering algorithms. In *Proceedings of the 2006 SIGCOMM workshop on Mining Network Data*, pages 281–286, 2006.

[3] M. Ester, H.-P. Kriegel, J. Sander, and X Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining*, pages 226–231. IEEE.

[4] M. Finsterbusch, C. Richter, E. Rocha, J.-A. Muller, and K. Hanssgen. A survey of payload-based traffic classification approaches. *IEEE Communications Surveys & Tutorials*, 16(2):1135–1156, 2014.

[5] Y. Fu, H. Xiong, X. Lu, J. Yang, and C. Chen. Service usage classification with encrypted internet traffic in mobile messaging apps. *IEEE Transactions on Mobile Computing*, PP(99):1–1, 2016.

[6] Fangfan Li, Arash Molavi Kakhki, David Choffnes, Phillipa Gill, and Alan Mislove. Classifiers unclassified: An efficient approach to revealing ip traffic classification rules. In *Proceedings of the Internet Measurement Conference (IMC)*, pages 239–245, 2016.

[7] Yanchi Liu, Zhongmou Li, Hui Xiong, Xuedong Gao, and Junjie Wu. Understanding of internal clustering validation measures. In *IEEE 10th International Conference on Data Mining (ICDM)*, pages 911–916, 2010.

[8] Thuy TT Nguyen and Grenville Armitage. A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys & Tutorials*, 10(4):56–76, 2008.

[9] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53 – 65, 1987.

[10] Wazen M. Shbair, Thibault Cholez, Jérôme François, and Isabelle Chrisment. A Multi-Level Framework to Identify HTTPS Services. In *IEEE/IFIP Network Operations and Management Symposium (NOMS 2016)*, pages 240–248, 2016.