# I Know What You Saw Last Minute - The Chrome Browser Case

Ran Dubin
Communication Systems Engineering
Ben-Gurion University of the Negev
Israel

Amit Dvir
Center for Cyber Technologies
Department of Computer Science
Ariel University
Israel

Ofir Pele
Center for Cyber Technologies
Department of Computer Science
Department of Electrical and Electronics Engineering
Ariel University
Israel

Ofer Hadar
Communication Systems Engineering
Ben-Gurion University of the Negev
Israel

***Abstract*—Previous research has shown that information can be extracted from encrypted multimedia streams. This includes video titles classification of non-HTTP adaptive streams (non-HAS). This paper presents an algorithm for *encrypted HTTP adaptive video streaming title classification*. We evaluated our algorithm on a new YouTube popular videos dataset that was collected from the Internet under real-world network conditions. Our algorithm's classification accuracy is 98%. We provide the dataset and the crawler for future research.**

***Index Terms*—HTTP Adaptive Video Streaming, HTTP2, Encrypted Traffic Classification, YouTube, Network Traffic Security**

## I. Introduction

Every day, hundreds of millions of Internet users view videos online, in particular on mobile phones whose numbers are clearly going to increase [1]. By 2019, the share of video traffic is expected to increase to 80% of the total IP traffic, up from 67% in 2014. By 2019 the IP network will have to accommodate a video traffic of around 135 exabytes per month [1]. Currently, most of the video streaming web sites are using HTTP Adaptive Streaming (HAS).

Dynamic Adaptive Streaming over HTTP (DASH) [2] is the *de facto* standard method for HAS. DASH is a Multi Bit Rate (MBR) streaming method that was designed to improve viewers' Quality of Experience (QoE) [3]. In DASH, each video is divided into short segments, typically a few seconds long ($2-16$ seconds), and each segment is encoded several times, each time with a different quality representation. The user (player) Adaptation Logic (AL) algorithm is responsible for the automatic selection of the most suitable quality representation for each segment, based on parameters such as client's playout buffer and network conditions. As a result, the quality representation in DASH can change between segments.

In DASH, each quality representation is encoded in variable bit rates (VBRs). VBR does not attempt to control the output bit rate of the encoder, so that the distortion will not vary significantly [4]. Since the adaptation logic selects the most suitable quality representation, where each has variable bit rate, information retrieval from video streams is challenging. YouTube, which often uses DASH over HTTP2 and HTTP byte-range mode [5], now occupies a market share of over 17% of the total mobile network bandwidth in North America [1], [6]. In this mode (HTTP byte-range mode), the byte range of each segment request can be different. Further information about byte-range will be discussed in Section II. Moreover, YouTube has started to encrypt their video services [7]. As a result, traditional Deep Packet Inspection (DPI) methods for information retrieval in general and video title classification in particular are not viable.

Recent works have shown that encryption is not sufficient to protect confidentiality [8]–[18]. Wright et al. [16] suggested a method that exploits the VBR codecs characteristics of encrypted Voice Over Internet Protocol (VOIP) streams for language identification.

Other works showed that video titles classification of encrypted video streams is possible [12]–[14]. These works use traffic patterns features such as packet size and the application layer information. Saponas et al. [12] uncovered security issues with consumer electronic gadgets that enables information retrieval such as video titles classification. Liu et al. [13] presented a method for video title classification of RTP/UDP Internet traffic. In [14] Liu et al. presented an improved algorithm that is more efficient and demonstrated excellent results on a bigger data set with real network conditions. They use the wavelet transform for constructing unique and robust video signatures with different compactness.

Since these works [12]–[14] have been conducted, there have been several changes in video traffic over the Internet:

- MBR adaptive streaming (the selected quality represen-

tation of the examined video title can change).
- HTTP byte-range selection over HTTP.
- HTTP version 2 [5].

In this paper, we present a new video title classification algorithm of YouTube video streams over DASH. The paper goal is to emphasize that the encryption is not enough for protecting YouTube DASH users. Attackers can use this method to understand the user viewing preferences for advertisement or use it as an Open Source Intelligence (OSINT) [19] vector for various purposes, while ISPs can use this method for load balancing and CDN optimization. Our algorithm classifies the video title once the viewer has finished seeing the movie. Inspired by previous works, we exploit the Variable Bit-Rate (VBR) encoding and represent each video stream as a set of peaks' bit rates. As for learning algorithms, we used SVM with the RBF kernel and a Nearest Neighbor (NN) with the set intersection as a similarity measure. We evaluated our algorithm on a new YouTube popular videos dataset that was collected from the Internet under real-world network conditions. We provide the dataset [20] and the crawler [21] for future research. Our algorithm's classification accuracy is 98%.

## II. YouTube Analysis

To have a better understanding about the traffic properties in the case of encrypted video streaming over DASH using HTTP2 (secure), we examined YouTube Chrome browser traffic in *fixed* and automatic (*auto*) download modes. In *fixed* mode, the player downloads a fixed video quality without quality representation adaptation. As a result, the player downloads the video in a short time period. In the *auto* mode the player selects the quality representation based on the network conditions, browser resolution, and buffer occupancy. As a result, the player examines the client conditions for each segment download in order to maximize the downloaded quality. Figs. 1a -1b depict Chrome browser traffic download patterns of a single video title stream. Note that the original download contained TCP re-transmissions and we filtered them using a TCP stack implementation [22].

For the analysis, we used the Fiddler [23] web debugging proxy for viewing different requests without the encryption. Fig. 2 illustrates the YouTube *auto* download mode with Chrome. Each video download has several flows. We found that often there are two flows both with audio and video. The short traffic segments contain audio while the longer contain video.

Fig. 3 shows three downloads over three different WiFi networks of the same video title, all with the same quality representation. The figure shows that the first two segments' download byte-range requests were the same. Due to different network load conditions, the next segment's download byte-range requests were different.

The above analysis leads to several insights concerning the factors that can hinder classification efforts:

1) YouTube uses HTTP Byte range requests. That is, the byte range of each segment request can be different. This depends on the client's network conditions and playout buffer levels.
2) Audio data and video data can be found in the same 5-tuple flow ({protocol (TCP/UDP), src IP, dst IP, src port, dst port}). In some cases we cannot distinguish between them. For example, using Fiddler we can see in Fig. 2 in the top flow, that the first two audio sections (the short sections) are very close to the two video sections (the long sections). This can cause a classification error.
3) Close video segments responses can be found in the same flow. For example, in Fig. 2 in the top flow, there are four segments (in the first video section) that are hard to separate.
4) The video and audio sections in each flow do not have a periodic order.
5) YouTube uses video advertisements that are often downloaded in high quality. As a result, distinguishing between them is not trivial.
6) The players download the same quality representation in parallel over different flows to accelerate the download.

## III. Proposed Algorithm

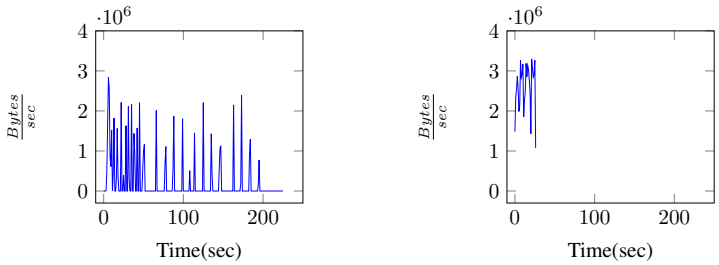
We suggest two algorithms for video title classification of YouTube encrypted video streams: SVM with an RBF kernel and a nearest neighbor algorithm. The proposed solution architecture is illustrated in Fig. 4.

The first two modules only pass YouTube video streams to the next modules. Each segment of network traffic enters the system separately and is first passed into the *Connection Matching* filter. This filter is responsible for checking whether the incoming flow is new or ongoing. It does so based on a five-tuple representation: {protocol (TCP/UDP), src IP, dst IP, src port, dst port}. If the incoming flow is new, the *DPI* filter decides whether it is a YouTube flow. This is done based on the SNI field in the *Client Hello* message that is part of the Secure Socket Layer (SSL) protocol. If the *DPI* module finds the following string: *googlevideos.com* (which identifies YouTube) in the SNI, the stream is passed to the *Feature Extraction* module (Section III-A). Any ongoing or new traffic flow that is not recognized by the *DPI* as video streaming is transparently passed into the network without further analysis. The *Classification Algorithm* modules are described in sections III-B and III-C. The *Pre-Processing* model removes audio and video features (see Section IV-C).

In this paper we assume that we know how to detect Chrome browser traffic based on the fact that each browser and OS has a different fingerprint [18]. We decided to work with Chrome due to its support of HTTP2 and its increasing popularity.

### A. Feature Extraction

DASH is streamed over a TCP transport protocol. Streaming applications have high bit rate consumption. Thus, feature extraction methods need to take TCP limitations such as re-transmission caused by network problems into account. Re-transmission adds additional data to the stream that can cause classification errors.




(a) Chrome auto mode over HTTP2. (b) Chrome fixed mode over HTTP2.

Fig. 1: YouTube Costa Rica in 4K - traffic traces from Chrome (Ver 43.0.2357.81) with $HTML5$ player in automatic and fixed quality selection modes.

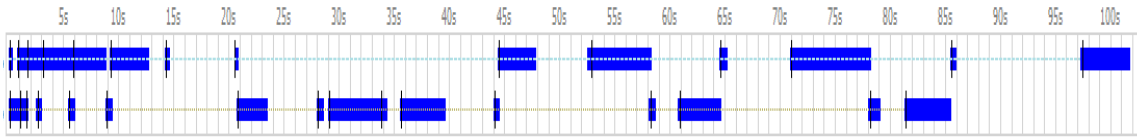



Fig. 2: YouTube Costa Rica 4k auto mode with Chrome. Each horizontal line represents different YouTube flows from the same download. The video quality is $720P$.

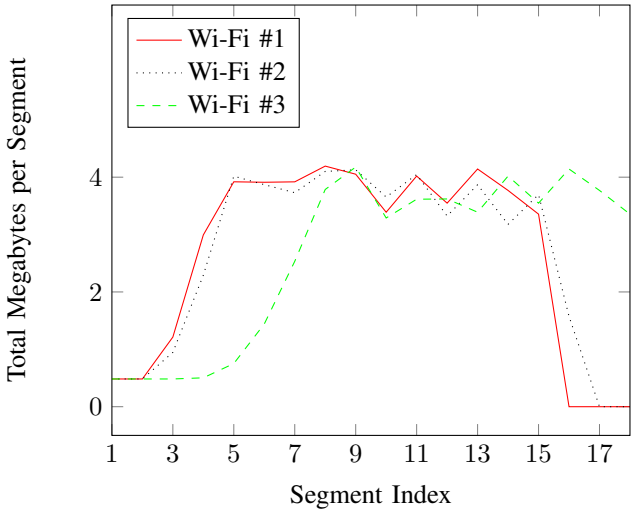



Fig. 3: Total megabytes per segment of three downloads over different Wi-Fi networks of the same video title, all with the same quality representation. Due to network conditions variability, there are differences between the networks.

Many recent works have suggested methods for encrypted traffic classification and several surveys have presented detailed description of the state of the art methods [24]–[28]. Several works have examined different statistical features such as session duration [29]–[31], number of packets in a session [30], [32], [33], different variance calculations of the minimum, maximum, and average values of inter-arrival packet time [30], [32], payload size information [32], [34], bit rate [34], [35], Round-Trip Time (RTT) [35], packet direction [36], or server sent bit rate [37]. Not all these features are important for video streams classification. For instance, the packet size is often MTU size in video streaming, as video streaming consumes high bandwidth and re-transmission occurs often. Moreover, TCP parameters such as server sent bit rate, inter-arrival packet time, RTT, and packet direction are weak features.

In Figure 1a we can see that flows contain peaks. That is, traffic bursts. The same characteristics were observed by Rao et al. [38] and Ameigeiras et al. [39] (coined in [38], [39] as "On/Off"). We decided to encode every peak of the stream to a feature. This feature is the Bit-Per-Peak (BPP); that is, the total number of bits in a peak after TCP ACK mechanism [17]. Working with the TCP ACK mechanism gives us the ability to filter out traffic re-transmission caused by delay and packet loss. Our feature extraction starts after we identify that this traffic flow is a Chrome YouTube video flow [18]. Any packet that enters the algorithm is verified by TCP stack implementation to prevent re-transmission packets from affecting our feature accuracy. After the TCP stack we accumulate the traffic throughput until a traffic silence is recognized.

### *B. Traffic Classification Algorithm - SVM with an RBF Kernel*

Each download was analyzed based on the steps in Fig. 4, where each download is represented as a BPP feature vector of a length of 40 features (this is the maximum needed feature length after the feature extraction). Downloads that don't have enough features are padded with zeros. Each feature vector is standardized with Gaussian with zero mean and unit variance.

We used a Support Vector Machine (SVM) with an RBF kernel [40], where the regularization parameter $C$ and $\gamma$ were found with 5-fold cross-validation over the following sets accordingly: $\{2^{-5}, 2^{-3}, ..., 2^{15}\}$ and $\{2^{-15}, 2^{-13}, ..., 2^{3}\}$.

Fig. 4: Proposed solution architecture.

### C. Traffic Classification Algorithm - Nearest Neighbor (NN)

In the Nearest Neighbor (NN) algorithm, each video stream (number $j$ of video title $i$) is represented by $\mathcal{S}_{ij}$, a set of BPP features (no duplicates). Note that each BPP-set may have different cardinality. The NN similarity score between two BPP-sets, $\mathcal{S}$ and $\mathcal{S}'$, is the cardinality of the intersection set:

$$\text{sim}(\mathcal{S}, \mathcal{S}') = |\mathcal{S} \cap \mathcal{S}'| \tag{1}$$

At test time, each video stream BPP-set, $\mathcal{S}_{\text{test}}$, is classified as the video title $i$ that has the maximum similarity score to one of the title training stream BPP-sets and the intersection size was larger then $Thr$, else we classify it as unknown:

$$\forall \;\; 1 \leq i \leq n, \;\; s_i = \max_{j=1}^{m_i} \text{sim}(\mathcal{S}_{\text{test}}, \mathcal{S}_{ij}) \tag{2}$$

$$y(\mathcal{S}_{\text{test}}) = \begin{cases} \underset{i=1}{\overset{n}{\text{argmax}}}\, s_i & \text{if } \left(\max_{i=1}^{n} s_i\right) > \text{Thr} \\ \text{unknown} & \text{otherwise} \end{cases} \tag{3}$$

## IV. Performance Evaluation

This section is structured as follows: First, we describe our dataset extraction and structure (Section IV-A). Second, we present the effect of the training set size on the classification accuracy (Section IV-B). Third, we describe the audio pre-processing effect in (Section IV-C). Fourth, we turn to evaluate the effect of packet loss and delay on the model's robustness in (Section IV-D). Finally, we evaluate the model's accuracy for video titles outside of the training dataset in Section IV-E.

### A. Dataset

To collect our dataset we used the Selenium web automation tool [41] with ChromeDriver [42] for the crawler [21], so it will simulate a user video download. The crawler receives the video title URL and the video duration in seconds. We start to record before the crawler start and open the browser and we finish after the timer time is up.

We collected a training set of encrypted video streams; the name of each video is transformed to title index: $y \in \{1 \dots V\}$ where $V = 30$, where each title was downloaded $n = 90$ times (each download may have different network conditions). Every segment of the stream is encoded to a feature. That is, the training dataset contained 2700 video streams of 30 unique video titles (90 different downloads per title), each downloaded in the automated quality representation selection mode. For each video in our dataset, YouTube provides a different subset of the following quality representations: $[144P, 240P, 360P, 480P, 720P, 1080P]$.

The video titles used in this study are popular YouTube videos from different categories such as news, video action trailers, and GoPro videos [20]. In this study we decided to focus on the Chrome browser since it is the most popular browser in the market. Chrome YouTube player has two different modes of operation: the fixed quality download mode and the adaptive quality selection mode. The Chrome YouTube player decides which quality representation to download based on its estimation of the client network conditions. As presented, both methods have different traffic signature patterns. Since the automated mode is the default mode and the diversity of its data is much more complex we decided to explore it in this work.

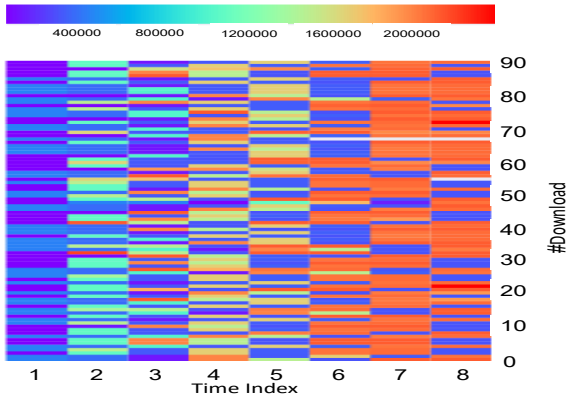
We have four testing datasets:

1) *test-adaptive-test-titles*: 300 video streams of 30 unique video titles (same titles as in the training phase). Each has 10 different downloads with an automated representation selection.
2) *test-adaptive-with-drop-test-titles*: 400 video streams of 10 unique video titles (titles taken from the training phase titles), each was downloaded with an additional added drop percentage from the following list: $\{1\%, 3\%, 6\%, 9\%\}$; each was downloaded in the automated mode.
3) *test-adaptive-with-delay-test-titles*: 400 video streams of 10 unique video titles (titles taken from the training phase titles); each was downloaded with an additional added drop percentage from the following list: $\{100, 300, 600, 900\}[ms]$; each was downloaded in the automated mode.
4) *test-adaptive-unknown-titles*: 200 video stream titles with one download per title were downloaded in the automated mode. The titles are new and were not included in the training steps. Please note, that in this dataset we didn't use ad-block since advertisement is considered as unknown titles.

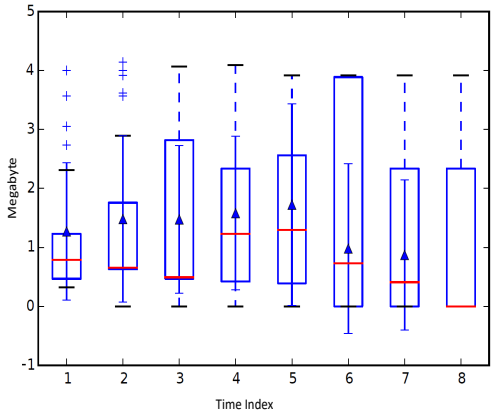
All the test video streams were different from the ones that were used in the training phase (because of network conditions). Furthermore, it is important to note that the added delay and drop affect the client player and cause it to select different (lower) representation than the normal use.

Fig. 5a illustrates an example of a single video title with 90 different downloads; each have 8 segments. We can see that each segment has a high bit rate variability. It is noteworthy that as the TCP window size increases, the variability at the beginning is a little smaller but still noticeable.

From Fig. 5b which illustrates the features boxplot, we can observe that the SVM classifier is less suitable for this task. This is due to the fact that the BPP features indexed by time have high variability due to the multiple video quality




(a) BPPs features of different streams (downloads) over time.




(b) BPPs features boxplot.

Fig. 5: 90 downloads of the Go Pro Snow Daze video title.

representations and HTTP byte-range, which change over BPP. Since the SVM in our experiments uses a vector of BPPs features that are indexed by time, it is more sensitive to those changes.

### *B. Accuracy Evaluation Using Different Training Dataset Sizes*

Fig. 6 depicts our accuracy with different numbers of training video titles. We used 10 test video streams for each title and present the average accuracy for each dataset size. The figure shows that the SVM classifier performance has a major increase from 5 to 60 titles. However, from 60 to 90 the accuracy gain increase is only 4%. It is interesting to see that even in a small scale dataset our proposed solution achieved high performance. When the dataset size is based on a single download the average accuracy is 79.66%. Each increase in the dataset size increased the accuracy until 98%. We will use our full dataset for the rest of the performance evaluation.

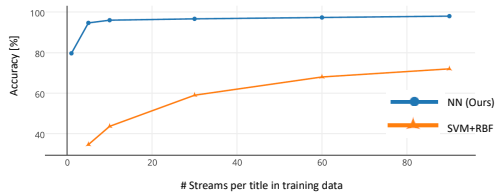



Fig. 6: Classification accuracy *vs.* different training dataset sizes.

### *C. Accuracy Evaluation Using Different Pre-Processing Methods*

In this test we evaluated the effect of audio removal pre-processing. We used the full training set (90 downloads per video title) with *test-adaptive-test-titles*. Using Fiddler, we found that low bit-rate audio traffic features has much lower entropy compared to video traffic with higher bit-rate ranges. Audio traffic responses range is from 65 KBytes to 400 KBytes. The pre-processing method filters out all low bit-rate peaks smaller than 400 KBytes. However, as presented earlier, it is not always possible to successfully distinguish in the encrypted traffic level between each audio/video response. We have found that the few first features of the video server responses are not necessary unique and are more correlated to the TCP window state. Usually, in the TCP slow start the TCP window is very small. After the first/second video segments the size of a video segment can be much larger. Since our dataset contains videos with motion we can filter out the audio correctly but for a music video with static pictures the video and audio data look similar and in those cases further research that will estimate the content type and video quality is needed. It is noteworthy that the segment real size in Fiddler is not necessarily the same in its network traffic representation, which can be affected by several factors explained before.

Table I summarizes the pre-processing methods' effect on the testing results with the full training dataset. The best accuracy for the SVM+RBF were achieved with pre-processing (72% accuracy). In our proposed solution, the result gave the best performance without pre-processing, achieving 98% accuracy. For the rest of the paper we will use the SVM+RBF classifier with pre-processing and our classifier without pre-processing. Fig. 7a depicts the SVM+RBF confusion matrix,

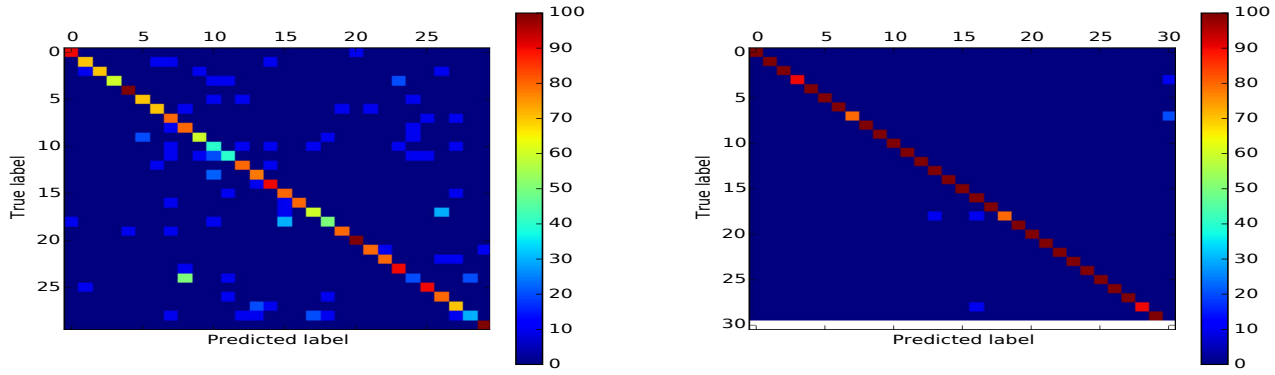
| Algorithm | with pre-processing | without pre-processing |
|---|---|---|
| NN | 95.66% | 98% |
| SVM+RBF | 72% | 66.66% |

TABLE I: The effect of pre-processing on algorithms accuracy.

while our confusion matrix is illustrated in Fig. 7b. In Fig. 7b index number 30 in the matrix shows that the stream was not classified against the dataset.

### *D. Evaluation of Robustness to Delay and Packet Losses*

In this section we artificially added additional packet loss and delay to the network (which already has natural packet loss




(a) SVM+RBF confusion matrix results.

(b) Our confusion matrix results.

Fig. 7: Classification confusion matrices results.

and delay). Packet loss has a very strong influence over the player representation selection. We observed that the quality degraded from $720P$ in the regular scenarios to $240P$ and $144P$ when encountering high packet loss ratio (9%). It is important to note that in our testing we added additional packet loss before the data capturing. Since our training data set was built from normative network traffic (low packet loss), these are conditions that the classifiers were not trained for. As a result, in this scenario, we can expect a low accuracy. Fig. 8a plots our algorithm and SVM robustness to packet losses. We can observe that the SVM classifier is much more vulnerable to packet drop compared to our suggested method. When encountering $1\%$ packet loss the SVM classifier drops from $72\%$ to $51\%$ while our algorithm was not affected. When the packet loss increases to $9\%$ we can see that our solution achieved $71\%$ accuracy while the SVM classifier achieved $20\%$ accuracy only.

Fig. 8b depicts our algorithm's robustness to various network delay. It is interesting to observe that both algorithms were more sensitive to delay compared to packet loss. When the player encounters severe delays it rebuffers a lot. We can see that after $100$ $[ms]$ the SVM accuracy was decreased to $50\%$ while the proposed solution accuracy was decreased to $96\%$.

### *E. Evaluation of Other Titles*

In our scenario the adversary sniffs the network and searches for a finite number of video titles (closed world assumption). The network traffic, on the other hand, contains many more titles which are not in the database - *unknown* titles. Therefore, it is important to evaluate the accuracy for unknown titles. We used 200 different titles that are not part of the training titles. We observed that not even one stream was mistakenly classified as one of our training titles. In future work, we will extend the number of unknown titles and investigate the subject further.

## V. Conclusions

In this work we presented a novel HTTP encrypted adaptive streaming video title classification algorithm for YouTube with Chrome browser. The solution was evaluated with two classification algorithms: SVM+RBF and NN. We evaluated the classification accuracy with titles inside and outside our dataset and found that our algorithm has high accuracy.

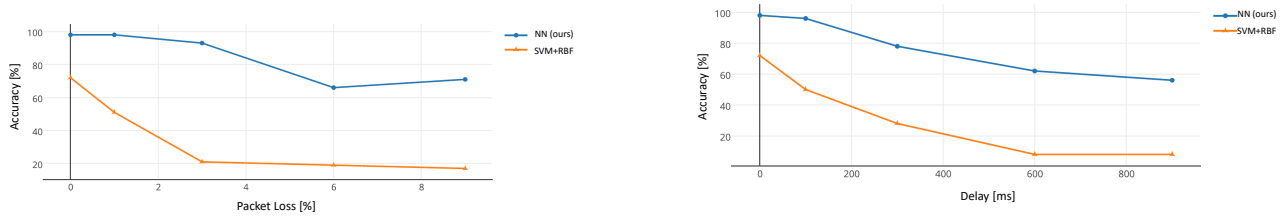
We found that exploiting the VBR nature of the audio and video encoded data can be suitable to detect the title name. We found that the data set size doesn't need to be large in-order to achieve good classification results. However, the dataset should represent all network conditions with all possible browser resolutions in order to effectively recognize each title. We open source our dataset [20] and Chrome encrypted data crawler [21] for future research in the area.

## VI. Acknowledgment

This work was partly supported by the Israel National Cyber Bureau.

## References

[1] Cisco. The zettabyte era: Trends and analysis, 2015.
[2] ISO/IEC. Information technology - Dynamic adaptive streaming over HTTP (DASH), May 2014.
[3] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hofeld, and P. Tran-Gia. A Survey on Quality of Experience of HTTP Adaptive Streaming. *IEEE COMMUNICATION SURVEYS AND TUTORIALS*, 17(1):469–492, 2015.
[4] M. R. Izquierdo and D. S. Reeves. A survey of statistical source models for variable bit-rate compressed video. *Multimedia System*, 7(3):199–213, 1999.
[5] Hypertext Transfer Protocol Version 2. M. Belshe and R. Peon and M. Thomson. RFC 7540, IETF, May 2015.
[6] Sandvine. Sandvine global internet phenomena report h1 , 2014, 2014.
[7] Google. Google webmaster central blog: Https as a ranking signal, august, 2014, 2014.
[8] M. Crotti, F. Gringoli, P. Pelosato, and L. Salgarelli. A statistical approach to IP level classification of network traffic. In *International Conference on Communications*, pages 170–176, June 2006.
[9] T. Okabe, T. Kitamura, and T. Shizuno. Statistical traffic identification method based on flow-level behavior for fair VoIP service. In *IEEE Workshop on VoIP Management and Security*, pages 35–40, April 2006.




(a) Accuracy results for additional packet loss percentage

(b) Accuracy results for additional LAN network delay

Fig. 8: Accuracy results for different network conditions.


[10] Dawn Xiaodong Song, David Wagner, and Xuqing Tian. Timing analysis of keystrokes and timing attacks on ssh. In *Proceedings of the 10th Conference on USENIX Security Symposium - Volume 10*, pages 25–25, 2001.

[11] Brice Canvel, Alain Hiltgen, Serge Vaudenay, and Martin Vuagnoux. Password interception in a ssl/tls channel. In *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 583–599. Springer Berlin Heidelberg, 2003.

[12] T. Scott Saponas, Jonathan Lester, Carl Hartung, Sameer Agarwal, and Tadayoshi Kohno. Devices that tell on you: Privacy trends in consumer ubiquitous computing. In *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, pages 5:1–5:16, 2007.

[13] Yali Liu, C. Ou, Zhi Li, C. Corbett, B. Mukherjee, and D. Ghosal. Wavelet-based traffic analysis for identifying video streams over broadband networks. In *IEEE Global Telecommunications Conference*, pages 1–6, Nov 2008.

[14] Yali Liu, Ahmad-Reza Sadeghi, Dipak Ghosal, and Biswanath Mukherjee. Video streaming forensic content identification with traffic snooping. In *Information Security*, volume 6531 of *Lecture Notes in Computer Science*, pages 129–135. Springer, 2011.

[15] AM. White, AR. Matthews, KZ. Snow, and F. Monrose. Phonotactic reconstruction of encrypted voip conversations: Hookt on fon-iks. In *Security and Privacy (SP), 2011 IEEE Symposium on*, pages 3–18. IEEE, 2011.

[16] CV. Wright, L. Ballard, F. Monrose, and GM. Masson. Language identification of encrypted voip traffic: Alejandra y roberto or alice and bob? In *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, pages 1–12, 2007.

[17] R. Dubin, A. Dvir, O. Pele, O. Hadar, I.Raichman, and O. Trabelsi. Real time video quality representation classification of encrypted http adaptive video streaming-the case of safari. In *IEEE Digital Media Industry and Academic Forum, Santorini, Greece*. IEEE, 2016.

[18] J. Muehlstein, Y. Zion, M. Bahumi, I. Kirshenboim, R. Dubin, A. Dvir, and O. Pele. Analyzing HTTPS encrypted traffic to identify user operating system, browser and application. In *CCNC-Workshop*, 2017.

[19] S. Agarwal, A. Sureka, and V. Goyal. Open source social media analytics for intelligence and security informatics applications. In *International Conference on Big Data Analytics*, pages 21–37. Springer, 2015.

[20] Dataset. The research dataset. http://www.cse.bgu.ac.il/title_fingerprinting/.

[21] Crawler. The research chrome youtube encrypted network traffic crawler. https://github.com/randubin/YouTube_video_title_downloader.

[22] R. Dubin, O. Hadar, A. Noam, and R. Ohayon. Progressive download video rate traffic shaping using tcp window and deep packet inspection. In *WORLDCOMP*, 2012.

[23] Fiddler-The Free Web Debugging Proxy by Telerik. http://www.telerik.com/fiddler, 2012.

[24] P. Velan, M. Čermák, P. Čeleda, and M. Drašar. A survey of methods for encrypted traffic classification and analysis. *International Journal of Network Management*, 2015.

[25] Z. Cao, G. Xiong, Y. Zhao, Z. Li, and L. Guo. A survey on encrypted traffic classification. In *Applications and Techniques in Information Security*, pages 73–81. Springer, 2014.

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

[27] S. Valenti, D. Rossi, A. Dainotti, A. Pescapè, A. Finamore, and M. Mellia. Reviewing traffic classification. In *Data Traffic Monitoring and Analysis*, pages 123–147. Springer, 2013.

[28] A. Dainotti, A. Pescape, and KC. Claffy. Issues and future directions in traffic classification. *Network, IEEE*, 26(1):35–40, 2012.

[29] Vern Paxson. Empirically derived analytic models of wide-area tcp connections. *IEEE/ACM Transactions on Networking (TON)*, 2(4):316–336, 1994.

[30] R. Alshammari and AN. Zincir-Heywood. Unveiling skype encrypted tunnels using gp. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–8. IEEE, 2010.

[31] S. Zander, T. Nguyen, and G. Armitage. Self-learning ip traffic classification based on statistical flow characteristics. In *Passive and Active Network Measurement*, pages 325–328. Springer, 2005.

[32] D. Zhang, C. Zheng, H. Zhang, and H. Yu. Identification and analysis of skype peer-to-peer traffic. In *Internet and Web Applications and Services (ICIW), 2010 Fifth International Conference on*, pages 200–206, 2010.

[33] I. Paredes-Oliva, I. Castell-Uroz, P. Barlet-Ros, X. Dimitropoulos, and J. Sole-Pareta. Practical anomaly detection based on classifying frequent traffic patterns. In *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, pages 49–54, 2012.

[34] D. Bonfiglio, M. Mellia, M. Meo, and D. Rossi. Detailed analysis of skype traffic. *Multimedia, IEEE Transactions on*, 11(1):117–127, 2009.

[35] KT. Chen, CY. Huang, P. Huang, and CL. Lei. Quantifying skype user satisfaction. 36(4):399–410, 2006.

[36] E. Hjelmvik and W. John. Statistical protocol identification with spid: Preliminary results. In *Swedish National Computer Networking Workshop*, 2009.

[37] R. Bar-Yanai, M. Langberg, D. Peleg, and L. Roditty. Realtime classification for encrypted traffic. In *Experimental Algorithms*, pages 373–385. Springer, 2010.

[38] Ashwin Rao, Arnaud Legout, Yeon-sup Lim, Don Towsley, Chadi Barakat, and Walid Dabbous. Network characteristics of video streaming traffic. CoNEXT, 2011.

[39] Pablo Ameigeiras, Juan J. Ramos-Muoz, Jorge Navarro-Ortiz, and Juan M. Lpez-Soler. Analysis and modelling of youtube traffic. *TETT*, 2012.

[40] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[41] Selenium. http://www.seleniumhq.org/. Accessed: 2016-02-28.

[42] Chromedriver - webdriver for chrome. https://sites.google.com/a/chromium.org/chromedriver/. Accessed: 2016-02-28.