Computer Communications 109 (2017) 24–37

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Video frame size modeling for user-generated traffic in an enterprise-like environment

Marios Kastrinakis[a], Ghada Badawy[b], Mohammed N. Smadi[c], Polychronis Koutsakis[d,*]

[a] School of ECE, Technical University of Crete, Greece
[b] Computing Infrastructure Research Center, McMaster University, Canada
[c] ECE Department, McMaster University, Canada
[d] School of Engineering and Information Technology, Murdoch University, Australia

ARTICLE INFO

Article history:
Received 25 February 2017
Revised 12 May 2017
Accepted 15 May 2017
Available online 19 May 2017

ABSTRACT

Smart mobile devices have displaced personal computers in many daily applications such as internet browsing and email. However, for content creation, users still need to use a large display, keyboard and mouse. Many initiatives are currently working on enabling I/O functionality for content creation and peripheral access, and on preserving the grab-and-go experience where the mobile device is not tethered to the docking station but merely placed in proximity of it and the traffic is carried over Wi-Fi. Maintaining the Quality of Service (QoS) and Experience (QoE) of low-latency, high fidelity video (for example the desktop view of a smart device) when transmitted over a Wi-Fi link in heavily loaded environments has been proven problematic. In this work, we propose for the first time in the relevant literature to the best of our knowledge, a highly accurate video traffic model that is capable of predicting the volume of video traffic generated by an average user's computer during a day. Our modeling techniques are tested on real user-generated screen mirroring traffic from a large shared cube space similar to an enterprise environment, and can be easily used as source traffic generators in order to facilitate the study of H.264 transmission performance over wireless networks.

1. Introduction

Smart mobile devices such as smartphones and tablets are becoming more powerful every day with the advancement of mobile computing chips from the likes of Qualcomm, NVidia and Intel, while at the same time major software and operating system companies develop their products in a single code base suitable for many platforms [1]. In addition, a 2013 survey [2] placed smartphones' popularity at around 85%, surpassing all other kinds of computing devices. The above facts seem to point towards a future where smart mobile devices might replace computers completely, in personal and corporate environments.

On the other hand, users still want to use a large display, keyboard and mouse for content creation and when they are not on the move. This use case has been addressed using tether technology such as the Mobile High-Definition Link (MHL) [38], however to maintain the grab-and-go experience there is a trend of replacing the need for cables with a wireless link that connects the mobile device to a wireless docking station. In such a setup, the video on the mobile device display is mirrored unto a larger display with low-latency to ensure the interactivity of the end-user is maintained. The actual latency limits are application dependent. Due to its attachment rate in the mobile industry, Wi-Fi is a key candidate for carrying this docking traffic and Miracast [3], which was recently ratified as a standard to allow mobile device display mirroring over Wi-Fi, is also a candidate for enabling the video component of any docking station. At the heart of a Miracast source, an H.264 encoder streams over a Wi-Fi peer-to-peer (P2P) mode without needing access to an overlaid Wi-Fi network. However, in enterprise environments where an overlaid Wi-Fi network is not only present but heavily used, contention due to medium access between the Wi-Fi P2P docking link and the overlaid Wi-Fi network will have a direct impact on the QoE for the docking video. The impact and potential QoE degradation will depend, among many other factors, on the characteristics of the video traffic, which in turn is dependent on the video content being compressed, the encoder implementation and the target latency

* Corresponding author.
E-mail addresses: makastrinakis@electronics.tuc.gr (M. Kastrinakis), gbadawy@gmail.com (G. Badawy), smadim2@mcmaster.ca (M.N. Smadi), p.koutsakis@murdoch.edu.au (P. Koutsakis).

http://dx.doi.org/10.1016/j.comcom.2017.05.008

profile. For example, video chatting on mobile devices demands high bandwidth and if the offered bandwidth is decreased video quality may be compromised [41].

Streaming over best-effort networks that were not designed to provide stable QoS, especially wireless networks, makes it inefficient to use the same representation of a video for the duration of a streaming session. Instead, it must be adapted to dynamically varying networks conditions such as throughput, packet loss rate, and delay jitter [42]. These problems can be significantly mitigated if the volume of video traffic that will be generated in the network can be predicted.

Traffic characterization and modeling of multimedia services are required for an efficient network operation. The generated models can be used as traffic rate predictors, during the network operation phase (online traffic modeling), or as video generators for estimating the network resources, during the network design phase (offline traffic modeling). In the offline case, traffic models can be used as video generators, to select appropriate network parameters during the network design phase, such as utilization, and/or number of multiplexed sources that achieve an acceptable video quality. In this framework, the reliability of the network can be evaluated. For example, the probability of refusing a new video call or the probability of network overload can be estimated. On the other hand, online traffic models are very useful for traffic management algorithms and congestion control schemes, which prevent the network from possible overload [51]. In such schemes, the emphasis is to increase the network resources utilization while maintaining the desired level of QoS. Dynamic resource allocation (DRA) schemes are especially important for live streams, where the video stream characteristics are not known in advance. In order to provide an accurate estimation of the needed network resources for a certain flow, which indicate the cost of transmitting such a flow, the chosen DRA scheme has to be able to predict the required bandwidth for future video frames. To adjust the bandwidth assignment for a certain video stream, DRA renegotiates the assigned bandwidth for that flow. The main goals for a DRA scheme are: to predict the longest possible period with the least prediction error, and to provide the best possible resource utilization with the lowest achievable frame delay [50]. This will ensure that no degradation takes place in the QoE of accepted video sessions [43].

Hence, in this work we develop H.264 video traffic models for low-latency, candidate Miracast source implementation for content that resembles a typical desktop user in an enterprise-like environment. We discuss well-known models, which fail to accurately capture this type of video traffic and we propose a highly accurate model based on the combination of clustering with Markov chains and the use of the Jaccard index similarity coefficient. To the best of our knowledge, this is the first time that a model tested on real user-generated screen mirroring traffic is developed and presented. It is also the first time that the concept of the Jaccard index is used for video traffic modeling purposes. We make our dataset available to the community so that possible other modeling approaches can be tried by other researchers.

The paper is structured as follows. Section 2 discusses related work. Section 3 presents the video traffic encoding of the data that we worked with. Section 4 includes the data collection methodology, data statistics and the statistical tests that we have used during the development and testing of our models. In Section 5 we analyze our models and discuss the respective results. Section 6 includes our conclusions and ideas for future work.

2. Related work

Multiple efforts on video traffic modeling have been conducted in the literature. Video models which have been proposed include first order autoregressive (AR) models [5], discrete autoregressive (DAR) models [4,6,7,40], Markov renewal processes (MRP) [8], MRP transform expand sample (TES) [9,10], finite state Markov chain [11,12], a combination of wavelet and linear modeling [20] and gamma beta auto regression (GBAR) models [13,14]. In [15] the authors analyzed a number of mobile video streams and created a model that provides both video frame and RTP packet generators. The model was created and verified against “The Matrix” and “Lord of the Rings” movies. In [16], the authors create a video traffic model that takes interdependence between different frame types into consideration (I, P and B frames). The authors in [17] list a number of Variable Bit Rate (VBR) video traffic models and compare these models against three video traces “Star Wars IV”, “Tokyo Olympics” and “NBC 12 News”. They showed that some of the models work for some videos but not for others. They could not find a universal model that works with all types of videos. Other recent efforts include [45] in which the authors propose their models on top of the model in [20] and they exploit the hierarchical prediction structure inherent in H.264 for intra-GoP (Group of Pictures); [46], in which the authors propose a non-linear autoregressive model for long-range video traffic prediction (without separating traffic of different frame types) and they introduce adaptive algorithms to obtain the parameters of their model; [32], where the authors use linear regression to predict B-Frames’ sizes, with the goal of possibly dropping B-Frames (based on the prediction) in the case of network congestion. The works in [47,48] focus on modeling new types of video traffic, such as 4K and 3D video. The authors in [47] use a seasonal autoregressive model for modeling and prediction of 4K video traffic. They analyze over 17,000 video frames and show that their proposed methodology provides good accuracy in high definition video traffic modeling. The work in [48] proposes and evaluates a new model for multiview video (which is used to support 3D video applications) that is based on a Markov process.

The above-referenced papers model video traces from movies, which are significantly different, in terms of content, than those created by desktop applications used in an enterprise environment. Our goal in this study is to fill this gap by studying these applications and the video traffic they generate, in order to build a highly accurate model. There is very little work done on what applications are mostly used in enterprise environments. In [18] the author shows that employees use Microsoft Outlook the most from the Microsoft Office suite. The author does not mention which other applications are used. Also there is no characterization of the video generated by the Microsoft Office suite or similar tools.

3. Video encoding

We worked with two different datasets, encoded with the H.264 video coding standard. H.264 or MPEG-4 Part 10, AVC is a video coding standard developed by ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG). It is the most widely accepted video coding standard (since MPEG-2) and it covers a wide area of video applications ranging from mobile services and videoconferencing to IPTV, DTV and HD video storage [19]. According to the H.264 standard, an encoded video trace features two distinct characteristics: (1) every video frame comes from one of three different types of frames, and (2) video frames are organized in groups with a specific structure.

There are three different types of frames, I-Frames (Intra-coded Frames), P-Frames (Predicted Frames) and B-Frames (Bi-directional predicted Frames). P-Frames are smaller than I-Frames and B-Frames are the smallest [22]. Video frames are grouped together in Group of Pictures (GOP) structures that specify the order in which intra- and inter-frames are arranged. A GOP pattern specifies the amount and order of P and B-Frames between two successive I-Frames. Every GOP contains a single I-Frame with which it starts.

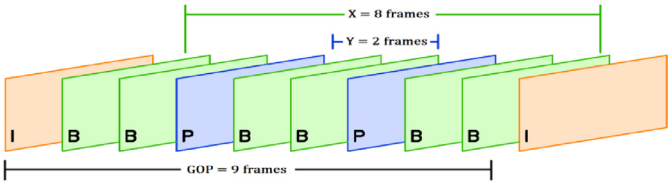

Fig. 1. Graphical example of a GOP structure.

The GOP pattern is defined by the distance *X* between I-Frames and the distance *Y* between P-Frames or between the I-Frame and the succeeding P-Frame. For example, in Fig. 1 we present a GOP structure of 9 frames, where *X* distance equals to 8 and *Y* distance equals to 2. In general, in the H.264 standard the amount of B-Frames is greater than the amount of I or P-Frames inside a GOP structure.

4. Data collection methodology and statistical tests

Our work is based on real user-generated data from a large shared cube space resembling an enterprise environment. Each participant in the data collection ran trace collection scripts for about a month. One script polled the operating system every 33.3 ms to record the name of the main application that the participant was working on. Another script recorded the participant's screen at 30 fps using encoding parameters that resembled a Miracast hardware encoder as much as possible (more details on this in Section 4.2.1). The actual video was not recorded, but only the statistics of the encoded video were collected (i.e., I and P frame sizes). The scripts started automatically each work day at 8 am and stopped at 7 pm. When a participant locked his/her screen the scripts would report that the user is idle for that duration and video traces collection would stop till the user unlocks his/her machine. All of the users were using Windows 7 machines.

Some details on the network architecture follow:

A smartphone will either host applications natively (e.g. Windows Mobile Phone) or will act as a thin client to a backend computing platform (e.g. Citrix Thin Client running on Android Phone). In both instances the actual applications will be rendered by the smartphone GPU before being re-encoded and sent over Wi-Fi to a Miracast dongle connected to a large display. Applications will appear on the phone and on the large display exactly as if they were being rendered onto a desktop PC with one application in the foreground and multiple applications in the background (i.e. either docked or in the background of the display, so users were allowed to use multiple applications at a time). The Windows 7 setup was merely used to capture the application that an end-user has in the foreground and to capture and feed the raw video data (i.e. the output of the GPU showing foreground and background applications side by side) into a carefully configured video encoder (i.e. FFmpeg, as explained below). This encoder is configured to "mimic" the configuration of an H.264 Miracast source configuration running a mobile phone. The encoder configuration parameters have been set using datasheets of Miracast sources.

4.1. Recording methods

A recording framework was deployed on every host machine. It was running and logging in the background during the recording period. The FFmpeg [23] program was used for video traffic recording. It logged the compressed H.264 video information (i.e., frames sizes, GOP structure, frames' time of arrival, etc.) of the host's machine desktop. The frame resolution was the same as the PC's screen resolution, i.e., it is not a constant; we used different resolutions depending on the different PC screens that were being used. The default windows resolution is 1920 × 1080, so this was often the case for our videos. The frame rate was 30 fps. It is worth mentioning that even though FFmpeg was running constantly, it was capable to log video traffic information only if the host's machine GPU was active (i.e., the host machine was not in hibernation, sleep or monitor energy saving mode). As for active applications usage recording, a Windows PowerShell [24] script was used for logging the name of the application in the foreground, followed by the current timestamp. Windows PowerShell was programmed to log the application's name every 33.3 ms (in order to keep up with FFmpeg logs, where we had one frame every 33.3 ms). We should also note that Windows PowerShell is capable to report the application's name only if the host machine is unlocked and the user is not logged off. The average PSNR of our videos was 65 dB.

4.2. Datasets overview

4.2.1. Encoding

In this study, we worked with two different types of datasets. The main difference between the two datasets lies in the different encoding of video traces. The first dataset (Dataset 1) has been encoded with the High 4:2:2 Profile of the H.264 standard, which is typical for professional applications. This profile can generate I, P and B frames. However, in our datasets the – *tune zero latency* command was used in FFmpeg to prohibit the encoder from producing B-Frames, in order to minimize latency. For this dataset, we have a GOP structure of 60 frames in length, where every GOP starts with an I-Frame and the rest 59 frames are of type P. The second dataset (Dataset 2) has been encoded with different encoding parameters. Those parameters try to resemble a Miracast hardware encoder as closely as possible. Since I-Frames sizes are much larger than P-Frames, Miracast encoders do not use I-Frames but use Periodic Intra Refresh [25] instead. This enables each frame (in our case each I-Frame) to be capped to the same size by using a column of intra blocks that move across the video trace from one side to the other, thereby "refreshing" the image. In effect, instead of a big keyframe (in our case an I-Frame), the keyframe is spread over many frames (in our case P-Frames). For this dataset, we do not have a GOP structure. We only have P-Frames with an exception of one I-Frame whenever the host computer starts or its user logs on.

4.2.2. Recording periods and datasets statistics

Our recording framework ran on different periods of time, between March and May 2015 for the first dataset and between June and July 2015 for the second dataset. We replaced every user's name with a different letter from the alphabet for reasons of anonymity. In Table 1 we present some general statistics of our two datasets, such as general information about our records, as well as total, minimum, average and maximum sizes of our video traffic frames over all applications. It is worth mentioning, that the average P-Frame size in Dataset 2 is larger by a factor of ≈14 in comparison with the P-Frames in Dataset 1, as shown in Table 1. Our collected data, for both datasets, can be downloaded from [44].

4.3. Statistical tests and evaluation metrics

We developed and tested various modeling techniques on our data, as we analyze further in Section 5. For our modeling, it was necessary to try to fit our data with a number of well-known distributions. These fitting attempts, together with the statistical tools used for assessing the accuracy of the fits, are presented in this section. We also present the metrics we utilized for assessing the quality of our proposed models, which will be presented in Section 5. In the following two subsections, we explain the procedure that we have followed in order to try to fit our data

Table 1
Dataset 1 and Dataset 2 statistics over all applications.

Statistics	Dataset 1	Dataset 2
# of recording days	24	22
# of users	3	4
# of applications	29	26
# of video traffic records	14,932,183	20,892,611
Total size of video traffic (GB)	120	424
MIN video traffic size (B)	159	190
AVG video traffic size (B)	8032	20,298
MAX video traffic size (B)	598,613	422,435
# of I-Frames	249,061	290
Total size of I-Frames (GB)	99	0092
MIN I-Frame size (B)	3290	129,932
AVG I-Frame size (B)	397,525	316,900
MAX I-Frame size (B)	1,536,577	395,780
# of P-Frames	14,683,122	20,892,321
Total size of P-Frames (GB)	21	423,908
MIN P-Frame size (B)	159	190
AVG P-Frame size (B)	1425	20,293
MAX P-Frame size (B)	598,613	422,435

with a number of well-known distributions. The data fitting procedure consists of two basic steps. The first is the parameters estimation method for each chosen distribution. The second is the data generation method in order to reproduce data according to the specific distribution.

4.3.1. Maximum likelihood estimation

In order to find the parameters of a distribution based on our data, we used the Maximum Likelihood Estimation (MLE) method. In general, given a statistical model, MLE returns estimates for the model's parameters at a confidence level *alpha* (usually alpha = 95%). In our case, the model is a distribution which we want to investigate on whether it underlies our data and we want to confirm or reject this assumption. Before that, due to the fact that every distribution has a vector Θ that contains its parameters, we need to find an estimation $\hat{\Theta}$ of this vector, based on our data. Hence, we used the MLE method in order to seek a vector $\hat{\Theta}$, which can be as close as possible to the true Θ, in order to estimate the parameters of the distribution assumed to underlie our data.

We used a number of distributions that are well-known in the literature for various types of video traffic characterization and modeling. More specifically, we studied the Uniform, Exponential, Gamma, Lognormal, Geometric, Negative Binomial, Generalized Extreme Value (GEV), Weibull, Pearson Type V and Log-logistic distribution. In order to generate random data according to these distributions, we used the MLE method for the parameters estimation and the built-in Matlab functions for the data generation.

4.3.2. Statistical tests

We have used three statistical tests during this work, in order to evaluate the accuracy of the distributions fits with our data. We briefly present the three tests below.

4.3.2.1. Q–Q plots. The Quantiles–Quantiles Plot or Q–Q Plot is a powerful Goodness of Fit (GoF) test [27] which compares two datasets graphically, in order to determine whether the datasets come from populations with a common distribution and statistical characteristics. If they do, the point of the plot should lie along a 45-degree reference line approximately, which passes from the axis start point. A Q–Q plot is a plot of the quantiles of the data versus the quantiles of the fitted distribution. A z-quantile of X is any value x such that $P(X \leq x) = z$. In our case, we have plotted the quantiles of the real data versus the quantiles of the data generated via the distribution.

4.3.2.2. Kolmogorov–Smirnov test. In order to further verify the validity of our results, we used the Kolmogorov–Smirnov test [29]. The Kolmogorov-Smirnov (KS) test tries to determine if two datasets differ significantly. It has the advantage of making no assumption about the distribution of data, i.e., it is non-parametric and distribution free. The KS test uses the maximum vertical deviation between the two curves as its statistic D. Although it is a good statistical tool, the KS test has the drawback that it gives the same weight to the difference between the actual data and the fitted distribution for all values of data, whereas many compared distributions differ primarily in their tails. It tests if the null hypothesis is accepted or rejected at an *alpha* significance level (usually alpha = 5%). The null hypothesis is that the population we are testing is drawn from a specific distribution with *alpha* chance of error.

The KS test can also be used, the way we use it in this study, as a goodness of fit test. This means that we do not actually expect to see if the test accepts or rejects a null hypothesis (even though it would be an excellent result if the null hypothesis was accepted) but to see how "far" the actual data are from the fitted distribution. This is the Two-Sample Kolmogorov–Smirnov Test. The test measure is given by Eq. (1) for two given Cumulative Distribution Functions (CDFs) F_1 and F_2,

$$D_{n,n'} = \sup\left(|F_{1,n}(x) - F_{2,n}(x)|\right) \tag{1}$$

The null hypothesis is rejected at the level "a" significance if:

$$D_{n,n'} > \; c(a) \cdot \sqrt{\frac{n \; + \; n'}{n \cdot \; n'}} \tag{2}$$

The values of $c(a)$ are defined for various significance levels and n and n' are the number of samples. We should bear in mind that the Two-Sample Kolmogorov–Smirnov Test only tells us half the tale, meaning that it only tells us the maximum distance between two distributions and not which distribution our data come from.

Finally, it should be mentioned that the KS test has two limitations. First, it works only with continuous distributions (hence it cannot provide results for the Geometric and the Negative Binomial distribution) and second, it is more sensitive at the "center" of the CDFs of the distributions rather than the "tails". We try to address this limitation by using the Anderson-Darling test.

4.3.2.3. Anderson–Darling test. The Anderson–Darling (AD) test is a modification of the KS test [30] that it is more sensitive at the "tails" of the CDFs of the distributions rather than the "center". It belongs, like as KS test, to the family of Quadratic Empirical Distribution Function statistics, which measures the distance between the empirical CDF, $F_n(x)$ and the hypothesized CDF, $F(x)$ as

$$\mathrm{D}\,(F_n,\, F) = n \int_{-\infty}^{\infty} (F_n(x) - F(x))^2 w(x)\, \mathrm{d}F(x) \tag{3}$$

over the ordered sample values $x_1 < x_2 < \ldots < x_n$, where $w(x)$ is a weight function that favors the "tails" of the CDF and n is the number of samples in the dataset. The weight function for the AD test is

$$w(x) \; = \; [F(x) \; \cdot \; (1 - F(x))]^{-1} \tag{4}$$

The AD test statistic is

$$A_n^2 \; = -n - \sum_{i=1}^{n} \frac{2i \; - 1}{n} \; [\ln\left(F(X_i)\right) + \ln\left(1 - F(X_{n+1-i})\right)] \tag{5}$$

where $X_1 < X_2 < \ldots < X_n$ are the ordered sample values and n is the number of samples in the dataset. Even though the KS test is distribution free, there is a form of the AD test that is not. It makes use of the specific distribution parameters to be evaluated. The appropriate critical values need to be selected for the distribution we wish to check. This allows the test to be more sensitive but it also makes it impossible to use with a large variety

of distributions. Currently, tables of critical values exist for the Normal, Uniform, Lognormal, Exponential, Weibull, Extreme Value I, Generalized Pareto and Logistic distribution. In this study, we use the non-parametric version of the AD test because we are testing distributions for which no known critical values exist.

4.4. Accuracy evaluation metrics

We present, below, the two metrics that we used, in order to evaluate the accuracy of our models.

The Mean Absolute Percentage Error (MAPE) is a widely used metric [31] that shows the average difference (i.e., the average error) between the real values and the corresponding measured (in our case predicted by our models) values. For a set of pairs of real-generated values in a dataset, the MAPE is calculated as:

$$\text{MAPE} = \frac{\sum_{i=1}^{n} \frac{|X_{P,i} - X_{R,i}|}{X_{R,i}}}{n} \cdot 100\% \quad (6)$$

where $\{X_{R,i}, X_{P,i}\}$ is the ith pair of real and generated values in a dataset of n pairs.

The Relative Percentage Error (RPE) [32] is a metric that shows the overall difference (i.e., the overall error) between the real values and the corresponding measured (in our case predicted by our models) values, as a percentage of the overall size of the real values. It depicts how large the prediction error is relative to the real values, in a percentage form. For a set of pairs of real-generated values in a dataset, the RPE is calculated as:

$$\text{RPE} = \frac{\sum_{i=1}^{n} |X_{P,i} - X_{R,i}|}{\sum_{i=1}^{n} |X_{R,i}|} \cdot 100\% \quad (7)$$

where $\{X_{R,i}, X_{P,i}\}$ is the ith pair of real and generated values in a dataset of n pairs.

5. Modeling and results

In this section, we present, analyze and evaluate the four different modeling approaches that we have used in this work. One model, which combines clustering with Markov chains and the use of the Jaccard index similarity coefficient, is proposed here for the first time in the relevant literature, to the best of our knowledge. We discuss why the model provides high accuracy in modeling video traffic generated by an average user's computer, during a day, while other models fail. All of our results have been derived via Matlab.

5.1. Application and Distribution Aware Model

The Application and Distribution Aware (ADA) Model, is the first and simplest approach that we have developed and tested, in order to model our data. It is based on the assumption that the video traffic of every unique application in our datasets is characterized by a distinct distribution.

The ADA model consists of two basic steps. The first is the parameters' estimation of each distribution that possibly characterizes the application using the MLE method and the dataset. The second is the predicted data generation according to this specific distribution using the estimated parameters of the previous step. Given that the distribution that best characterizes the application's data is unknown, the steps have to be repeated for a wide range of well-known distributions and the ones that give the lowest RPE and MAPE will be selected.

We applied the ADA model for the set of ten well-known distributions presented in Section 4.3.1. In order to evaluate the accuracy of the model, we applied the Q–Q plot, KS and AD tests for each case and we examined if those tests' results agree with those of RPE and MAPE.

In the beginning, we tried to model the size of the video traffic as-is, i.e., without separately modeling I and P-Frames. The problem with this approach was that the sizes of I-Frames differ significantly from the sizes of P-Frames in terms of minimum, average, standard deviation and maximum size (according to Table 1) and hence no distribution could serve as a competent model that would reproduce those wide range differences with low errors. Therefore, we proceeded to model I and P frames separately.

We should mention that we also tried to model our data per user and per day of capture separately but we concluded that the improvement to our results was not significant, given its much larger complexity.

Finally, it should be emphasized that the ADA model has two inherent disadvantages. The first is that the percentage of outliers among P frames varies and the rule for the split will need to be tailored to each specific dataset. The second is that it does not incorporate the autocorrelation between successive or neighbor video frames, which is well-known to exist either for short or long-term, i.e., Short Range Dependence (SRD) or Long Range Dependence (LRD) [33–35]. Our own results, which will be presented in the following sections, confirm that for all traces SRD exists but LRD is weak. Therefore, ADA is used as a first, simple approach for the modeling of P frames and as a benchmark against which our other models will be compared.

5.1.1. ADA model evaluation results

We ran the ADA model for every distinct application of Dataset 1 and Dataset 2. There are applications in Dataset 2 without results for the I-Frames due to the fact that according to Dataset's 2 encoding, we have only one I-Frame every time the host computer starts or its user logs on.

We first present the results for the I-Frames and P-Frames ADA modeling of Dataset 1 and Dataset 2, according to the RPE, MAPE metrics and the Q–Q plot, KS and AD tests and over all applications.

From Tables 2 and 3 that refer to Dataset 1, we observe that the ADA model achieves good accuracy on modeling I-Frames but fails in modeling P-Frames. For the I-Frames, the errors are below 10% for most applications. Additionally, in most cases the best distribution per application (the one leading to the smallest error) is the same when both RPE and MAPE are used. No clear conclusion on the best distribution can be derived from the KS and AD test, however, a fact that indicates that our data differs between the "center" and the "tails". As for the P-Frames, we observe very high errors in terms of RPE and significant errors in terms of MAPE. Also, we observe that MAPE, as well as the KS and AD tests indicate for most of the applications that GEV gives the best distribution fit, a fact related with the high concentration of very small sized P-Frames in Dataset 1.

From Tables 4 and 5 that refer to Dataset 2, we confirm again that the ADA model gives highly accurate results on the modeling of I-Frames (4–5% RPE and MAPE). For this dataset the model achieves better, but still not satisfactory results for the P-Frames (RPE 13.8%, MAPE 9.4%). Regarding P-Frames, we observe that there are several applications for which RPE, MAPE, KS and AD agree for the best distribution, which indicates that the video traffic is more "smoothed" due to the usage of Periodic Intra Refresh in comparison with Dataset 1. The reason that the number of applications in Table 4 is smaller than that in Table 5 is that, as explained in Section 4.2.1, I-Frames are generated in Dataset 2 only whenever the host computer starts or its user logs on. Therefore, we have very few I-Frames and only for the applications that were open during the capturing initialization.

Table 2
ADA model results for I-Frames over all applications of Dataset 1.

Application	I-Frames					
	RPE		MAPE		KS	AD
	Best distribution	Error (%)	Best distribution	Error (%)	Best distribution	Best distribution
Acrobat Reader	GEV	14.0934	Gamma	23.0565	LogLogistic	GEV
Microsoft Excel	Weibull	11.8120	GEV	12.5411	LogLogistic	GEV
Foxit Reader	GEV	7.2657	LogLogistic	5.5538	LogLogistic	LogLogistic
InSite	GEV	9.4328	GEV	13.0520	GEV	GEV
Matlab	LogLogistic	2.7068	LogLogistic	3.7226	LogLogistic	LogLogistic
Microsoft Outlook	Weibull	2.9172	NegBinomial	3.0791	Gamma	GEV
Microsoft PowerPoint	Weibull	2.8837	Weibull	3.7738	LogLogistic	LogLogistic
Ent. Device Manager	Weibull	11.1717	Uniform	24.0001	Uniform	GEV
Snipping Tool	Uniform	11.9218	NegBinomial	13.3137	Gamma	LogLogistic
Microsoft Word	LogLogistic	3.9727	LogLogistic	3.3878	LogLogistic	LogLogistic
WinMerge	Weibull	2.5459	Weibull	2.6596	LogNormal	Weibull
WinSCP	LogLogistic	5.5527	LogLogistic	4.9612	LogLogistic	LogLogistic
Xwin Cygwin	GEV	14.1766	GEV	12.8750	Gamma	GEV
Windows Calculator	GEV	5.5712	GEV	9.0079	GEV	GEV
Google Chrome	Weibull	7.3737	Weibull	7.9375	LogLogistic	Weibull
Command Line	Weibull	4.7144	Weibull	6.7775	GEV	Weibull
Communicatior	PearsonV	5.4010	PearsonV	5.5730	PearsonV	GEV
Mozilla Firefox	LogLogistic	2.3355	LogLogistic	2.3476	LogLogistic	LogLogistic
Google Earth	GEV	3.2203	GEV	4.2721	GEV	GEV
G-Simple	GEV	20.0270	GEV	11.5566	LogLogistic	GEV
Internet Explorer	Weibull	3.5108	Weibull	4.0624	Weibull	Weibull
KDiff3	NegBinomial	2.2392	LogLogistic	2.4725	GEV	LogLogistic
Kile LaTeX	GEV	3.4498	GEV	4.8836	Weibull	GEV
Windows Paint	NegBinomial	9.1849	NegBinomial	9.7558	PearsonV	PearsonV
Windows Notepad	PearsonV	29.7553	LogLogistic	23.0799	GEV	GEV
Notepad++	Weibull	4.9762	GEV	5.2875	GEV	Weibull
Windows PowerShell	LogLogistic	3.6729	LogLogistic	4.7701	GEV	LogLogistic
Windows Task Manger	GEV	7.3913	GEV	10.9649	GEV	GEV
VLC	GEV	9.8352	GEV	16.5711	Weibull	GEV
	Average error (%)	7.6935	Average error (%)	8.8033		

Table 3
ADA model results for P-Frames over all applications of Dataset 1.

Application	P-Frames					
	RPE		MAPE		KS	AD
	Best distribution	Error (%)	Best distribution	Error (%)	Best distribution	Best distribution
Acrobat Reader	GEV	65.5001	GEV	6.0566	GEV	GEV
Microsoft Excel	GEV	41.8876	GEV	9.9055	GEV	GEV
Foxit Reader	PearsonV	83.9585	GEV	15.8851	GEV	GEV
InSite	GEV	71.3259	GEV	11.6295	GEV	GEV
Matlab	GEV	79.7061	GEV	11.9074	GEV	GEV
Microsoft Outlook	PearsonV	80.5754	GEV	12.3231	GEV	GEV
Microsoft PowerPoint	PearsonV	75.6485	GEV	16.7599	GEV	GEV
Ent. Device Manager	Weibull	49.7211	PearsonV	45.7621	GEV	GEV
Snipping Tool	GEV	35.4452	GEV	12.0030	GEV	GEV
Microsoft Word	GEV	72.1663	GEV	9.3328	GEV	GEV
WinMerge	PearsonV	79.4037	GEV	17.9503	GEV	GEV
WinSCP	GEV	76.8190	GEV	10.0013	LogLogistic	GEV
Xwin Cygwin	PearsonV	52.1366	GEV	24.0481	GEV	GEV
Windows Calculator	GEV	68.5243	GEV	8.7824	GEV	GEV
Google Chrome	GEV	73.9916	GEV	11.0672	GEV	GEV
Command Line	GEV	71.2440	GEV	11.1097	GEV	GEV
Communicatior	PearsonV	67.8365	GEV	13.9133	GEV	GEV
Mozilla Firefox	Weibull	80.8619	GEV	22.7592	GEV	GEV
Google Earth	Weibull	78.7539	LogLogistic	37.7865	GEV	GEV
G-Simple	GEV	51.8762	GEV	10.8519	GEV	GEV
Internet Explorer	GEV	85.9765	GEV	12.7529	GEV	GEV
KDiff3	Weibull	81.8432	GEV	32.4424	GEV	GEV
Kile LaTeX	PearsonV	69.3276	GEV	26.4580	GEV	GEV
Windows Paint	PearsonV	60.1628	GEV	14.9701	GEV	GEV
Windows Notepad	GEV	23.1330	GEV	7.2919	LogNormal	GEV
Notepad++	PearsonV	81.3766	GEV	14.0268	GEV	GEV
Windows PowerShell	GEV	53.3213	GEV	14.4497	LogLogistic	GEV
Windows Task Manger	GEV	66.3710	GEV	14.2695	GEV	GEV
VLC	Weibull	60.2636	PearsonV	36.2930	GEV	GEV
	Average error (%):	66.8675	Average error (%):	16.9927		

Table 4
ADA model results for I-Frames over all applications of Dataset 2.

Application	I Frames					
	RPE		MAPE		KS	AD
	Best distribution	Error (%)	Best distribution	Error (%)	Best distribution	Best distribution
Microsoft Excel	NegBinomial	3.9328	NegBinomial	3.8858	Uniform	LogNormal
Microsoft Outlook	GEV	4.0023	GEV	5.3359	Weibull	GEV
Microsoft PowerPoint	Weibull	5.1782	Weibull	6.1074	Gamma	Weibull
Microsoft Word	NegBinomial	5.1894	NegBinomial	6.0813	GEV	Weibull
Google Chrome	GEV	3.5474	LogLogistic	3.4277	LogLogistic	Weibull
Mozilla Firefox	GEV	5.6683	GEV	8.5764	LogLogistic	Weibull
Notepad++	Gamma	4.3898	Gamma	4.2871	Uniform	LogNormal
Windows PowerShell	LogLogistic	5.4552	LogLogistic	5.2876	Gamma	GEV
	Average error (%)	4.6704	Average error (%)	5.3736		

Table 5
ADA model results for P-Frames over all applications of Dataset 2.

Application	P Frames					
	RPE		MAPE		KS	AD
	Best distribution	Error (%)	Best distribution	Error (%)	Best distribution	Best distribution
Acrobat Reader	GEV	15.5873	GEV	13.1683	LogLogistic	GEV
Microsoft Excel	GEV	5.1208	GEV	3.6504	LogLogistic	GEV
Foxit Reader	LogLogistic	20.4611	LogNormal	9.6586	LogLogistic	LogLogistic
Matlab	NegBinomial	7.8308	LogNormal	6.0371	LogNormal	LogNormal
Microsoft Outlook	NegBinomial	6.6657	NegBinomial	5.2914	Gamma	Gamma
Microsoft PowerPoint	NegBinomial	6.5026	LogNormal	5.0167	Gamma	Gamma
Ent. Device Manager	GEV	12.8802	LogNormal	7.8266	Gamma	LogNormal
Snipping Tool	NegBinomial	5.6112	NegBinomial	5.4936	Gamma	Gamma
Microsoft Word	LogNormal	11.6892	LogNormal	4.5745	LogNormal	LogNormal
WinMerge	LogLogistic	30.5505	PearsonV	8.5241	PearsonV	GEV
WinRAR	LogLogistic	14.2884	LogNormal	8.8457	GEV	LogNormal
Xwin Cygwin	LogLogistic	8.1104	GEV	5.5709	LogLogistic	GEV
Windows Calculator	Weibull	12.8847	PearsonV	10.1943	PearsonV	PearsonV
Google Chrome	LogLogistic	21.8600	LogLogistic	7.2852	LogLogistic	LogLogistic
Command Line	Exponential	13.0109	LogNormal	10.5705	LogLogistic	LogNormal
Mozilla Firefox	LogLogistic	16.6754	LogNormal	7.9470	LogNormal	LogNormal
IrfanView	GEV	11.5825	LogNormal	5.2874	LogNormal	LogNormal
Internet Explorer	LogNormal	10.2767	LogNormal	4.6048	LogNormal	LogNormal
KDiff3	LogLogistic	18.7731	LogNormal	11.4546	LogLogistic	LogNormal
Windows Paint	Gamma	6.1956	GEV	4.5473	GEV	GEV
Windows Notepad	GEV	11.8933	GEV	7.6194	GEV	GEV
Notepad++	NegBinomial	8.6298	Gamma	6.4529	LogNormal	Gamma
Windows PowerShell	GEV	11.9188	NegBinomial	39.6663	GEV	GEV
Windows Task Manger	LogLogistic	11.1016	LogNormal	8.2597	LogNormal	LogNormal
VLC	LogLogistic	23.8497	LogLogistic	8.9453	LogLogistic	GEV
VMware Player	Weibull	35.2235	LogLogistic	28.3031	GEV	PearsonV
	Average error (%)	13.8144	Average error (%)	9.4152		

5.2. Gamma Beta Autoregressive Model (GBAR)

The results presented in Section 5.1 revealed that no single distribution can provide the best fit for each application. The distribution fit that provided the overall lowest RPE and MAPE when used for all applications was that of the Gamma distribution.

The rather poor ADA modeling results for P-Frames' sizes and the relatively better performance of the Gamma fit in most cases led us to implement and evaluate a very well-known model from the literature, the Gamma Beta Autoregressive (GBAR) Model [13]. GBAR has been proposed as a source model for VBR encoding of videoconference traffic (which is generally characterized by lower bit rates and motion is naturally more limited, hence it is closer in nature to screen mirroring traffic than standard movies). The main characteristic of the GBAR model is that the GBAR process is calculated based on a gamma distribution with parameters estimated from the dataset.

We have ran our model for every distinct application of Dataset 1 and Dataset 2, and separately for I and P frames. The results in Tables 6 and 7 show that the GBAR model offers worse accuracy in the modeling of I-Frames than the ADA model and fails clearly in the modeling of P-Frames. It should be noted that we only present results for two applications for the I-Frames of dataset 2. The reason is again that I-Frames are generated in Dataset 2 only whenever the host computer starts or its user logs on. Hence, in most applications the number of I-Frames was very small (less than 10) and the GBAR model could not be implemented.

The reason for the failure of the model in the case of P-Frames can be understood by studying the lag-1 autocorrelation values for both datasets. The average lag-1 autocorrelation of P-Frames is 0.3189 over all the applications of the 1st Dataset, for which the RPE and MAPE are very high. In Dataset 2, where the lag-1 autocorrelation is much higher (0.6239 on average, over all the applications) the GBAR model achieves lower errors for P-Frames than in Dataset 1, but still the errors are very high.

It should also be mentioned that the Hurst Parameter values of P-Frames range between 0.66 and 0.76 for all the applications under study for the first dataset, and between 0.68 and 0.82 for the second dataset. This indicates a time series with persistent behavior, and is further confirmed by the fact that P-Frame autocorrelations decayed hyperbolically, not exponentially, as shown in Figs. 2 and 4 below, which is another indication [37] of our traffic

Table 6
GBAR model results over all applications of Dataset 1.

Dataset 1	I-Frames		P-Frames	
Application	RPE (%)	MAPE (%)	RPE (%)	MAPE (%)
	Lag-1			
Acrobat Reader	9.4642	10.7000	63.5521	114.3876
Microsoft Excel	10.8625	12.0686	78.9786	129.6888
Foxit Reader	13.1874	14.9773	79.2341	200.5665
InSite	16.1095	18.4930	67.5287	119.0524
Matlab	9.3213	10.3228	90.2057	190.2595
Microsoft Outlook	10.9923	11.6934	73.9894	156.7260
Microsoft PowerPoint	10.2097	11.2995	88.1576	186.2553
Ent. Device Manager	18.0470	18.0709	45.6511	126.8678
Snipping Tool	2.5803	2.6052	42.4876	48.1055
Microsoft Word	8.6189	9.0612	70.6048	120.3422
WinMerge	0.3549	0.3558	59.0836	87.9919
WinSCP	10.9479	10.4257	86.8678	241.2759
Xwin Cygwin	31.0633	28.3023	42.4526	53.6665
Windows Calculator	8.2530	8.6519	84.7720	183.0306
Google Chrome	9.1101	10.1492	67.4956	124.0817
Command Line	7.9546	8.6795	56.5733	115.6246
Communicatior	9.3565	9.3141	87.7423	158.3650
Mozilla Firefox	11.1808	11.8360	80.9414	209.7440
Google Earth	13.1201	12.9532	77.4905	202.6313
G-Simple	17.3985	21.2785	53.0577	66.6445
Internet Explorer	12.1647	13.5991	74.8897	149.9265
KDiff3	9.4442	9.9702	93.7578	250.7903
Kile LaTeX	10.3123	13.3016	90.6473	233.6368
Windows Paint	18.6522	22.2498	80.4213	117.7523
Windows Notepad	14.1418	15.2191	62.0133	68.8571
Notepad++	11.5905	13.2449	76.1578	160.3242
Windows PowerShell	0.0009	0.0009	42.9225	64.9375
Windows Task Manger	16.9339	21.6649	62.1791	119.9986
VLC	19.6761	15.9164	57.2844	124.1358
Average error (%)	11.7603	12.6347	70.2462	142.2644

Table 7
GBAR model results over all applications of Dataset 2.

Dataset 2	I-Frames		P-Frames	
Application	RPE (%)	MAPE (%)	RPE (%)	MAPE (%)
	Lag-1			
Acrobat Reader			22.6685	21.4522
Microsoft Excel			17.4442	18.1185
Foxit Reader			30.1714	27.9252
Matlab			17.3298	18.0293
Microsoft Outlook			18.1343	17.8578
Microsoft PowerPoint			18.7265	20.3936
Ent. Device Manager			23.2407	28.7581
Snipping Tool			8.5492	10.1748
Microsoft Word			20.7295	18.8142
WinMerge			36.8734	53.8155
WinRAR			23.0900	21.4420
Xwin Cygwin			15.7283	14.0607
Windows Calculator			11.2782	12.3995
Google Chrome	3.4903	3.4018	18.5517	19.8051
Command Line			22.4217	23.3680
Mozilla Firefox			22.0732	20.2732
IrfanView			21.1866	22.1343
Internet Explorer			23.2647	23.8388
KDiff3			29.9645	26.3157
Windows Paint			18.6802	19.6575
Windows Notepad			22.7928	22.9972
Notepad++			19.4063	19.3469
Windows PowerShell	9.3244	8.8772	21.9069	26.8374
Windows Task Manger			18.9469	19.3739
VLC			20.6457	21.8365
VMware Player			33.3690	40.7958
Average error (%)	6.4074	6.1395	21.4298	22.6854

being a self-similar process with long-range dependence. Still, the autocorrelation values are not large enough for the GBAR model to benefit from them in terms of its accuracy.

5.3. Linear regression model

A Linear Regression (LR)-based model has been proposed in [32], for predicting the size of future B-Frames of MPEG-4 encoded video traffic. The model is based on the fact that B-Frames are constructed based on the reference frames, namely I and P-Frames or even based on previous B-Frames. As a consequence, the size of B-Frames may be strongly correlated with the size of their reference frames. The authors in [32] calculate the B-Frames correlation with their reference frames and the B-Frames autocorrelation, in order to locate the two most relevant frames per B-Frame in a GOP and by that to construct linear regression equations, which will be able to predict the B-Frames based on previous I, P or B-Frames.

Given the lack of accuracy of both the ADA and the GBAR model in predicting P-Frames sizes, we have followed a similar approach, in order to develop our own LR model, for predicting the sizes of future P-Frames of our video traffic in Dataset 1 and Dataset 2.

5.3.1. LR model analysis

The GOP size is 60 frames in Dataset 1 and we do not have a GOP structure in Dataset 2. Hence, in our LR model, we are based on previous I and P-Frames for Dataset 1 and on previous P-Frames for Dataset 2, in order to predict the P-Frames' sizes.

We initially calculated the correlation among the 59 P-Frames and the 1 I-Frame in a GOP for Dataset 1, the autocorrelation among the 59 P-Frames in a GOP for Dataset 1 and the autocorrelation among 60 P-Frames in Dataset 2. We define this number of P-Frames in Dataset 2 as a *Window*. The number of frames in the window is chosen in order to be equal to the number of frames of a GOP in Dataset 1, in order to make comparison between the results in the two Datasets, given the absence of a GOP structure in Dataset 2.

In general, let X denote the size of each P-Frame, Y denote the size of each I-Frame, σ_X denote the standard deviation of X and σ_Y the standard deviation of Y. Then the coefficient of correlation is calculated as

$$\rho_{X,Y} = \frac{E(XY) - \overline{XY}}{\sigma_X \sigma_Y} \tag{8}$$

and the autocorrelation is calculated as

$$r(k) = \frac{E\left[\left(X_m - \bar{X}\right)\left(X_{m+k} - \bar{X}\right)\right]}{\sigma_X^2} \tag{9}$$

where m denotes the present frame and k the lag.

We selected the two frames with the highest correlation for every P-Frame position in a GOP or a Window to construct the linear regression equations for predicting the P-Frames' sizes. Eq. (10) presents the equations' format:

$$F_P = a \cdot F_{P-1} + b \cdot F_{P-2} + c_P \tag{10}$$

where F_P denotes the size of the current P-Frame that we want to predict and F_{P-1} and F_{P-2} denote the size of the two previous frames with the highest correlation with P, which are used for the prediction. The a, b and c_P model parameters are estimated by employing the least squares error method. Parameter c_P is the disturbance term.

5.3.2. LR model evaluation results

We have applied the LR model to eight major applications of Dataset 1 and Dataset 2 (Microsoft Excel, Microsoft Word, Microsoft PowerPoint, Microsoft Outlook, Google Chrome, Mozilla Firefox, Internet Explorer and Matlab).

Figs. 2–4 present graphically the correlation values among the I and P frames for Dataset 1 and the autocorrelation values for Dataset 2. As shown in the figures, the two most relevant frames for frame N in Dataset 1 are the (N-1) and (N-2) P-Frames for

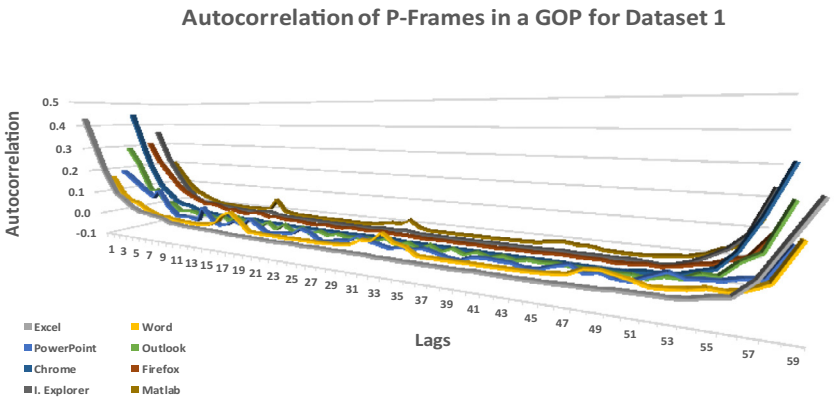

Fig. 2. Autocorrelation of P-Frames in a GOP for Dataset 1.

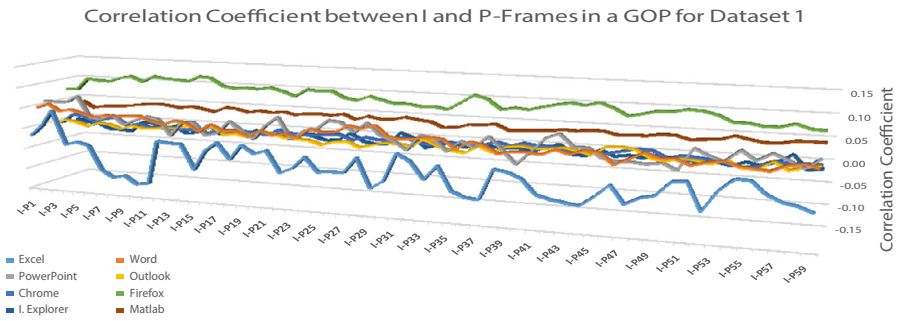

Fig. 3. Correlation Coefficient for I and P-Frames in a GOP for Dataset 1.

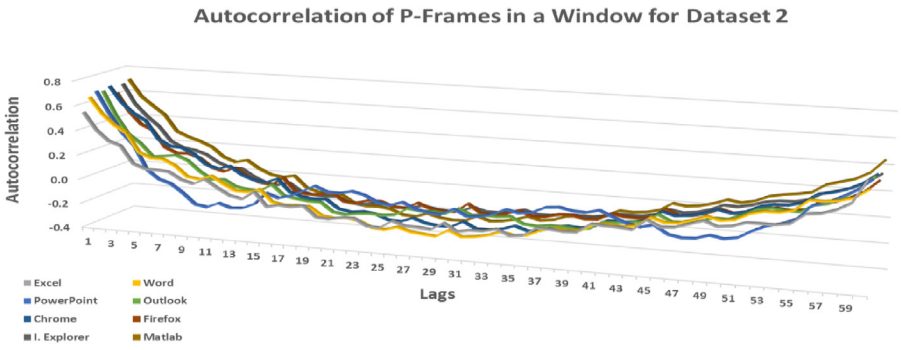

Fig. 4. Autocorrelation of P-Frames in a GOP for Dataset 2.

PowerPoint, Firefox and Matlab and the (N-1) and (N-59) for Excel, Word, Outlook, Chrome and Internet Explorer. As for Dataset 2, the most relevant frames, in terms of autocorrelation, are the (N-1) and (N-2) P-Frames for every major application with an exception for Excel, where the (N-1) and (N-60) P-Frames are the most relevant.

Having found the two "closest" frames for every P-Frame position in a GOP (for Dataset 1) or a Window (for Dataset 2), we can define the linear regression equations. They are presented in Eqs. (11) and (12) indicatively, for Dataset 2 (t denotes the current GOP).

Word, PowerPoint, Outlook, Chrome, Firefox, Internet Explorer and Matlab

$$\begin{aligned}
\widehat{P_{1,t}} &= a_1 \cdot P_{60,t-1} + b_1 \cdot P_{59,t-1} + c_1 \\
\widehat{P_{2,t}} &= a_2 \cdot P_{1,t} + b_2 \cdot P_{60,t-1} + c_2 \\
\widehat{P_{3,t}} &= a_3 \cdot P_{2,t} + b_3 \cdot P_{1,t} + c_3 \\
&\vdots \\
\widehat{P_{60,t}} &= a_{60} \cdot P_{59,t} + b_{60} \cdot P_{58,t} + c_{60}
\end{aligned} \tag{11}$$

Excel

$$\begin{aligned}
\widehat{P_{1,t}} &= a_1 \cdot P_{60,t-1} + b_1 \cdot P_{1,t-1} + c_1 \\
\widehat{P_{2,t}} &= a_2 \cdot P_{1,t} + b_2 \cdot P_{2,t-1} + c_2 \\
\widehat{P_{3,t}} &= a_3 \cdot P_{2,t} + b_3 \cdot P_{3,t-1} + c_3 \\
&\vdots \\
\widehat{P_{60,t}} &= a_{60} \cdot P_{59,t} + b_{60} \cdot P_{60,t-1} + c_{60}
\end{aligned} \tag{12}$$

Our results showed that the LR model fails to predict the P-Frames' sizes for both datasets. The average RPE and MAPE values for Dataset 1 were 91.05% and 45.84%, respectively. The RPE and MAPE values for Dataset 2 were 90.05% and 81.13%, respectively. The reason, once again, is the low correlation and autocorrelation values.

5.4. Markovian-Clustering models

In [36], a traffic model for layered video traffic is proposed. It is based on a Markovian arrival process and on a clusters' detection algorithm. Although our study is quite different since our traces'

Table 8
Optimal number of clusters for every tested application and for both datasets.

Application	# of clusters	
	Dataset 1	Dataset 2
Microsoft Excel	11	4
Microsoft Word	11	7
Microsoft PowerPoint	7	4
Microsoft Outlook	7	4
Google Chrome	11	4
Mozilla Firefox	11	7
Internet Explorer	7	7
Matlab	11	4

traffic is not layered, we decided to use a conceptually similar approach with [36]. We developed a Markovian-Clustering (MC) model, in order to predict the sizes of I-Frames and P-Frames from Dataset 1 and Dataset 2. We have applied the MC model to the same eight major applications of Dataset 1 and Dataset 2 that we applied the LR model.

In our proposed approach, we view the video trace sequence as a vector containing all the I-Frames or P-Frames' sizes (depending on which type we wish to model). We place all the vector's elements as points on the 1-D plane and we then use the K-Means clustering algorithm [28] in order to cluster similar-sized frames. We selected the K-Means algorithm due to the fact that the volume of the data we wanted to cluster was very large (for example, Dataset 2 contains over $5{\bullet}10^6$ P-Frames) and K-Means deals better with large datasets than other clustering algorithms (e.g., the Hierarchical clustering algorithm). The distance metric that we used for clustering is the cityblock Distance, which calculates the sum of absolute differences (i.e., the L_1 distance). Even though K-Means is a powerful clustering algorithm, it has a significant drawback. The K amount of clusters has to be selected heuristically. The Elbow method [39] is a well-known approach that can be used to determine the proper value of k. In our case, we concluded after several experiments that the optimal numbers of clusters per tested application (i.e., the number of clusters that leads to the highest modeling accuracy) are the ones depicted in Table 8.

Next, we constructed a Markov chain based on the above clustering results. Each cluster corresponds to one state of the Markov chain. We computed the Transition Probability Matrix $T = [P_{i,j}]^2$ for the Markov chain, which contains $K \times K$ elements, following Eq. (13).

$$P_{i,j} = \frac{\#\text{ of jumps from state } i \text{ to state } j}{\#\text{ of jumps from state } i} \tag{13}$$

Finally, we found the best distribution fit for the data in each cluster.

5.4.1. Jaccard index-infused MC model

The MC model performs separate clustering on 1-D data for I-frames and P-Frames based on the actual size of every frame type. Although its results, which will be presented in Section 5.4.2, were quite satisfactory, we tried another approach to further improve the results of P-Frame modeling by moving from the 1-D to the 3-D plane. To do so, we employed, for the first time in the relevant video traffic modeling literature to the best of our knowledge, the concept of the Jaccard Index.

The Jaccard Index [21], also known as the Jaccard similarity coefficient, is a statistic used for comparing the similarity and diversity of two sample sets. The Jaccard coefficient measures similarity between finite sample sets and is defined in general as the size of the intersection divided by the size of the union of two sample sets, as depicted in Eq. (14) below

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \tag{14}$$

where A and B denote the two sample sets. Jaccard Index is widely used in regionalization and species association analyses [26].

In our case, we wanted to find for every P-Frame in a GOP (for Dataset 1) or in a Window (for Dataset 2), the two "closest" P-Frames, with an approach different from a simple autocorrelation calculation (as it was shown to perform poorly when used in the GBAR and LR models). This approach is the use of the Jaccard Index in the following way.

For every P-Frame, denoted as X, in a GOP or a Window, we calculate its Jaccard Index with every other P-Frame, denoted as Y, in the same GOP or Window. The sample sets A and B in this Jaccard Index calculation are the "neighboring frames" of X and the "neighboring frames" of Y respectively. As a "neighboring frame" P^* of a P-Frame P, we define every P-Frame that satisfies the following two rules:

1. The absolute difference between the sizes of P and P^* does not exceed the standard deviation of P-Frames' sizes.
2. The arrival of P^* does not change the autocorrelation (lag-1) of P-Frames in the trace, more than 10% compared to the change that occurred from the arrival of P.

Via this definition, we found that the two "closest neighbors" of each P-Frame are the previous and the following one, for all eight major applications of both datasets. Fig. 5 presents this result graphically for Dataset 2. Note that these two "closest neighbors" are different than those depicted by the autocorrelation values and used in the LR model.

The x-axis and the y-axis in Fig. 5 represent the P-Frame position in a Window for Dataset 2, and the z-axis represents the Jaccard Index value between two P-Frames defined by x and y. As shown in the figure, the Jaccard Index has a value equal to 1 on the $x = y$ line (because every frame has the same Jaccard Index with itself) and gets smaller, as the distance from $x = y$ line grows.

We then view the video trace sequence as a vector $\langle R_p(t), R_c(t), R_n(t)\rangle$, $t = 2, 3, 4, \ldots$ Here $R_c(t)$ denotes the frame size of the tth P-Frame, R_p(t) denotes the frame size of the P-Frame before $R_c(t)$ (i.e., $R_p(t) = R_c(t-1)$) and $R_n(t)$ denotes the frame size of the P-Frame after $R_c(t)$ (i.e., $R_n(t) = R_c(t+1)$). We place all the $\langle R_p(t), R_c(t), R_n(t)\rangle$ pairs as points on the 3-D plane, where $R_p(t)$, $R_c(t)$ and $R_n(t)$ is viewed as the x-coordinate, y-coordinate and z-coordinate of the corresponding point respectively. Hence, each P-Frame is clustered by taking into account not only its own size but also the size of its adjacent frames.

As in the MC model, we find the best distribution fit for the data in each cluster. We name this new model Jaccard Index-Infused MC Model (JIMC) and we evaluate it in the next section.

Finally, a direct outcome of the MC and JIMC models is an algorithmic procedure for source video traffic generation (generating I and P video frame sizes for a desired application). This algorithmic procedure can be summarized, for the MC model, as follows:

- *Step 1:* Based on a training video trace of the desired application, acquire an a priori knowledge for the number of K-Means clusters, the Transition Probability Matrix T of the corresponding Markovian chain and the best distribution fit for the data in each cluster, using the MC modelling procedure presented in 5.4.
- *Step 2:* Generate the first frame size, according to the distribution fit that characterizes the cluster where the first frame from the training video trace belongs.
- *Step 3:* Generate a random number, in order to transit to the next cluster, based on the Transition Probability Matrix T.

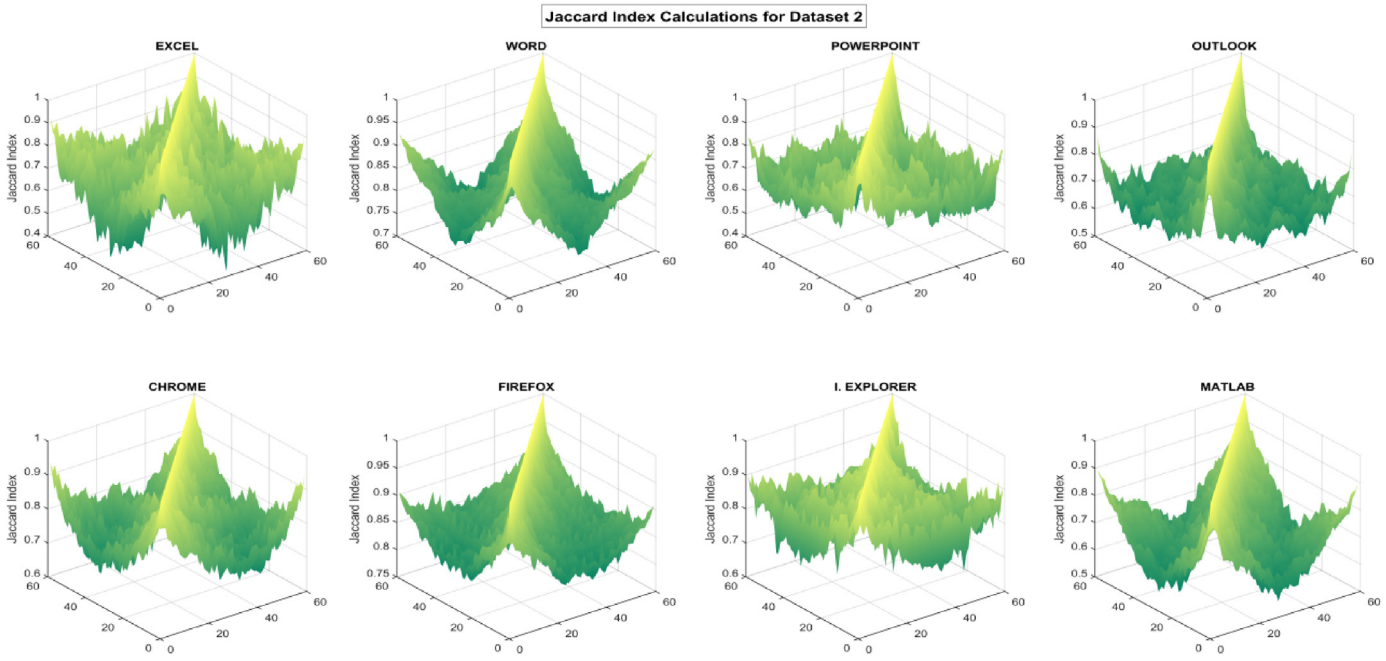

Fig. 5. Jaccard Index calculations for all major applications of Dataset 2.

Table 9
MC model results for I-Frames for the major applications of Dataset 1.

Application	Dataset 1 (I-Frames)	
	RPE (%)	MAPE (%)
Microsoft Excel	5.7373 ± 1.9091	7.0698 ± 2.1869
Microsoft Word	3.2203 ± 0.8452	2.7560 ± 0.4853
Microsoft PowerPoint	3.2565 ± 0.5409	3.9140 ± 0.5829
Microsoft Outlook	1.9270 ± 0.2369	2.0751 ± 0.2764
Mozilla Firefox	2.5907 ± 0.5557	2.6731 ± 0.5507
Google Chrome	1.6458 ± 0.2805	1.9540 ± 0.3414
Internet Explorer	2.0524 ± 0.3525	2.4550 ± 0.4463
Matlab	0.9867 ± 0.1635	1.0735 ± 0.1742
Average Error (%)	2.6771	2.9963

- *Step 4:* Generate a frame size according to the distribution fit that characterizes the new cluster.
- *Step 5:* Repeat Step 3 and 4, as many times as the number of frames that need to be generated.

The generative algorithmic procedure that related to the JIMC Model is the same with the aforementioned, with the only difference being in Step 1, where the modelling procedure is conducted according to the JIMC model instead of the MC one.

5.4.2. MC and JIMC Models' evaluation results

Before presenting our results, we should mention that the usage of the K-Means algorithm (with random selection of the initial centroids) and the usage of a random number generator (in order to change clusters according to the Markov chain's transition probabilities) naturally lead to some fluctuations in the results. For this reason, all of our results were derived for 95% confidence intervals, constructed the usual way [27].

Tables 9 and 10 present the results of the MC model for Dataset 1 and Dataset 2, in terms of the model's accuracy in predicting I-Frames and P-Frames' sizes. No results were obtained for the I-Frames of Dataset 2, due to the encoding, which as explained earlier generates too few I-Frames.

As shown from the results, the MC model succeeds in predicting the I-Frames and P-Frames' sizes for both datasets with high accuracy. The RPE and MAPE errors are below 5% for all applications, with an exception for the I-Frames of Microsoft Excel and the P-Frames of Internet Explorer of Dataset 1, where the errors are slightly higher.

Table 11 presents the results of the JIMC model for Dataset 1 and Dataset 2, in terms of the model's accuracy in predicting P-Frames' sizes.

Our JIMC model is shown to succeed in predicting the P-Frames' sizes for both datasets with high accuracy. We should mention that we have experimented with different values in the 2nd rule of the definition of a "neighboring frame"; we have used values of up to 20% difference in autocorrelation, with negligible differences in the results. The two "closest neighbors" to a P-Frame remained its previous and next one.

In comparison with the MC model, JIMC is clearly better for the Miracast-like dataset (much lower MAPE, lower RPE) but underperforms for the 1st dataset. The reason is that in Dataset 2 the "ties" among P-Frames (size similarities, similar changes in autocorrelation, higher Hurst parameter) are stronger than in Dataset1. We also need to add that, to avoid overfitting in our models' evaluation, we have also conducted two more rounds of statistical tests. In the first we used 80% of the data as a training set and the other 20% as a test set, and in the second we used 50% of the data as a training set and the other 50% as a test set (for each application, as every application is modeled separately). The training and validation subsets are equally stratified for each type of dataset, as the two types of datasets have a different video frame structure. In both cases there were no qualitative changes in the models' evaluation results, and the quantitative changes were negligible (the differences in the results did not exceed 0.5%).

Fig. 6 presents the Q–Q Plots of the real and predicted I- and P-Frames, for two major applications (Matlab and Google Chrome) of Dataset 1 and Dataset 2, over all models. The Q–Q Plots confirm the high accuracy of both our proposed models (MC and JIMC) for the I-Frames and P-Frames of the two datasets.

Fig. 7 contains the summarized results, in terms of average errors, over eight major applications of both datasets, for ease of comparison of all models used in our work. It is clear again that the MC and JIMC models vastly outperform the other models in

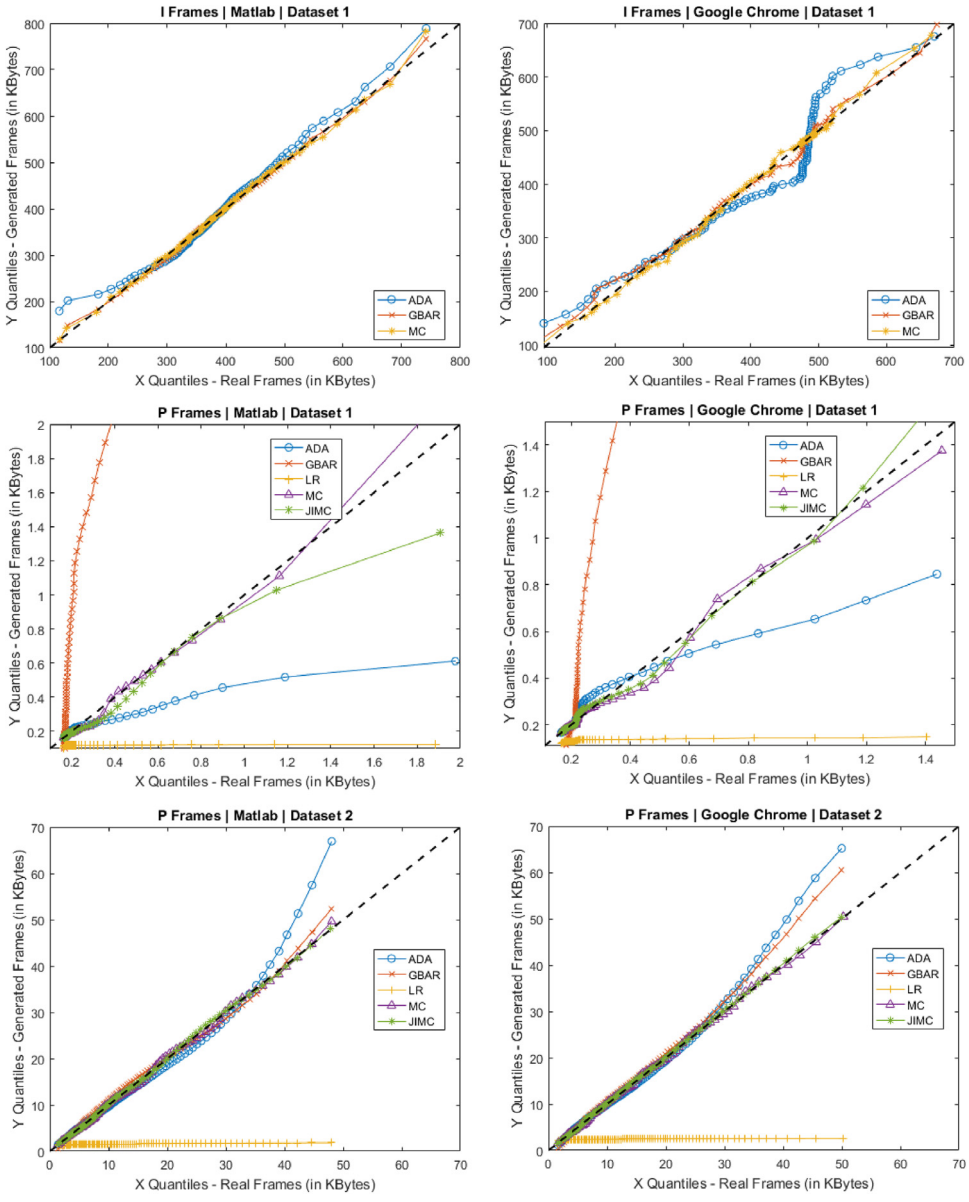

Fig. 6. Q–Q plots for I- and P-Frames of two major applications, using all models.

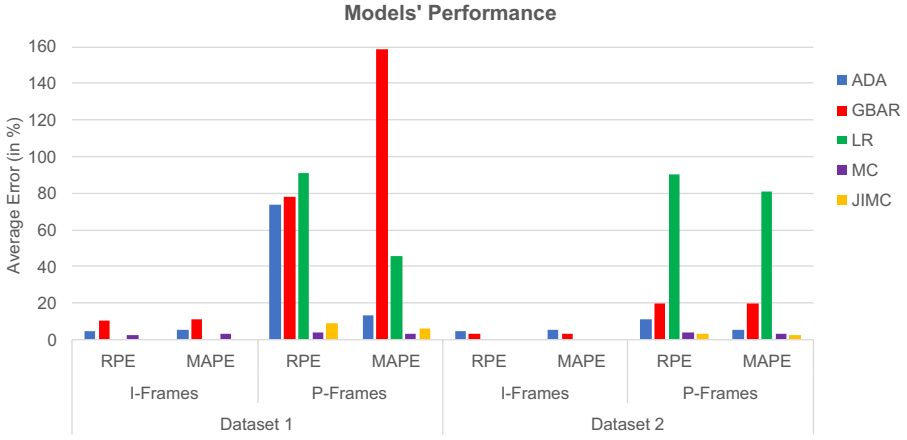

Fig. 7. Performance comparison in terms of average errors, for I-Frames and P-Frames for the major applications of both datasets, over all models.

Table 10
MC model results for P-Frames for the major applications ofboth datasets.

Application	Dataset 1 (P-Frames)		Dataset 2 (P-Frames)	
	RPE (%)	MAPE (%)	RPE (%)	MAPE (%)
Microsoft Excel	6.3818 ± 1.9190	6.2201 ± 0.1014	3.9994 ± 0.0641	5.2104 ± 0.0172
Microsoft Word	2.5862 ± 1.5973	2.7232 ± 0.1565	2.8518 ± 0.1129	2.3135 ± 0.0239
Microsoft PowerPoint	3.3388 ± 1.1632	3.4381 ± 0.2899	6.2012 ± 0.9905	2.6965 ± 0.0775
Microsoft Outlook	3.5491 ± 0.3730	2.1716 ± 0.0388	4.4974 ± 0.0349	3.4060 ± 0.0345
Mozilla Firefox	3.5119 ± 0.5430	2.5992 ± 0.1273	2.1300 ± 0.0444	2.1410 ± 0.0696
Google Chrome	3.9998 ± 0.4944	3.1880 ± 0.0287	3.0148 ± 0.0405	5.0947 ± 0.0237
Internet Explorer	5.0114 ± 0.7888	2.3996 ± 0.1427	3.7670 ± 0.2925	3.4289 ± 0.1084
Matlab	4.4341 ± 0.3912	2.4139 ± 0.0143	4.4852 ± 0.0150	3.1207 ± 0.0079
Average error (%)	4.1016	3.1442	3.8684	3.4265

Table 11
JIMC model results for major applications of both datasets.

Application	Dataset 1 (P-Frames)		Dataset 2 (P-Frames)	
	RPE (%)	MAPE (%)	RPE (%)	MAPE (%)
Microsoft Excel	6.9557 ± 0.8705	6.2363 ± 0.2823	3.7547 ± 0.1226	4.3616 ± 0.0928
Microsoft Word	8.3352 ± 1.0773	5.2356 ± 0.1155	2.4896 ± 0.1164	1.9312 ± 0.0749
Microsoft PowerPoint	10.7993 ± 1.6523	5.5227 ± 0.1607	3.7142 ± 0.0972	1.9619 ± 0.0945
Microsoft Outlook	8.5187 ± 0.9039	7.4242 ± 0.0952	4.2455 ± 0.0124	2.2490 ± 0.0140
Mozilla Firefox	8.2838 ± 0.5629	4.0421 ± 0.1568	2.5819 ± 0.2911	1.6661 ± 0.0627
Google Chrome	9.7563 ± 0.7707	8.2017 ± 0.0936	2.8272 ± 0.3704	1.3243 ± 0.0887
Internet Explorer	9.8408 ± 2.1705	7.1371 ± 0.2832	2.6597 ± 0.2645	2.3599 ± 0.1936
Matlab	8.3031 ± 0.2889	4.0608 ± 0.0902	3.2720 ± 0.0103	1.9951 ± 0.0126
Average error (%)	8.8491	5.9826	3.1931	2.2311

terms of accuracy in predicting the I-Frames and P-Frames' sizes of both datasets.

5.4.3. Applicability of the model to real-time video traffic prediction

Finally, we should note the following regarding our model's applicability to real-time video traffic prediction in an online environment. The problem of real-time prediction has been addressed several times in the relevant literature. One solution is that the modeling is done off-line, on a training dataset, and the modeling results are then applied online. This minimizes complexity, however it fails to capture the dynamic nature of newly generated traffic and blindly assumes that the offline model will be accurate on the test dataset as well, and will remain accurate. Models that have been proposed for real-time modeling of video traffic include [49–51]. In [49] the authors acknowledge the problem of the computational overhead of their scheme (which focuses on multiplexed videos) and they use a combination of exact and approximation schemes to contain computational overhead. In [50,51] the authors propose a simplified seasonal ARIMA model and a neural network model, respectively, for real-time video traffic prediction; in both cases, the authors argue that their schemes have relatively low complexity, hence they can be implemented in real-time. Similarly, we could argue for our work that after the offline estimations are made (based on the training dataset) for the split into clusters and the best distribution fit is found for each cluster, it is relatively simple to use the proposed model for real-time prediction because with newly arriving traffic it only needs to run the k-means algorithm and to recompute the transition probability matrix. Still, in the case when the newly generated frames of the test dataset are very different in sizes and correlation compared to those of the training dataset, the results for the training dataset for any of the aforementioned models (including ours) may not be able to be extrapolated to the test dataset; for example, in our model the best fit distribution within clusters might change. Hence, in our view the proposed JIMC model can be used offline to produce initially accurate estimates based on existing data, and when new screen mirroring applications are initialized online, the model can run in real-time with the understanding that only through video frames' buffering the model will practically keep up with the dynamically changing video traffic. This, however, is not a major issue if we take into account the fact that the reason for the modeling is its use in network traffic control; the very high accuracy of the proposed model ensures that even if call admission control/traffic policing decisions are made with a lag as high as a few seconds, the addition of one or few new users into a well-predicted network load will not cause significant discrepancies in the traffic control decision.

6. Conclusions and future work

This work addressed, for the first time in the relevant literature to the best of our knowledge, the problem of modeling real user-generated screen mirroring traffic. We have tested four modeling techniques for predicting the size of video traffic that is generated by an average user's computer during a day. We have worked with two different datasets of H.264 encoded video traffic traces, one encoded with the High 4:2:2 Profile of H.264 standard and the other encoded with parameters, which resemble a Miracast hardware encoder, since Miracast is a widely accepted screen mirroring standard.

We have shown that approaches such as the Gamma Beta Autoregression model and Linear Regression model, which have provided accurate models in the past for other video encoding schemes, fail to accurately predict video traffic, due to the autocorrelation characteristics of the type of traffic that we worked with.

We proposed the Markovian-Clustering Model for I-Frames and P-Frames' size prediction, which we modified by incorporating the Jaccard Index into the model. We have shown that the Markovian-Clustering model has excellent accuracy for I-Frames' size prediction, as well as for P-Frames' sizes prediction for the 1st Dataset. We have also shown that the Jaccard Index -Infused MC model has even higher accuracy in P-Frames sizes' prediction for the 2nd Dataset (i.e., for prediction of Miracast-like encoded video traffic).

Given that smart mobile devices are strong candidates to replace computers in corporate environments, we believe that our work provides a solid basis for future studies on modeling video

traffic generated by real computer usage behavior. In the future, we intend to evaluate our models on a wider variety of corporate and daily computer applications and use our models in wireless resource allocation problems. Most importantly, we intend to use in our future work the very recently introduced Screen Content Coding (SCC) extension of the HEVC encoding scheme, and model the respective data that will be collected in order to compare the results against the present work.

References

[1] D. Gilbert, Microsoft Wants to Replace Your PC With Your Smartphone, International Business Times, 30 Apr. 2015 [Online]:http://www.ibtimes.co.uk/microsoft-wants-replace-your-pc-your-smartphone-1499042.

[2] Naked Security, INFOGRAPHIC: Users Weighed Down by Multiple Gadgets – Survey Reveals the Most Carried Devices, Sophos, 14 Mar. 2013 [Online]:https://nakedsecurity.sophos.com/2013/03/14/devices-wozniak-infographic/.

[3] Wi-Fi Alliance, Discover Wi-Fi: Wi-Fi CERTIFIED Miracast™, 2012. [Online] http://www.wi-fi.org/discover-wi-fi/wi-fi-certified-miracast.

[4] A. Lazaris, P. Koutsakis, M. Paterakis, A new model for video traffic originating from multiplexed MPEG-4 videoconference streams, Perform. Eval. 65 (1) (2008) 51–70.

[5] M. Nomura, T. Fuji, N. Ohta, Basic characteristics of variable rate video coding in ATM environment, IEEE J. Sel. Areas Commun. 7 (5) (1989) 752–760.

[6] D.M. Lucantoni, M.F. Neuts, A.R. Reibman, Methods for performance evaluation of VBR video traffic models, IEEE/ACM Trans. Netw. 2 (1994) 176–180.

[7] D.P. Heyman, A. Tabatabai, T.V. Lakshman, Statistical analysis and simulation study of video teleconference traffic in ATM networks, IEEE Trans. Circuits Syst. Video Technol. 2 (1) (1992) 49–59.

[8] A.M. Dawood, M. Ghanbari, Content-based MPEG video traffic modeling, IEEE Trans. Multimed. 1 (1) (1999) 77–87.

[9] B. Melamed, D.E. Pendarakis, Modeling full-length VBR video using Markov-renewal modulated TES models, IEEE J. Sel. Areas Commun. 16 (5) (1998) 600–611.

[10] H. Zhu, A. Matrawy, I. Lambadaris, Models and tools for simulation of video transmission on wireless networks, in: Proceedings of the *Canadian Conference on Electrical and Computer Engineering*, 2004, pp. 781–784.

[11] K. Chandra, A.R. Reibman, Modeling one- and two-layer variable bit rate video, IEEE/ACM Trans. Netw. 7 (3) (1999) 398–413.

[12] Q. Ren, H. Kobayashi, Diffusion approximation modeling for Markov modulated bursty traffic and its applications to bandwidth allocation in ATM networks, IEEE J. Sel. Areas Commun. 16 (5) (1998) 679–691.

[13] D.P. Heyman, The GBAR Source Model for VBR Videoconferences, IEEE/ACM Trans. Netw. 4 (4) (1997) 554–560.

[14] M. Frey, S. Ngyuyen-Quang, A gamma-based framework for modeling variable-rate video sources: the GOP GBAR model, IEEE/ACM Trans. Netw. 8 (6) (2000) 710–719.

[15] A.K. Al Tamimi, C. So-In, R. Jain, Modeling and resource allocation for mobile video over WiMAX broadband wireless networks, IEEE J. Sel. Areas Commun. 28 (3) (2010) 354–365.

[16] C.H. Liew, C. Kodikara, A. Kondoz, Video traffic model for MPEG4 encoded video, in: Proceedings of the *62nd IEEE Vehicular Technology Conference-Fall*, 2005, pp. 1854–1858.

[17] S. Tanwir, H. Perros, A survey of VBR video traffic models, IEEE Commun. Surv. Tutor. 15 (4) (2013) 1778–1802.

[18] J.E. Dunn, Microsoft Office applications Barely Used by Many employees, New Study Shows, Techworld, 01 May 2014 [Online]. Available: http://www.techworld.com/news/security/microsoft-office-applications-barely-used-by-many-employees-new-study-shows-3514565/.

[19] D. Marpe, T. Wiegand, G. Sullivan, The H.264/MPEG4 advanced video coding standard and its applications, IEEE Commun. Mag. 44 (8) (2006) 134–143.

[20] M. Dai, Y. Zhang, D. Loguinov, A unified traffic model for MPEG-4 and H.264 video traces, IEEE Trans. Multimed. 11 (5) (2009) 1010–1023.

[21] P. Jaccard, Nouvelles Recherches Sur la Distribution Florale, Bull. Soc. Vaud. Sci. Nat. 44 (1908) 223–270.

[22] "Trace files and statistics: H.264/AVC video trace library," [Online]. Available: http://trace.eas.asu.edu/h264/. Accessed May 10, 2017 .

[23] FFmpeg Organization, About FFmpeg, 20 Dec. 2000. [Online]. Available: https://www.ffmpeg.org/about.html.

[24] Microsoft Corporation, Scripting With Windows Powershell, Microsoft TechNet Library, 4 Aug. 2014 [Online]. Available: https://technet.microsoft.com/en-us/library/bb978526.aspx.

[25] Techex, "X264," [Online]. Available: http://www.techex.co.uk/codecs/x264. Accessed May 10, 2017.

[26] R. Real, Tables of significant values of Jaccard's index of similarity, Misc. Zool. 22 (1) (1999) 29–40.

[27] A.M. Law, W.D. Kelton, Simulation Modeling & Analysis, 2nd ed., McGraw-Hill, 1991.

[28] J.A. Hartigan, M.A. Wong, A K-means clustering algorithm, J. R. Stat. Soc.. Ser. C (Appl. Stat.) 28 (1) (1979) 100–108.

[29] F.J. Massey, The Kolmogorov–Smirnov Test for Goodness of Fit, J. Am. Stat. Assoc. 46 (1951) 68–78.

[30] T.W. Anderson, D.A. Darling, Asymptotic theory of certain "Goodness of Fit" criteria based on stochastic processes, Ann. Math. Stat. 23 (2) (1952) 193–212.

[31] C. Tofallis, A better measure of relative prediction accuracy for model selection and model estimation, J. Oper. Res. Soc. 66 (2015) 1352–1362.

[32] L.I. Lanfranchi, B.K. Bing, MPEG-4 bandwidth prediction for broadband cable networks, IEEE Trans. Broadcast. 54 (4) (2008) 741–751.

[33] D.P. Heyman, T.V. Lakshman, What are the implications of long-range dependence for VBR-video traffic engineering, IEEE/ACM Trans. Netw. 4 (3) (1996) 301–317.

[34] B.K. Ryu, A. Elwalid, The importance of long-range dependence of VBR video traffic in ATM traffic engineering: myths and realities, in: Proceedings of the *ACM SIGCOMM*, 1996, pp. 3–14.

[35] J. Beran, et al., Long-range dependence in variable bit-rate video traffic, IEEE Trans. Commun. 43 (2–4) (1995) 1566–1579.

[36] J.-A. Zhao, B. Li, I. Ahmad, Traffic model for layered video: An approach on Markovian arrival process, in: Proceedings of the *International Conference on Multimedia and Expo*, 2003.

[37] O. Rose, Estimation of the Hurst parameter of long-range dependent time series, Feb. 1996. Technical report[Online]: https://www.eecis.udel.edu/~mills/fractal/tr137.pdf.

[38] MHL Consortium. [Online]: http://www.mhltech.org/.

[39] R.L. Thorndike, Who belongs in the family? Psychometrica 18 (4) (1953) 267–276.

[40] A.M. Kholaif, et al., Energy efficient H.263 video transmission in power saving wireless LAN infrastructure, IEEE Trans. Multimed. 12 (2) (2010) 142–153.

[41] X. Qi, et al., A context-aware framework for reducing bandwidth usage of mobile video chats, IEEE Trans. Multimed. 18 (8) (2016) 1640–1649.

[42] K. Miller, et al., A control-theoretic approach to adaptive video streaming in dense wireless networks, IEEE Trans. Multimed. 17 (8) (2015) 1309–1322.

[43] Q.M. Qadir, et al., A novel traffic rate measurement algorithm for quality of experience-aware video admission control, IEEE Trans. Multimed. 17 (5) (2015) 711–722.

[44] M. Kastrinakis, G. Badawy, M.N. Smadi, P. Koutsakis, [Online]: http://users.isc.tuc.gr/~makastrinakis/docs/Screen_Mirroring_Video_Trace_Datasets.zip.

[45] A. Pulipaka, P. Seeling, M. Reisslein, Traffic models for H.264 video using hierarchical prediction structures, in: Proceedings of the *IEEE GLOBECOM*, 2012.

[46] H. Kalbkhani, M.G. Shayesteh, N. Haghighat, Adaptive LSTAR model for long-range variable bit rate video traffic prediction, IEEE Trans. Multimed. 19 (5) (2017) 999–1014.

[47] D.R. Markovic, A.M. Gavrovska, I.S. Reljin, 4 K video traffic analysis using seasonal autoregressive model for traffic prediction, in: Proceedings of the *24th Telecommunications Forum (TELFOR)*, 2016.

[48] S. Tanwir, D Nayak, H. Perros, Modeling 3D video traffic using a Markov modulated gamma process, in: Proceedings of the *International Conference on Computing, Networking and Communications (ICNC)*, 2016.

[49] D. Sarkar, U.K. Sarkar, W. Zhou, Bandwidth estimation for multiplexed videos using multinomial model, Comput. Commun. 30 (2) (2007) 269–279.

[50] A.K. Al Tamimi, R. Jain, C. So-In, Dynamic resource allocation based on online traffic prediction for video streams, in: Proceedings of the *4th IEEE International Conference on Internet Multimedia Services Architecture and Application (IMSAA)*, 2010.

[51] A.D. Doulamis, N.D. Doulamis, S.D. Kollias, An adaptable neural-network model for recursive nonlinear traffic prediction and modeling of MPEG video sources, IEEE Trans. Neural Netw. 14 (1) (2003) 150–166.