TECHNICAL UNIVERSITY OF CRETE
SCHOOL OF ELECTRONIC AND COMPUTER ENGINEERING
TELECOMMUNICATIONS DIVISION
INFORMATION PROCESSING AND NETWORKS LABORATORY

Traffic Modeling based on Real User-Generated Data from a Wireless Service Provider

by

Marios Kastrinakis

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DIPLOMA OF ELECTRONIC AND COMPUTER ENGINEERING

Chania, December 2015

THESIS COMMITTEE
Associate Professor Polychronis Koutsakis, *Thesis Advisor*
Professor Michael Paterakis
Associate Professor Antonios Deligiannakis

Acknowledgements

I would like to thank heartfully a lot of people for helping me not only to complete my thesis but most important for their support throughout my studies.

First of all, I would like to thank my family and mostly my parents Antonis and Alexandra that supported me and helped me succeed in my goals and my studies all those five wonderful years.

I would like to thank wholeheartedly my thesis advisor Polychronis Koutsakis, who has offered me valuable knowledge on the subject of Computer Networks during my studies and most important for his trust, excellent guidance, support and cooperation from the beginning till the end of my thesis.

In addition, I would like to thank Professor Michael Paterakis and Associate Professor Antonios Deligiannakis for their participation in the examination committee as well as Dr. Mohammed Smadi and Dr. Ghada Badawy for collecting and providing our working data.

Last but not least, I would like to thank my dear friends who are close to me all those years and Ioanna Choudalaki for her love and emotional support.

To my family.

Abstract

Nowadays, smart mobile devices tend to replace personal computers, such as desktops and laptops, in daily applications such as internet browsing, emailing, office and basic corporate applications. On the other hand, users still want to use a monitor, keyboard and mouse, as usual, when they are not on the move and the most common way to do that is by using a wireless docking station over a Wi-Fi network. High quality video transmission (for example the desktop view of a smart device) over Wi-Fi networks on heavily loaded environments has been proven problematic in terms of Quality of Service (QoS) and fair bandwidth allocation between users. In this thesis, we developed and tested four different modeling techniques for predicting the volume of video traffic that is generated by an average user's computer during a day. 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 to predict the video frames' sizes of the specific type of video traffic. Our models can be easily used as source traffic generators in order to facilitate the study of H.264 transmission performance over wireless networks.

Table of Contents

List of Abbreviations

AD: Anderson-Darling
ADA: Application and Distribution Aware
AR: Autoregressive
AVC: Audio Video Converter
B-Frame: Bi-directional predicted Frame
CDF: Cumulative Distribution Function
CI: Confidence Interval
DAR: Discrete Autoregressive
DTV: Digital Television
GBAR: Gamma Beta Autoregressive
GEV: Generalized Extreme Value
GoF: Goodness of Fit
GPU: Graphics Processing Unit
FPS: Frames Per Second
HQ: High Quality
IEC: International Electrotechnical Commission
I-Frame: Intra-coded Frame
IID: Independent and Identically Distributed
ISO: International Organization of Standardization
IPTV: Internet Protocol Television
ITU: International Telecommunication Union
JIMC: Jaccard Index -Infused Markovian - Clustering
KS: Kolmogorov-Smirnov
LRD: Long Range Dependence
LR: Linear Regression
MAPE: Mean Absolute Percentage Error
MC: Markovian - Clustering
MLE: Maximum Likelihood Estimation
MPEG: Moving Pictures Experts Group
MRP: Markov Renewal Process
PC: Personal Computer
P-Frame: Predicted Frame
QoS: Quality of Service
Q-Q: Quantiles-Quantiles
RPE: Relative Percentage Error
RTP: Real Time Protocol
SRD: Short Range Dependence
TES: Transform Expand Sample
VBR: Variable Bitrate
Wi-Fi: Wireless Fidelity
VCEG: Video Coding Experts Group

List of Figures

List of Tables

1 Introduction and Related Work

Smart mobile devices 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 (such as smartphones and tablets) will replace computers completely, on daily and corporate environments.

On the other hand, users still want to use a monitor, keyboard and mouse as usual, when they are not on the move and the most common way to do that is by using a wireless docking station over a Wi-Fi network. Screen mirroring is a very popular and demanding wireless application. People's demand for enjoying any content anywhere, anytime and on any device is driving the need for reliable connectivity between content source and sink devices in their home, car and office. This fact, together with the advent of wireless communications is changing the enterprise environment where some offices are now looking into using the hand held devices as PC replacements. To provide the same experience to the employee using a hand held device, the device has to be connected to a Bluetooth keyboard and mouse and the device's screen has to be mirrored to a bigger display. Screen mirroring is made possible by using a Wi-Fi direct technology called Miracast. Wi-Fi Direct is a technology defined by the Wi-Fi Alliance that is used in ad-hoc networks to connect devices directly without the need of an overlaid network. Miracast [3] allows devices to send video and audio files securely over a Wi-Fi direct link. Miracast allows the video to be encoded using H.264 which is one of the most popular video encoding standards that is currently used for video recording, compression, and streaming. H.264 is popular because it provides a good quality for lower bit rates than previous standards.

This use case adds extra challenges to an already hard problem. One of these challenges is that the wireless connection between the video source and sink has to be reliable for a long time (around 8 hours) and in a very dense environment (one pair every 2 meters). Moreover, the required wireless bandwidth and acceptable delay for these connections vary greatly as the applications that the employees use vary and due to the fact that Wi-Fi was initially designed as a best effort, listen-before-talk technology intended for low utilization networks. Hence, Quality of Service (QoS) can be degraded significantly even for a small number of users and outages can be observed during video transmission. A highly accurate model capable to predict the volume of video traffic generated by an average user's computer, can be very helpful in dealing with congestion and providing fair bandwidth allocation between users in Wi-Fi networks.

In this thesis, we develop such a model for the first time in literature (to the best of our knowledge). Our models can be easily used as source traffic generators in order to facilitate the study of H.264 transmission performance over wireless networks. Our work is structured in six chapters. The first chapter includes this introduction, the second chapter refers to the video traffic encoding of the data that we worked with and the third chapter describes the methods that we have followed in order to collect our data as well as some extra information about our datasets' structure. In the fourth chapter, we present the statistical tests that we have used during the development and testing of our models and in the fifth chapter, we analyze and comment on our models as well as on their respective results. In the sixth and final chapter, our conclusions and ideas for future work can be found.

1.1 Related Work

There are multiple video traffic models in the literature but they all base their models on movies which are very different in nature than desktop applications. According to [4], video models which have been proposed include first order autoregressive (AR) models [5], discrete autoregressive (DAR) models [6] [7], Markov renewal processes (MRP) [8], MRP transform expand sample (TES) [9] [10], finite state Markov chain [11] [12], 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 traffic model for H.264 encoded video that takes interdependence between different frame types into consideration (I, P and B) and again the model was validated against "Lord of the Rings" movie. 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 the others. They could not find a universal model that works with all types of videos.

The above-reference work models video traces that are significantly different, in terms of content than those created by 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 what other applications do the employees use when they are not using Microsoft Office. Also there is no characterization of the video generated by the Microsoft Office suite or similar productivity tools.

2 Video Encoding

In this thesis, we worked with two different datasets. They have been 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].

A video trace is a sequence of still pictures displayed within short time intervals, in order to create the illusion of moving scene. Each distinct picture is named as a "frame" and the number of displayed frames per second represents the frame rate of the video trace and is calculated as in Equation (2.1).

$$\text{Frame Rate} = \frac{\text{x Frames}}{\text{1 Second}} = \text{x fps} \tag{2.1}$$

A graphical example for two different video traces, one with 6 fps (Trace A) and one with 24 fps (Trace B) is depicted in Figure (2.1).

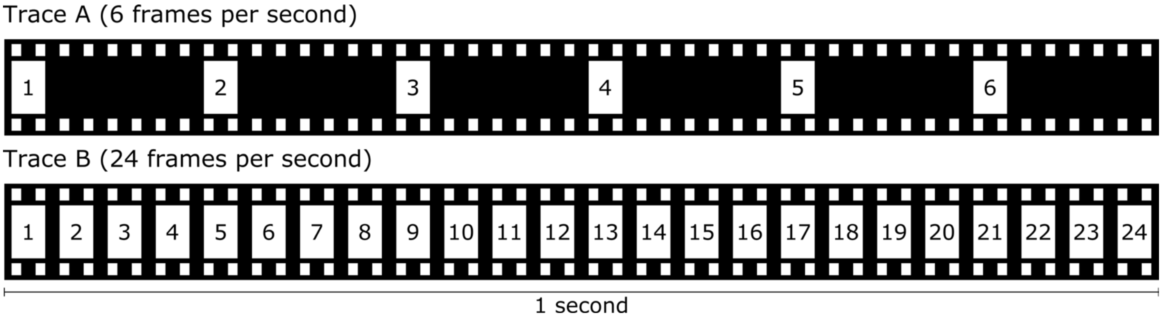


Figure 2.1: Graphical example of frame rate.

In uncompressed video traces, each displayed frame comes from a complete image of the scene. In order to achieve effective transmission and storage of a video trace (especially in HQ video traces), use of compression methods is necessary and this is how a coding standard as H.264 becomes useful [20].

The compression techniques used in H.264 are based on inter-frame prediction mainly and on other techniques such as quantization and entropy coding secondly. H.264 uses motion compensation where image frames are broken up into blocks and movement is predicted based on pre-coded frames. For a block to be coded, prediction images are searched for the most similar block and the motion between these blocks is represented by a motion vector and prediction information [21].

2.1 Encoded Video Trace Structure

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). An I-Frame is a fully specified frame (picture) of the displayed scene, like a conventional static image file. It is completely self-referential, it does not use any information from any other frame in the trace and it provides a point of access to the compressed video data. A P-Frame on the other hand, contains only changes in the picture that occurred from the previous frame. The encoder has to reference backwards to the previous I-Frame or P-Frames in order to retrieve redundant picture information and thus P-Frames are saving space. Finally, a B-Frame uses differences between the current frame and both preceding and following frames in order to specify its information (i.e., it is predicted by looking at both directions – bidirectional prediction). An example of the I, B and P-Frames concept is depicted in Figure (2.2). Regarding their size, P-Frames are smaller than I-Frames and B-Frames are the smallest of the three [22].

Figure 2.2: Graphical example of I, B and P-Frames concept.

Video frames are grouped together in GOP structures (Group of Pictures) 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. 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 Figure (2.3), we can observe that we have a GOP structure of 9 frames, where X distance equals to 8 and Y distance equals to 2. In general, according to the H.264

standard, the amount of B-Frames is greater than the amount of I or P-Frames inside a GOP structure. More details on the data collection can be found in the next section.

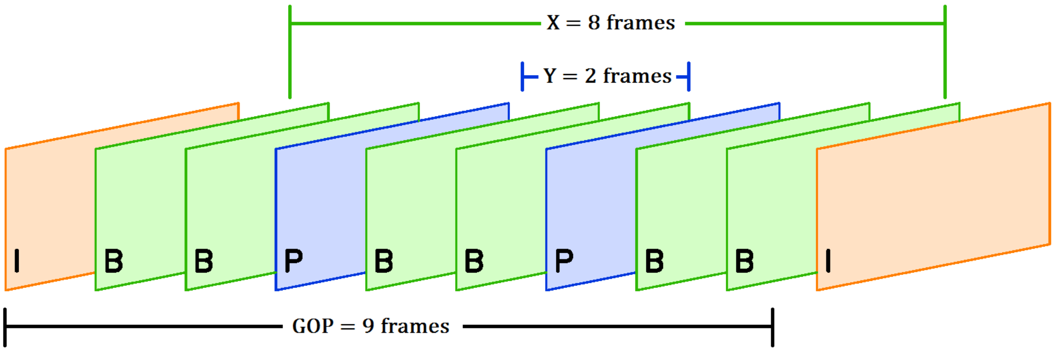


Figure 2.3: Graphical example of GOP structure.

3 Data Collection Methodology

Our work is based on real user-generated data from a large enterprise, whose name will not be referred for reasons of anonymity. Each employee of the enterprise that took park in the data collection, ran trace collection scripts for about a month. One script polled the operating system every 33.3 msec to record the name of the main application that the employee was working on. Another script recorded the employee's screen at 30 fps using encoding parameters that resembled a Miracast hardware encoder as much as possible. 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 A.M. and stopped at 7 P.M. When a user 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. More details on the encodings can be found in Section 3.2.

3.1 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 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 is the same as the PC's screen resolution (i.e., it is not a constant) and the frame rate is 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). The command used for FFmpeg setup is the following:

```
ffmpeg.exe -f gdigrab -video_size %CurrentHorizontalResolution%x%CurrentVerticalResolution% -
framerate 30 -an -i desktop -r 30 -t 10800 -vcodec libx264 -crf 10 -x264opts keyint=60: min-
keyint=60:no-scenecut -b 120000k -tune zerolatency -psnr -pix_fmt yuv422p -threads 0 -preset fast
-loglevel 48 -f null null 2> tmp\videostats_%@computername%_%@filename%.txt
```

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 msec (in order to keep up with FFmpeg logs, where we had 1 frame every 33.3 msec). 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.

3.2 Datasets Overview

3.2.1 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 size 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.

3.2.2 Recording Periods and Datasets Statistics

Our recording framework was running 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 have replaced every user's name with a different letter from the alphabet (i.e., UserA, UserB, UserD, UserE) for reasons of anonymity.

In Table (3.1), 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. In Tables (3.2) to (3.7), we summarize the same statistics for every application separately.

	Dataset 1	**Dataset 2**
# of Recording Days	24	22
# of Users	3	4
# of Applications	29	26
# of Video Traffic Records	14932183	20892611
Total Size of Video Traffic (GBytes)	120	424
MIN Video Traffic Size (Bytes)	159	190
AVG Video Traffic Size (Bytes)	8032	20298
MAX Video Traffic Size (Bytes)	598613	422435
# of I-Frames	249061	290
Total Size of I-Frames (GBytes)	99	0,092
MIN I-Frame Size (Bytes)	3290	129932
AVG I-Frame Size (Bytes)	397525	316900
MAX I-Frame Size (Bytes)	1536577	395780
# of P-Frames	14683122	20892321
Total Size of P-Frames (GBytes)	21	423,908
MIN P-Frame Size (Bytes)	159	190
AVG P-Frame Size (Bytes)	1425	20293
MAX P-Frame Size (Bytes)	598613	422435

Table 3.1: Dataset 1 and Dataset 2 statistics over all applications.

Dataset 1 I and P Frames	**# of Records**	**Total Size** (Bytes)	**MIN Size** (Bytes)	**AVG Size** (Bytes)	**MAX Size** (Bytes)
Acrobat Reader	1203240	7193455153	162	5978.40	899504
Microsoft Excel	81411	611832413	162	7515.35	739279
Foxit Reader	193312	1946742945	161	10070.47	1127390
InSite	210233	1530425284	163	7279.66	833711
Matlab	5541003	43488168101	159	7848.43	1396685
Microsoft Outlook	1501845	12338765111	160	8215.74	1444639
Microsoft PowerPoint	239325	2011160784	161	8403.47	1300609
Enterprise Device Manager	3551	30555040	162	8604.63	758180
Snipping Tool	3720	17300814	169	4650.76	385209
Microsoft Word	1020906	7166688310	161	7019.93	1343834
WinMerge	1245	9118536	161	7324.13	406078
WinSCP	78211	632606616	171	8088.46	1228935
Xwin Cygwin	1620	9345560	161	5768.86	615388
Windows Calculator	54916	445203628	161	8106.99	572823
Google Chrome	2004502	16546146863	161	8254.49	1242142
Command Line	128417	1085453712	160	8452.57	766030
Communicatior	17620	113288482	165	6429.54	640111
Mozilla Firefox	762798	8335279432	159	10927.24	1396106
Google Earth	149938	2809589467	161	18738.34	1536577
G-Simple	137239	350322699	162	2552.65	561190
Internet Explorer	958501	7661779538	161	7993.50	1092248
KDiff3	89492	986785150	161	11026.52	1275036
Kile LaTeX	40304	487629204	163	12098.78	957175
Windows Paint	65502	388890351	161	5937.08	678945
Windows Notepad	1450	4240516	170	2924.49	560425
Notepad++	396846	3329110155	160	8388.92	1399583
Windows PowerShell	28215	238280569	162	8445.17	635763
Windows Task Manger	13206	113862120	161	8622.00	725557
VLC	3615	52046620	164	14397.41	564589

Table 3.2: Dataset 1 statistics for I and P-Frames for every application separately.

Dataset 1 **I Frames**	**# of** **Records**	**Total Size** (Bytes)	**MIN** **Size** (Bytes)	**AVG** **Size** (Bytes)	**MAX** **Size** (Bytes)
Acrobat Reader	20059	6465212781	39115	322309.83	899504
Microsoft Excel	1359	538964417	63775	396588.97	739279
Foxit Reader	3212	1488990381	58592	463571.10	1127390
InSite	3518	1336711357	37245	379963.43	833711
Matlab	92347	36778295485	3380	398261.94	1396685
Microsoft Outlook	25090	10391304678	17792	414161.21	1444639
Microsoft PowerPoint	3990	1568816708	3290	393187.14	1300609
Enterprise Device Manager	55	22103241	28495	401877.11	758180
Snipping Tool	63	15154937	119253	240554.56	385209
Microsoft Word	17100	6274459767	28431	366927.47	1343834
WinMerge	20	7401594	278371	370079.70	406078
WinSCP	1305	569117856	125116	436105.64	1228935
Xwin Cygwin	26	8007678	201273	307987.62	615388
Windows Calculator	915	389945158	96604	426169.57	572823
Google Chrome	33418	13842060352	35051	414209.72	1242142
Command Line	2144	922789371	55410	430405.49	766030
Communicatior	295	96674157	138811	327709.01	640111
Mozilla Firefox	12714	5749222246	60849	452196.18	1396106
Google Earth	2498	2172024072	128526	869505.23	1536577
G-Simple	2293	291972833	46597	127332.24	561190
Internet Explorer	15987	5697199251	28619	356364.50	1092248
KDiff3	1486	716972971	101599	482485.18	1275036
Kile LaTeX	673	365887350	62498	543666.20	957175
Windows Paint	1093	316886605	96583	289923.70	678945
Windows Notepad	24	3629741	76471	151239.21	560425
Notepad++	6627	2630747671	26787	396974.15	1399583
Windows PowerShell	471	224313034	101805	476248.48	635763
Windows Task Manger	219	95192659	71224	434669.68	725557
VLC	60	24329443	135329	405490.72	564589

Table 3.3: Dataset 1 statistics for I-Frames for every application separately.

Dataset 1 **P Frames**	**# of** **Records**	**Total Size** (Bytes)	**MIN** **Size** (Bytes)	**AVG** **Size** (Bytes)	**MAX** **Size** (Bytes)
Acrobat Reader	1183181	728242372	162	615.50	407388
Microsoft Excel	80052	72867996	162	910.26	292361
Foxit Reader	190100	457752564	161	2407.96	439189
InSite	206715	193713927	163	937.11	355725
Matlab	5448656	6709872616	159	1231.47	571579
Microsoft Outlook	1476755	1947460433	160	1318.74	513768
Microsoft PowerPoint	235335	442344076	161	1879.64	408011
Enterprise Device Manager	3496	8451799	162	2417.56	279689
Snipping Tool	3657	2145877	169	586.79	45254
Microsoft Word	1003806	892228543	161	888.85	514237
WinMerge	1225	1716942	161	1401.59	224431
WinSCP	76906	63488760	171	825.54	598613
Xwin Cygwin	1594	1337882	161	839.32	96211
Windows Calculator	54001	55258470	161	1023.29	314185
Google Chrome	1971084	2704086511	161	1371.88	460425
Command Line	126273	162664341	160	1288.20	496366
Communicatior	17325	16614325	165	958.98	209845
Mozilla Firefox	750084	2586057186	159	3447.69	513537
Google Earth	147440	637565395	161	4324.24	503842
G-Simple	134946	58349866	162	432.39	202106
Internet Explorer	942514	1964580287	161	2084.40	547021
KDiff3	88006	269812179	161	3065.84	523077
Kile LaTeX	39631	121741854	163	3071.88	407035
Windows Paint	64409	72003746	161	1117.91	312623
Windows Notepad	1426	610775	170	428.31	74259
Notepad++	390219	698362484	160	1789.67	597929
Windows PowerShell	27744	13967535	162	503.44	242113
Windows Task Manger	12987	18669461	161	1437.55	230692
VLC	3555	27717177	164	7796.67	314342

Table 3.4: Dataset 1 statistics for P-Frames for every application separately.

Dataset 2 I and P Frames	# of Records	Total Size (Bytes)	MIN Size (Bytes)	AVG Size (Bytes)	MAX Size (Bytes)
Acrobat Reader	91191	2136792352	562	23432.05	408677
Microsoft Excel	802214	11382628974	600	14189.02	399526
Foxit Reader	150085	3859703201	547	25716.78	405206
Matlab	5338049	86952178144	446	16289.13	416726
Microsoft Outlook	3998224	86075641522	411	21528.47	411819
Microsoft PowerPoint	550143	9344958975	345	16986.42	400006
Enterprise Device Manager	2381	47255359	816	19846.85	395807
Snipping Tool	3420	56130953	1106	16412.56	369435
Microsoft Word	2440966	43708319774	558	17906.16	413769
WinMerge	620	15726092	1925	25364.66	392312
WinRAR	2505	76806323	2106	30661.21	392326
Xwin Cygwin	47593	830209970	1568	17443.95	403309
Windows Calculator	1585	23744668	3154	14980.86	41932
Google Chrome	2204967	54911911822	199	24903.73	422435
Command Line	161485	3945046228	712	24429.80	401922
Mozilla Firefox	2959650	81519056306	417	27543.48	415133
IrfanView	335731	6535872410	573	19467.59	401369
Internet Explorer	514782	9398257333	508	18256.77	403586
KDiff3	10155	313811196	1326	30902.14	399990
Windows Paint	304319	4706916493	535	15467.05	402482
Windows Notepad	16165	176375107	2111	10910.93	395659
Notepad++	912572	16449621518	476	18025.56	407272
Windows PowerShell	6693	101359114	190	15144.05	393740
Windows Task Manger	2840	65874645	1680	23195.30	397773
VLC	23251	785190493	495	33770.18	403801
VMware Player	11025	655871597	645	59489.49	395007

Table 3.5: Dataset 2 statistics for I and P-Frames for every application separately.

Dataset 2 I Frames	# of Records	Total Size (Bytes)	MIN Size (Bytes)	AVG Size (Bytes)	MAX Size (Bytes)
Acrobat Reader	---	---	---	---	---
Microsoft Excel	2	697894	314583	348947.00	383311
Foxit Reader	---	---	---	---	---
Matlab	1	384098	384098	384098.00	384098
Microsoft Outlook	28	9468049	129932	338144.61	395780
Microsoft PowerPoint	6	1801799	198561	300299.83	347910
Enterprise Device Manager	---	---	---	---	---
Snipping Tool	---	---	---	---	---
Microsoft Word	9	2874354	214923	319372.67	373591
WinMerge	---	---	---	---	---
WinRAR	---	---	---	---	---
Xwin Cygwin	---	---	---	---	---
Windows Calculator	---	---	---	---	---
Google Chrome	203	63983635	256644	315190.32	364414
Command Line	1	377542	377542	377542.00	377542
Mozilla Firefox	14	4426557	151974	316182.64	388909
IrfanView	---	---	---	---	---
Internet Explorer	---	---	---	---	---
KDiff3	---	---	---	---	---
Windows Paint	---	---	---	---	---
Windows Notepad	---	---	---	---	---
Notepad++	2	609506	283571	304753.00	325935
Windows PowerShell	24	7277716	252290	303238.17	393740
Windows Task Manger	---	---	---	---	---
VLC	---	---	---	---	---
VMware Player	---	---	---	---	---

Table 3.6: Dataset 2 statistics for I-Frames for every application separately.

For many applications no statistics are shown in the table above, due to the absence of I-Frames in Dataset 2.

Dataset 2 **P Frames**	**# of Records**	**Total Size** (Bytes)	**MIN Size** (Bytes)	**AVG Size** (Bytes)	**MAX Size** (Bytes)
Acrobat Reader	91191	2136792352	562	23432.05	408677
Microsoft Excel	802212	11381931080	600	14188.18	399526
Foxit Reader	150085	3859703201	547	25716.78	405206
Matlab	5338048	86951794046	446	16289.06	416726
Microsoft Outlook	3998196	86066173473	411	21526.25	411819
Microsoft PowerPoint	550137	9343157176	345	16983.33	400006
Enterprise Device Manager	2381	47255359	816	19846.85	395807
Snipping Tool	3420	56130953	1106	16412.56	369435
Microsoft Word	2440957	43705445420	558	17905.05	413769
WinMerge	620	15726092	1925	25364.66	392312
WinRAR	2505	76806323	2106	30661.21	392326
Xwin Cygwin	47593	830209970	1568	17443.95	403309
Windows Calculator	1585	23744668	3154	14980.86	41932
Google Chrome	2204764	54847928187	199	24877.01	422435
Command Line	161484	3944668686	712	24427.61	401922
Mozilla Firefox	2959636	81514629749	417	27542.11	415133
IrfanView	335731	6535872410	573	19467.59	401369
Internet Explorer	514782	9398257333	508	18256.77	403586
KDiff3	10155	313811196	1326	30902.14	399990
Windows Paint	304319	4706916493	535	15467.05	402482
Windows Notepad	16165	176375107	2111	10910.93	395659
Notepad++	912570	16449012012	476	18024.93	407272
Windows PowerShell	6669	94081398	190	14107.27	333654
Windows Task Manger	2840	65874645	1680	23195.30	397773
VLC	23251	785190493	495	33770.18	403801
VMware Player	11025	655871597	645	59489.49	395007

Table 3.7: Dataset 2 statistics for P-Frames for every application separately.

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 (3.1).

4 Statistical Tests and Evaluation Metrics

In this thesis, we developed and tested various modeling techniques on our data, as we analyze further in Chapter 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 chapter. We also present the metrics we utilized for assessing the quality of our proposed models, which will be presented in Chapter 5.

4.1 Distributions Fitting

In the following two subsections, we explain the procedure that we have followed in order to try to fit our data 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.

4.1.1 Maximum Likelihood Estimation

Maximum Likelihood Estimation (MLE) is a method of parameter estimation in statistics [26], which we used for finding the parameters of a distribution, based on our data.

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 $\widehat{\Theta}$ of this vector, based on our data. Hence, we used the MLE method in order to seek a vector $\widehat{\Theta}$, which can be as close as possible to the true Θ and by that tried to estimate the parameters of the distribution assumed to underlie our data.

This can be done by taking the joint probability density function of the observations (i.e., our data) given the, unknown to us, set of true parameters Θ and assuming their independency. This joint probability density function is

$$\mathrm{f}(\mathrm{x}_1,\mathrm{x}_2,\ldots,\mathrm{x}_\mathrm{n}|\Theta) = \mathrm{f}(\mathrm{x}_1|\Theta) \cdot \mathrm{f}(\mathrm{x}_2|\Theta) \cdot \ldots \cdot \mathrm{f}(\mathrm{x}_\mathrm{n}|\Theta) \tag{4.1}$$

where n is the amount of I.I.D. observations. Now, if we fix the values x_i as *parameters* of this function and consider Θ as the function's variable, we obtain the likelihood function, which is

$$\mathcal{L}(\Theta;x_1,x_2,...,x_n) = f(x_1,x_2,...,x_n|\Theta) = \prod_{i=1}^{n} f(x_i|\Theta) \qquad (4.2)$$

where ";" denotes a simple separation. For computational convenience, it is better to use the logarithmic version of (4.3), called the log-likelihood function, which is

$$\ln \mathcal{L}(\Theta;x_1,x_2,...,x_n) = \sum_{i=1}^{n} \ln f(x_i|\Theta) \qquad (4.3)$$

The average log-likelihood function is

$$\hat{e} = \frac{1}{n} \ln \mathcal{L} \qquad (4.4)$$

The MLE method finds the estimator $\widehat{\Theta}$ by finding a value of Θ that maximizes $\hat{e}(\Theta;x)$. Finally, if a maximum exists, it can be found and sometimes more than one estimates can be found that maximize the average log-likelihood function. For many models (in our case distributions) there is an explicit form to calculate the parameters but for many others there is not. In those cases, optimization methods (such as Newton's Method) have to be used.

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 those distributions, we used the MLE method for the parameters estimation and the built-in Matlab functions for the data generation.

4.2 Statistical Tests

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

4.2.1 Quantiles-Quantiles Plot

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 [28]. 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 generated data via the distribution.

4.2.2 Kolmogorov-Smirnov Test

In order to further verify the validity of our results, we performed the Kolmogorov-Smirnov test [29]. The Kolmogorov-Smirnov test or KS test tries to determine if two datasets differ significantly. The KS test 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. As explained in [27], the use of KS test is a good statistical tool; however it has the drawback that KS test give 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 5% 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 thought 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 called Two-Sample Kolmogorov-Smirnov Test. The test measure is given by Equation (4.5) for two given Cumulative Distribution Functions (CDFs) F_1 and F_2.

$$D_{n,n'} = \sup(|F_{1,n}(x) - F_{2,n}(x)|) \tag{4.5}$$

The null hypothesis is rejected at the level “a” significance if

$$D_{n,n'} > c(a) \cdot \sqrt{\frac{n + n'}{n \cdot n'}} \tag{4.6}$$

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, we would like to mention that the KS test has two limitations. First, it works only with continuous distribution (and that is the reason that we do not have results for the Geometric and Negative Binomial distribution in Chapter 5) and second, it is more sensitive at the "center" of the CDFs of the distributions rather than the "tails" (a limitation that we try to overwhelm by using Anderson-Darling test in the follow).

4.2.3 Anderson-Darling Test

The Anderson-Darling test or 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

$$n \int_{-\infty}^{\infty} (F_n(x) - F(x))^2 \ w(x) \ dF(x) \tag{4.7}$$

over the ordered sample values $x_1<x_2<...<x_n$, 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

$$w(x) = [F(x) \cdot (1 - F(x))]^{-1} \tag{4.8}$$

The AD test statistic is

$$A_n^2 = -n - \sum_{i=1}^{n} \frac{2i - 1}{n} \left[\ln(F(X_i)) + \ln(1 - F(X_{n+1-i}))\right] \tag{4.9}$$

where $X_1<X_2<...<X_n$ 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.

4.3 Accuracy Evaluation Metrics

In this final section of this chapter, we present the three metrics that we used, in order to evaluate the accuracy of our models.

4.3.1 Mean Absolute Percentage Error

The Mean Absolute Percentage Error (MAPE) [31] is a metric 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. It depicts, how large the prediction error is in a percentage form. For a set of pairs of real-generated values in a dataset, the MAPE is calculated according to (4.10).

$$\mathrm{MAPE} = \frac{\sum_{i=1}^{n} \frac{|X_{P,i} - X_{R,i}|}{X_{R,i}}}{n} \cdot 100\% \tag{4.10}$$

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

4.3.2 Relative Percentage Error

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 according to (4.11).

$$\mathrm{RPE} = \frac{\sum_{i=1}^{n} |X_{P,i} - X_{R,i}|}{\sum_{i=1}^{n} |X_{R,i}|} \cdot 100\% \tag{4.11}$$

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

4.3.3 Confidence Interval

The Confidence Interval (CI) [33] is an indicator of measurement's precision or how stable an estimation is, i.e., a measure of how close a measurement will be to the original estimate if the experiment were to be repeated. The procedure to calculate the confidence interval on the results of an experiment is the following.

Assume that you choose X samples from the total population of results in your experiment. Calculate the mean value $\overline{X}$ of those samples and the standard deviation σ. Choose the desired confidence level C in percentage format and find the critical value $Z_{a/2}$ from the Z-table [34], where $a = 1 - c$ and $c =$ *decimal format of C*. Find the standard error e according to formula $e = Z_{a/2} \cdot \frac{\sigma}{\sqrt{|X|}}$, where $|X|$ is the size of X. In the end, you can define the confidence interval as $CI = \overline{X} \pm e$.

5 Modeling Methodology and Results

In this chapter, we present, analyze and evaluate the four different modeling approaches (simple and more sophisticated), that we have used in this work. Some of them are taken from the literature, as they have been proposed or with tweaks that we have incorporated into them. One method is presented here for the first time in the relevant literature, to the best of our knowledge. This modeling approach is shown to be a robust and highly accurate for the prediction of video traffic that is generated by an average user's computer, during a day.

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. In the following subsections, we analyze the way that the ADA model works and we present our respective results.

5.1.1 Model Analysis

The ADA model is capable of modeling the I-Frames and the P-Frames of video traffic.

Its methodology consists of three basic steps. The first is the data separation into I-Frames and P-Frames according to their characterization in the records. The second is the parameters' estimation of each distribution that possibly characterizes the application using the MLE method and the dataset. The third is the predicted data generation according to this specific distribution using the estimated parameters of the previous step. The last two steps are applied on the I-Frames and the P-Frames separately. Given that the distribution that best characterizes the application's data is unknown, the last two steps have to be repeated for a wide range of well-known distributions and the ones that gives the lower RPE and MAPE will be selected.

We applied the ADA model for the set of ten well-known distributions presented in Subsection 4.1.1. In order to evaluate further the statistical behavior of our model, we applied the KS and AD test 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 of I and P-Frames). The problem with that approach was the fact 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 3.1) and due to this no distribution could serve as a competent model that would reproduce those wide range differences with low errors.

Further, as explained in detail in the following subsection, the ADA model fails to predict the P-Frames traffic with low errors (in Dataset 1 especially). This fact led us to further separate P-Frames into two "partitions" (P-Frames Lower Partition and P-Frames Upper Partition as we named them). The rule for the partitioning of P-Frames was different for the 1st and the 2nd Dataset (because the traffic has different characteristics between those two datasets as depicted in Table 3.1). For the 1st Dataset, the rule is that every P-Frame with size less than 1‰ of the largest P-Frame goes to the Lower Partition and every P-Frame with size greater or equal to 1‰ of the largest P-Frame goes to the Upper Partition. For the 2nd Dataset where we often encounter consecutive P-Frames with identical size, the rule is that all the P-Frames preceding the appearance of 5 consecutive P-Frames with the same size go to the Lower Partition and the rest go to the Upper Partition. Intuitively, those rules split the P-Frames into two groups. The first one contains small and similar sized P-Frames (Lower Partition) and the second one large and dissimilar sized P-Frames (i.e., our "outliers"). The usage of this rule in the 1st Dataset assigns ≈80% of the P-Frames in the Lower Partition and ≈20% in the Upper Partition and for the 2nd Dataset assigns ≈97% of the P-Frames in the Lower Partition and ≈3% in the Upper Partition.

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 to maintain this approach, given the much larger complexity of this approach.

Finally, it should be emphasized that the ADA model has an inherent disadvantage. 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) [35] [36] [37]. Our own results, which will be presented in the following sections, confirm that for all traces SRD exists. Therefore, ADA is used as a first, simple approach and as a benchmark against which our other models will be compared.

5.1.2 Model Results

In this subsection, we are going to evaluate the ADA model by presenting the results from our tests. We have run our 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 RPE, MAPE, KS and AD tests and over all applications. The real and predicted frame sizes are sorted in ascending order for comparison purposes.

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
Enterprise 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 5.1: ADA model results for I-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
Enterprise 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 5.2: ADA model results for P-Frames over all applications of Dataset 1.

From the above two tables that refer to Dataset 1, we observe that the ADA model achieves good accuracy on modeling I-Frames but fails in modeling P-Frames.

As for the I-Frames, we can see that we have low errors (below 10%) in terms of RPE and MAPE for most of our applications (with some exceptions such as Enterprise Device Manager, G-Simple and Windows Notepad) and the average RPE and MAPE over all applications stays below 9%. In addition, we can see that our model agrees on most of the cases for the best distribution per application between RPE and MAPE but we cannot reach a clear conclusion from the KS and AD test, a fact that indicates that our data differs between the "center" and the "tails".

As for the P-Frames, we have very high errors in terms of RPE and significant errors in terms of MAPE. Also, we observe that MAPE, KS and AD test indicate for most of the applications the GEV as best distribution, a fact related with the high concentration of very small sized P-Frames in the Dataset 1.

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 5.3: ADA model results for I-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
Enterprise 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		

Table 5.4: ADA model results for P-Frames over all applications of Dataset 2.

From the above two tables 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) but for this dataset is achieves decent results (RPE 13.8%, MAPE 9.4%) in modeling P-Frames as well.

We also observe once again that our model agrees in most cases about the best distribution per application between RPE and MAPE but we cannot reach a clear conclusion from the KS and AD test.

As for the P-Frames, we observe that there are enough applications for which RPE, MAPE, KS and AD agree for the best distribution (in almost every case the KS test agrees with the AD test), which indicates that the video traffic is more "smoothed" due to the usage of Periodic Intra Refresh in comparison with Dataset 1.

In the following figures, we present the Q-Q Plots of the real and predicted I-Frames and P-Frames for eight major applications (Microsoft Excel, Microsoft Word, Microsoft PowerPoint, Microsoft Outlook, Google Chrome, Mozilla Firefox, Internet Explorer and Matlab) from Dataset 1 and Dataset 2. These applications correspond to ≈82% and ≈90% of the overall recorded video traffic in the two datasets, respectively.

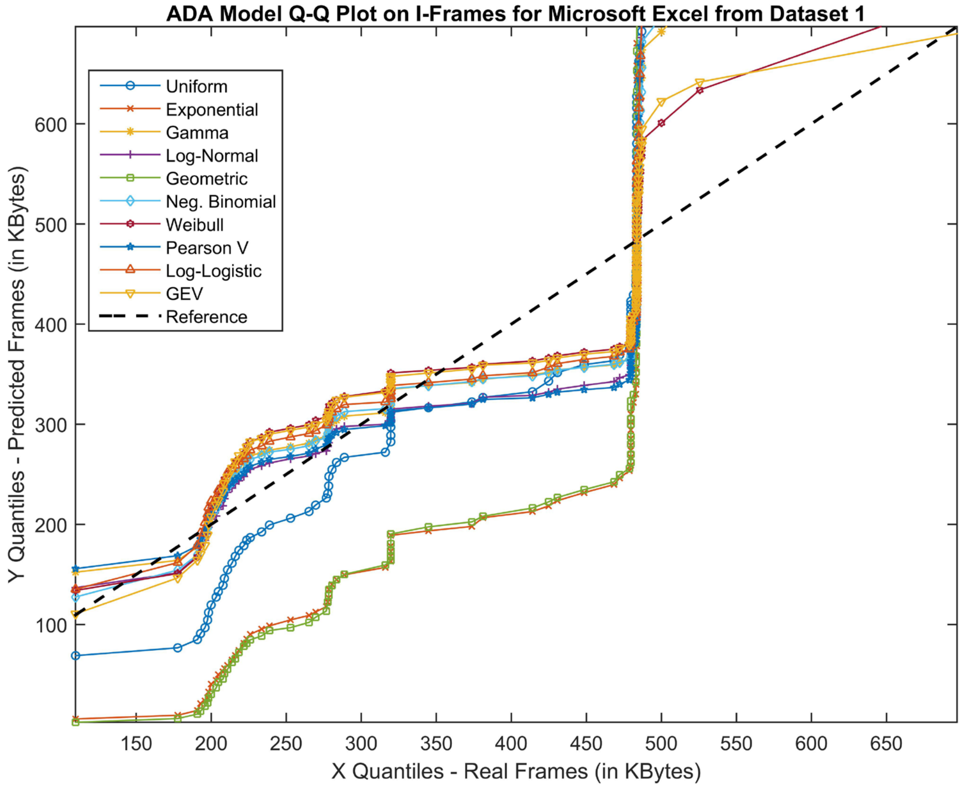


Figure 5.1: Q-Q Plot for Dataset's 1 Microsoft Excel I-Frames from ADA Model.

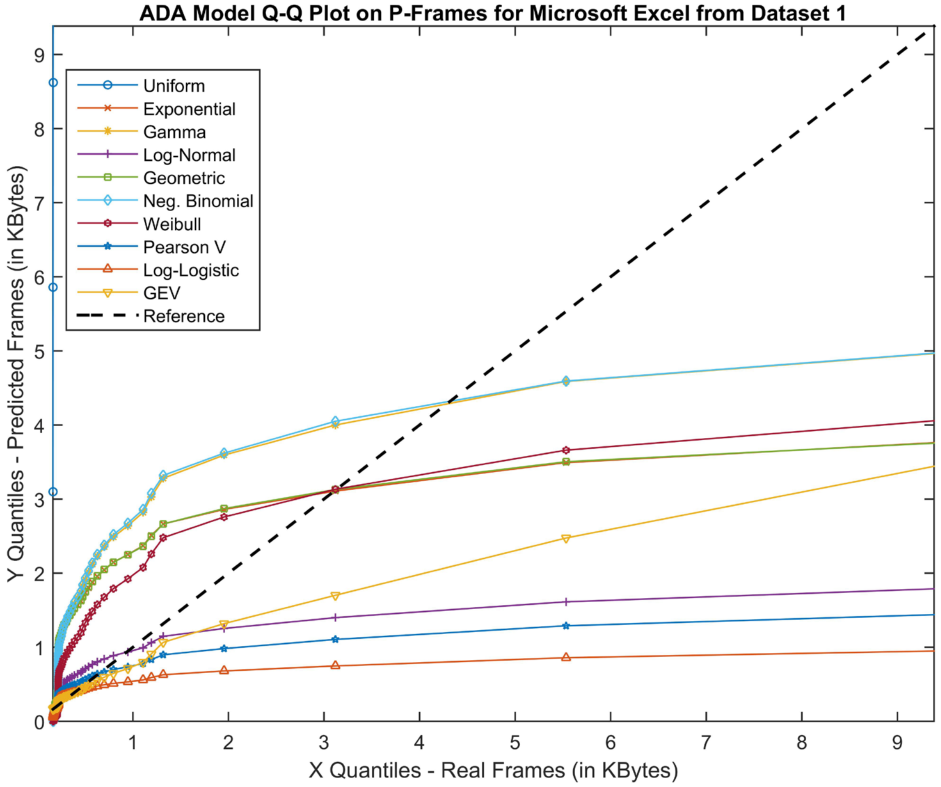


Figure 5.2: Q-Q Plot for Dataset's 1 Microsoft Excel P-Frames from ADA Model.

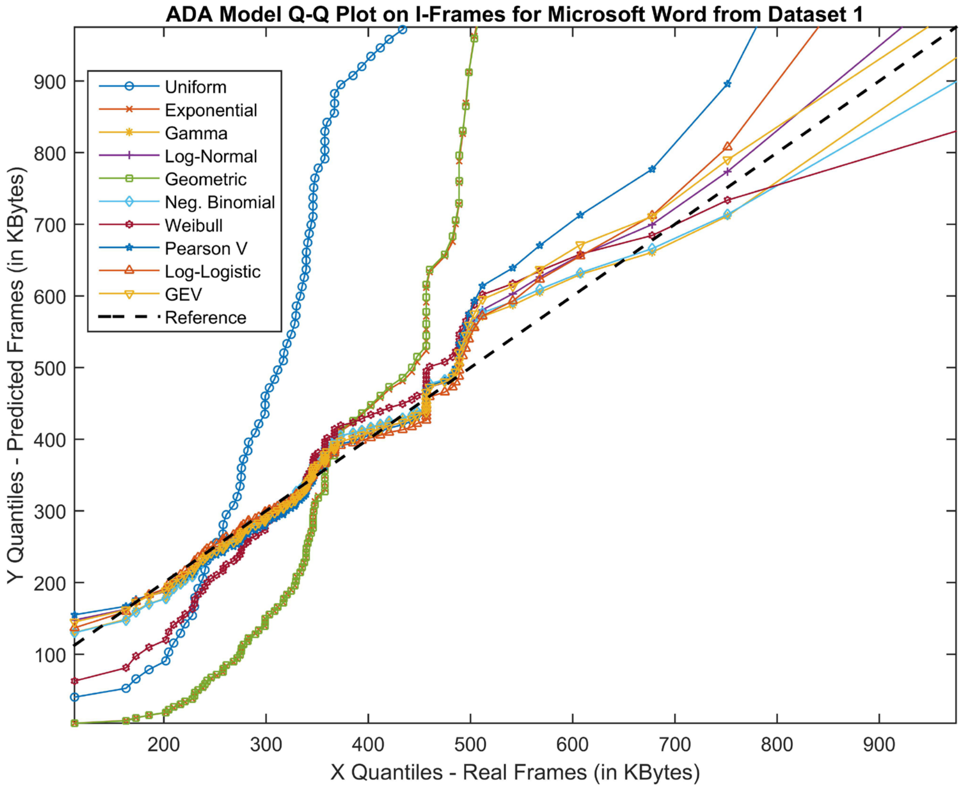


Figure 5.3: Q-Q Plot for Dataset's 1 Microsoft Word I-Frames from ADA Model.

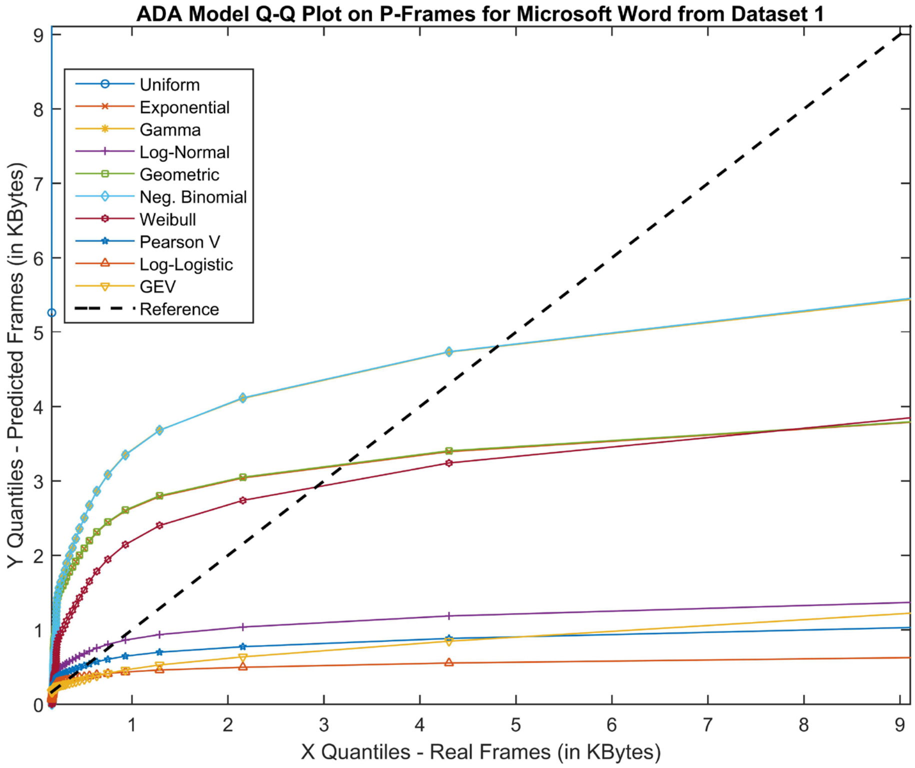


Figure 5.4: Q-Q Plot for Dataset's 1 Microsoft Word P-Frames from ADA Model.

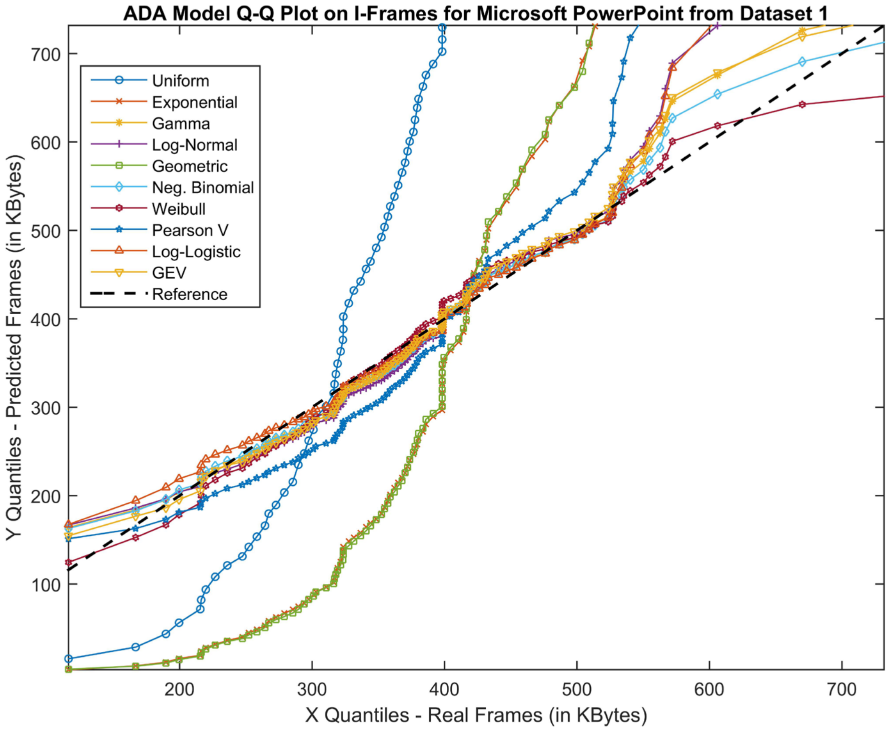


Figure 5.5: Q-Q Plot for Dataset's 1 Microsoft PowerPoint I-Frames from ADA Model.

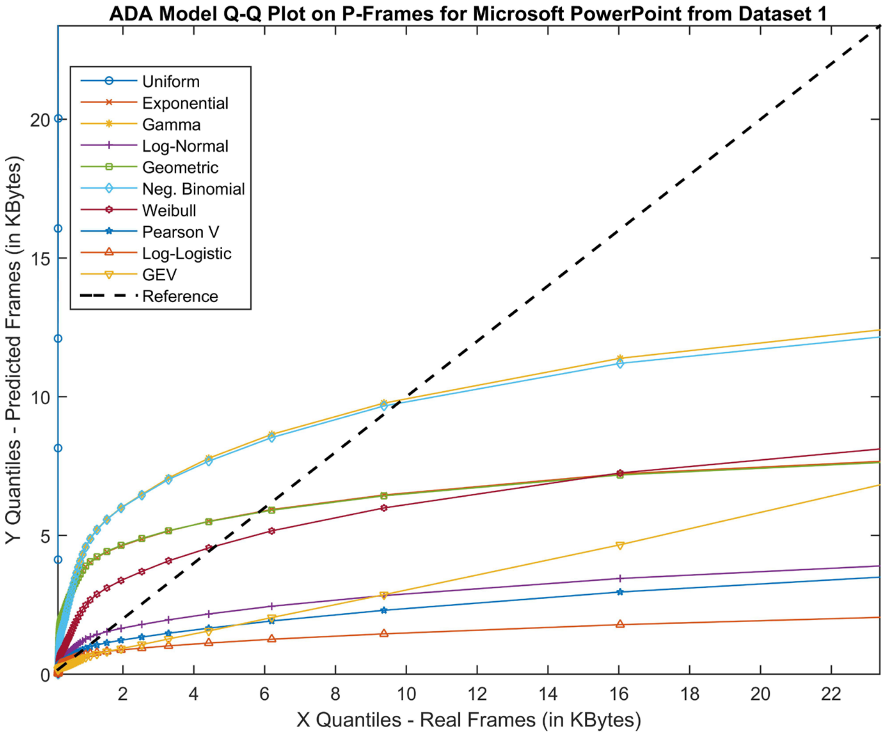


Figure 5.6: Q-Q Plot for Dataset's 1 Microsoft PowerPoint P-Frames from ADA Model.

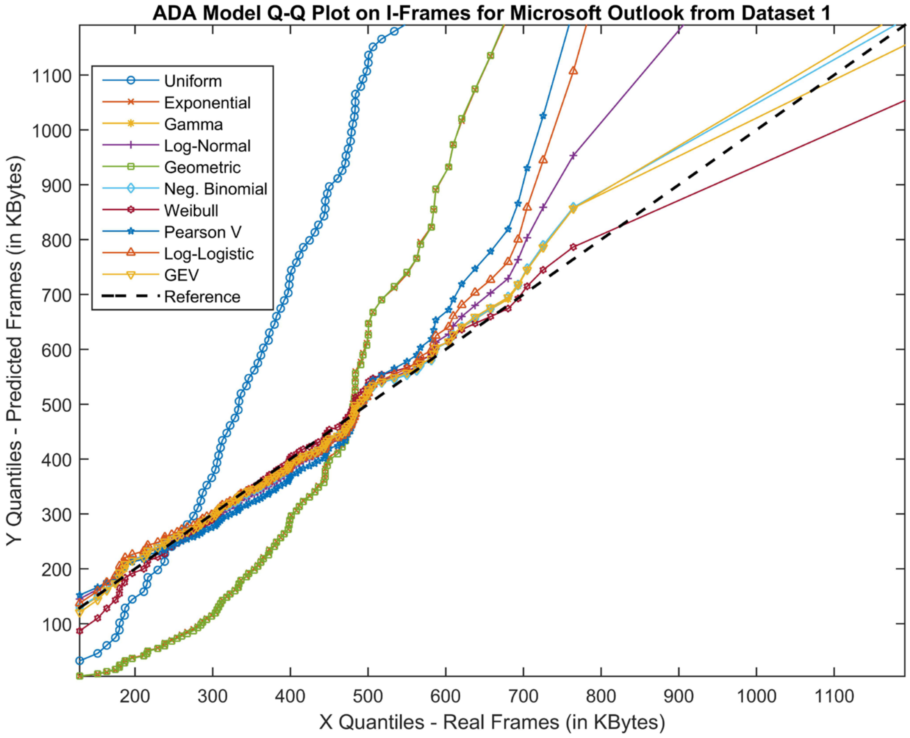


Figure 5.7: Q-Q Plot for Dataset's 1 Microsoft Outlook I-Frames from ADA Model.

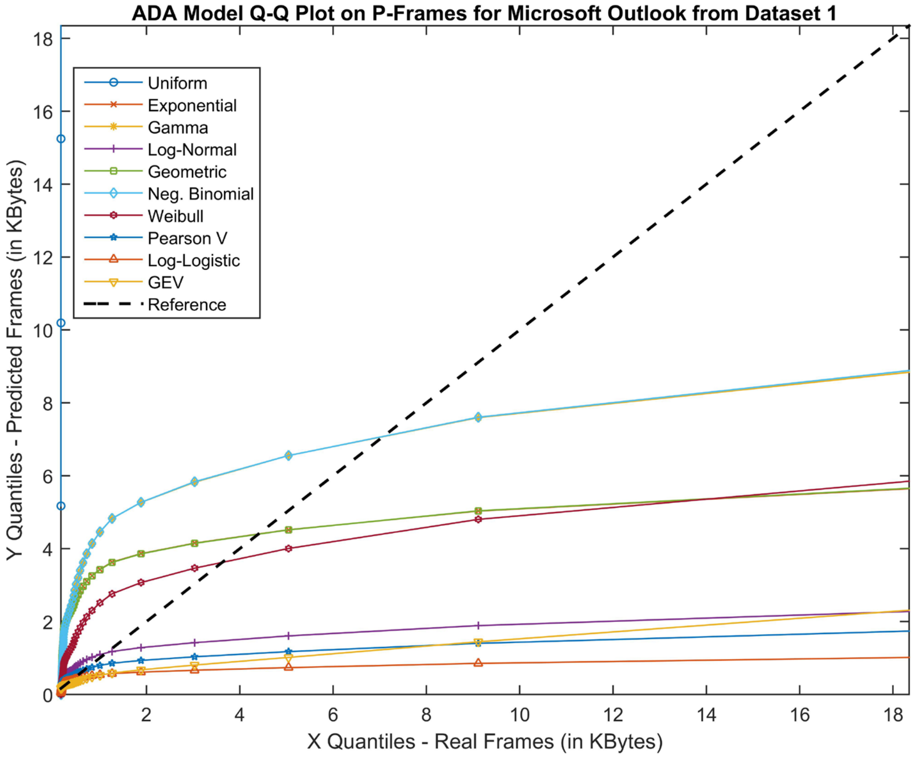


Figure 5.8: Q-Q Plot for Dataset's 1 Microsoft Outlook P-Frames from ADA Model.

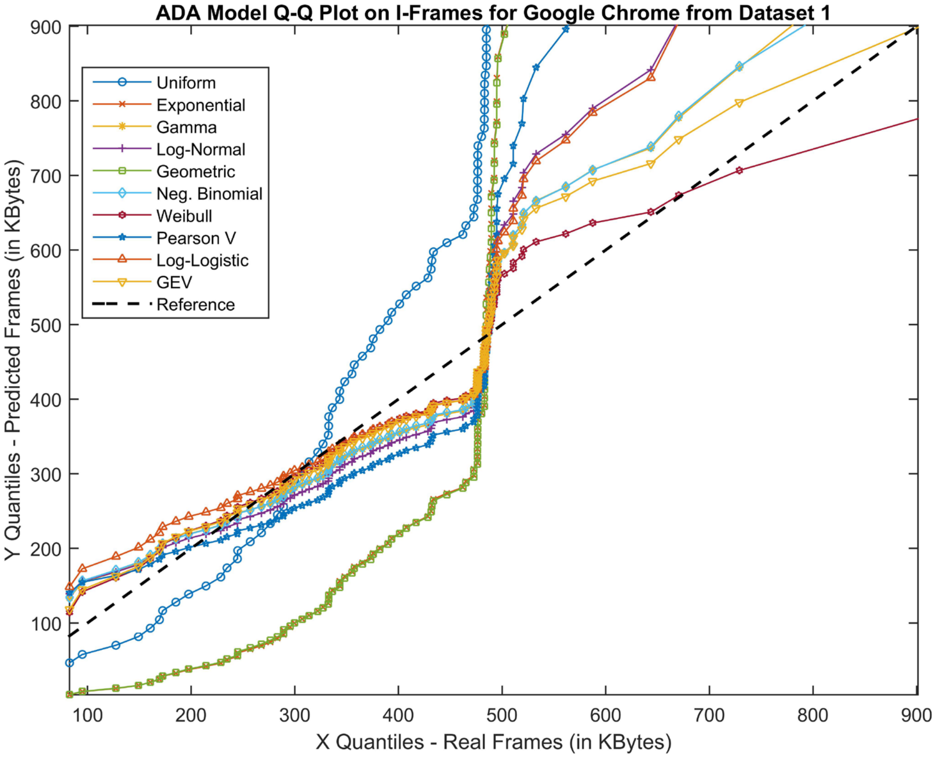


Figure 5.9: Q-Q Plot for Dataset's 1 Google Chrome I-Frames from ADA Model.

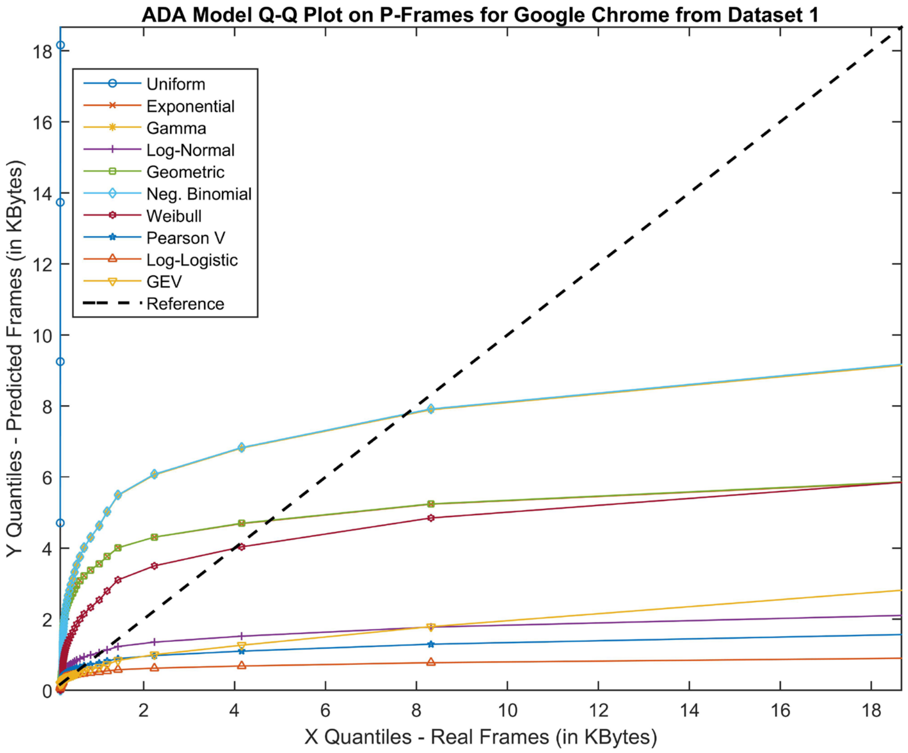


Figure 5.10: Q-Q Plot for Dataset's 1 Google Chrome P-Frames from ADA Model.

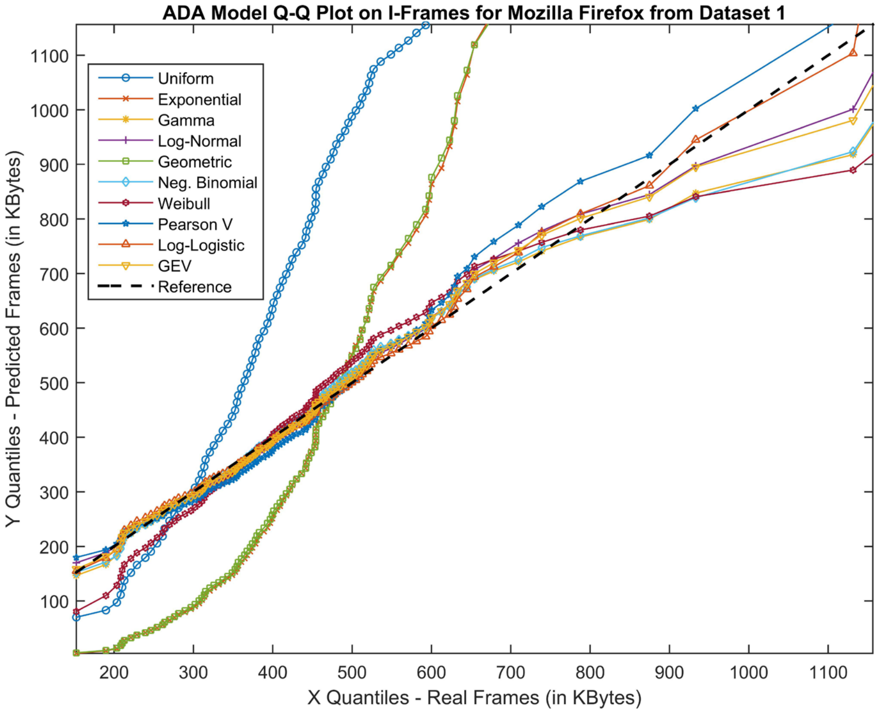


Figure 5.11: Q-Q Plot for Dataset's 1 Mozilla Firefox I-Frames from ADA Model.

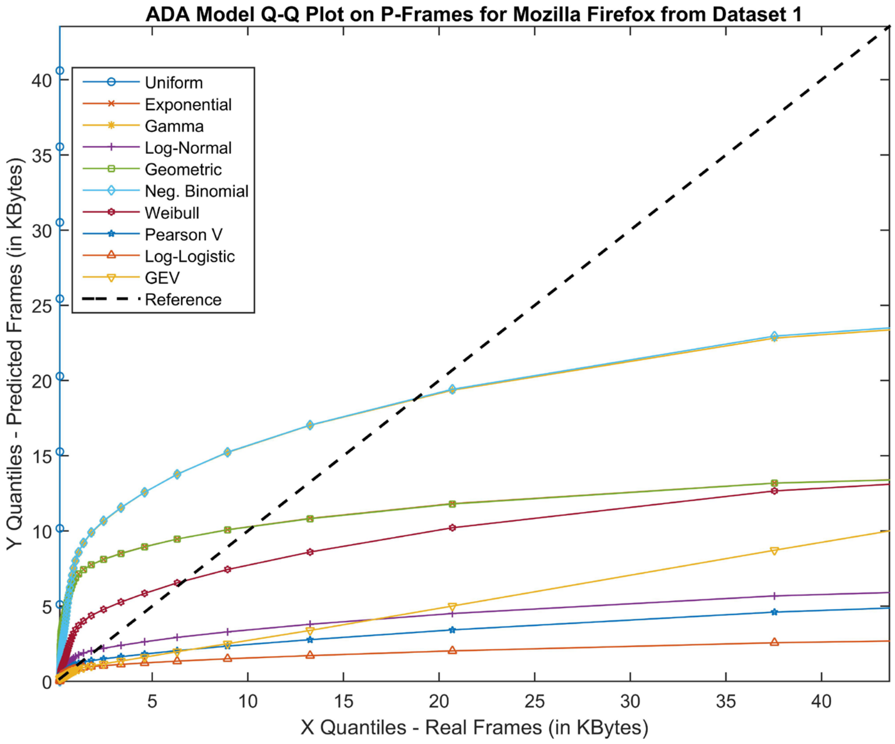


Figure 5.12: Q-Q Plot for Dataset's 1 Mozilla Firefox P-Frames from ADA Model.

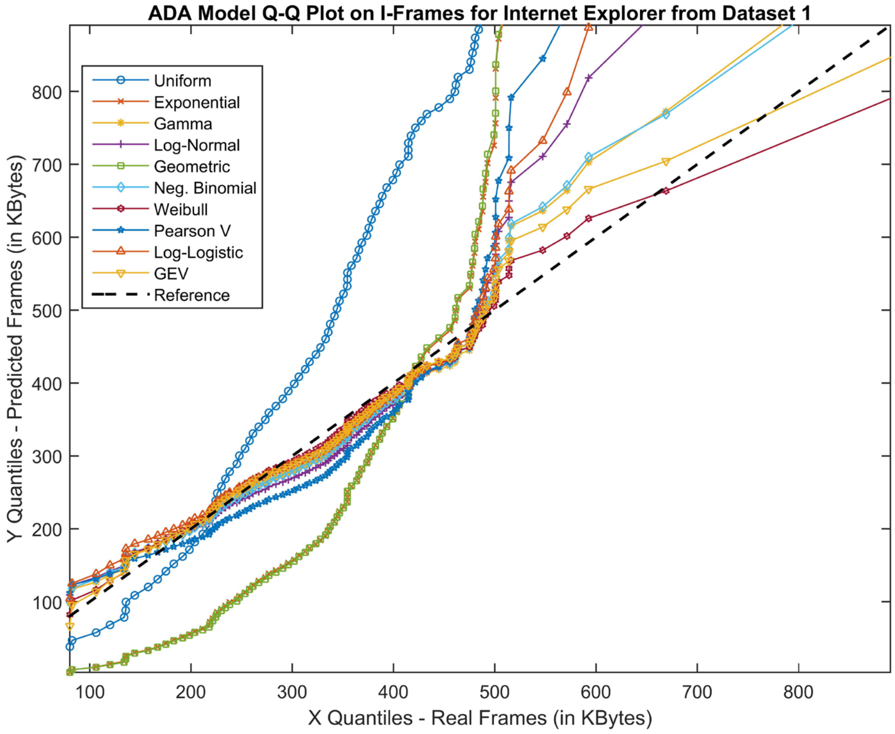


Figure 5.13: Q-Q Plot for Dataset's 1 Internet Explorer I-Frames from ADA Model.

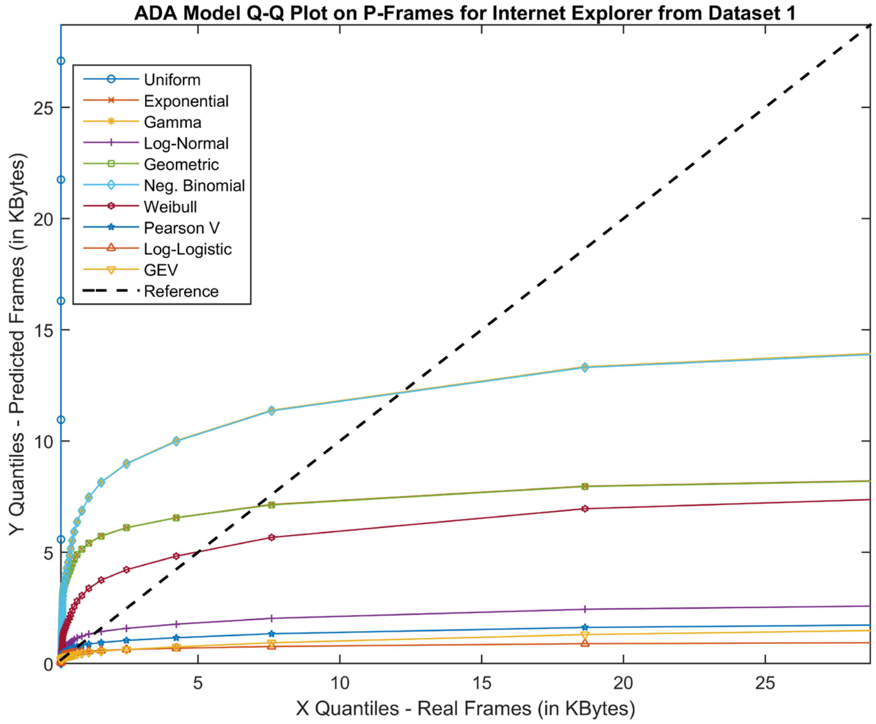


Figure 5.14: Q-Q Plot for Dataset's 1 Internet Explorer P-Frames from ADA Model.

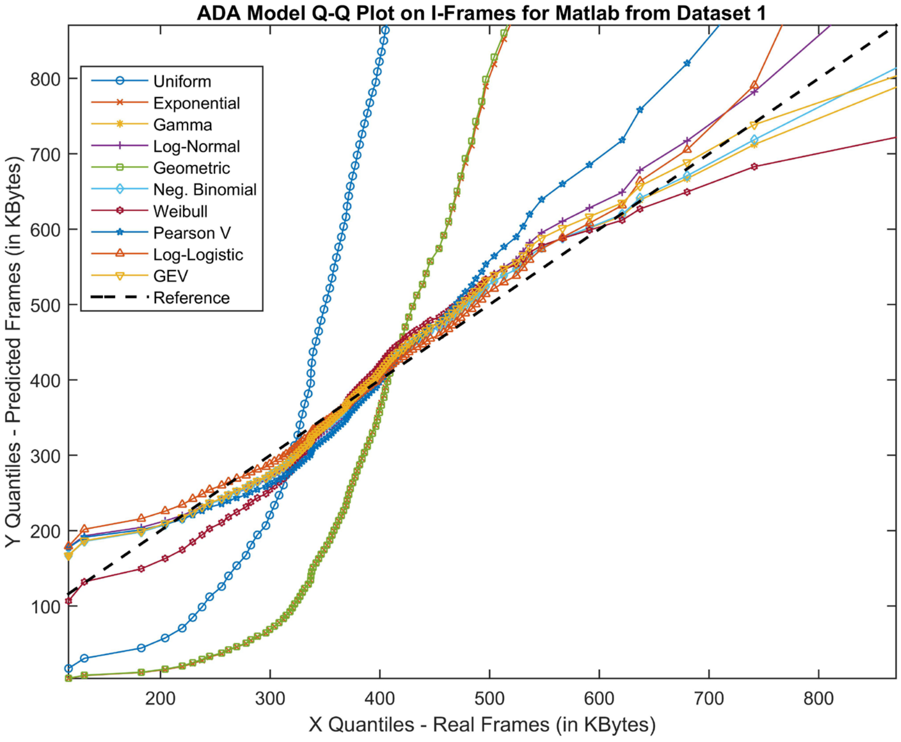


Figure 5.15: Q-Q Plot for Dataset's 1 Matlab I-Frames from ADA Model.

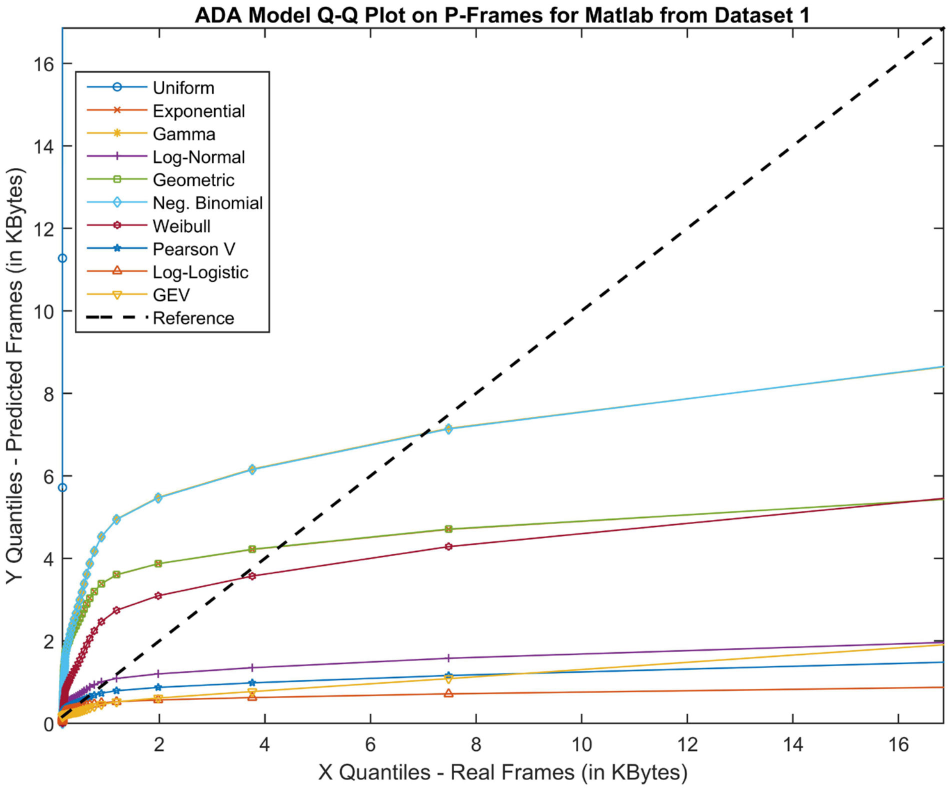


Figure 5.16: Q-Q Plot for Dataset's 1 Matlab P-Frames from ADA Model.

As we can see from Figures (5.1) to (5.16), the Q-Q Plots' results confirm the best distribution fits that the RPE and MAPE metrics indicated for these eight major applications of Dataset 1 in Tables (5.1) and (5.2).

On the other hand, we confirm again that the ADA model is not capable to predict with accuracy the P-Frames of Dataset 1 as depicted in the Q-Q Plots figures of P-Frames. The distributions curves deviate strongly from the Reference Line.

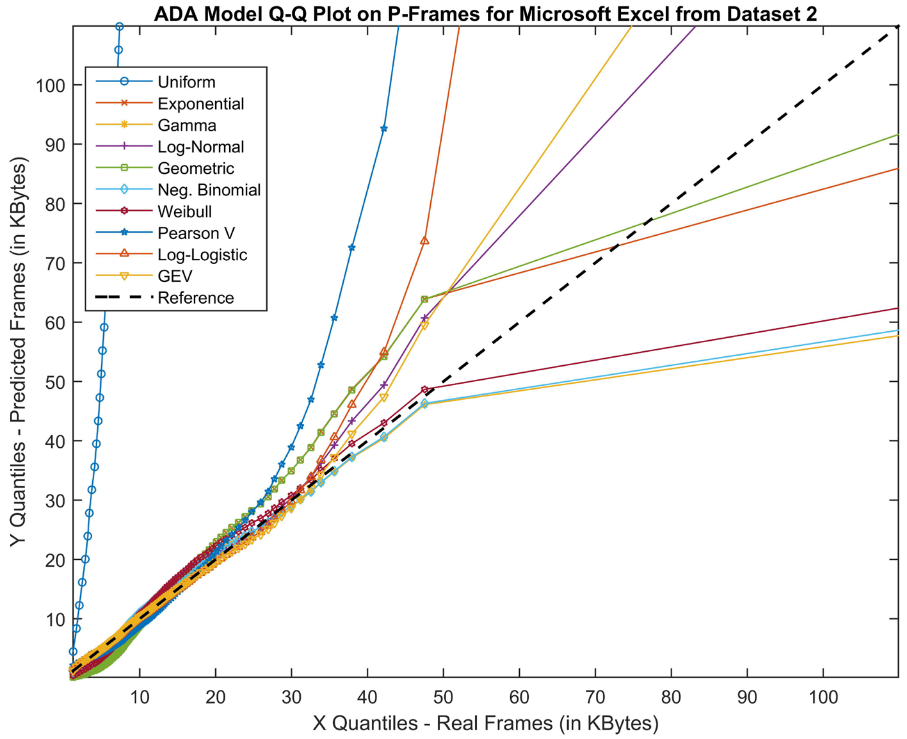


Figure 5.17: Q-Q Plot for Dataset's 2 Microsoft Excel P-Frames from ADA Model.

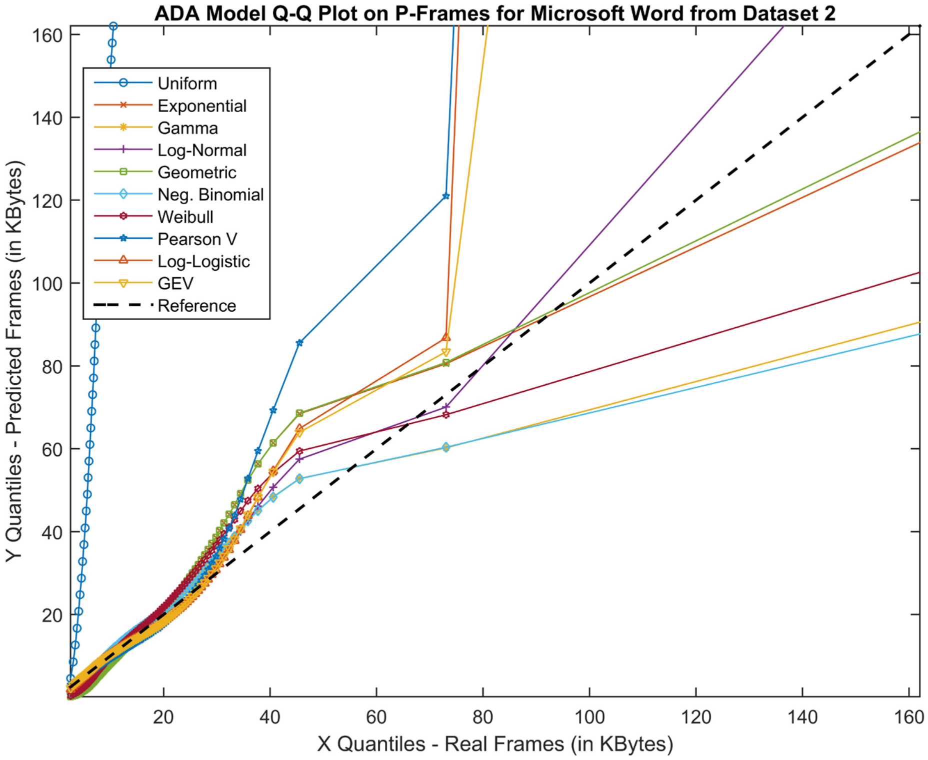


Figure 5.18: Q-Q Plot for Dataset's 2 Microsoft Word P-Frames from ADA Model.

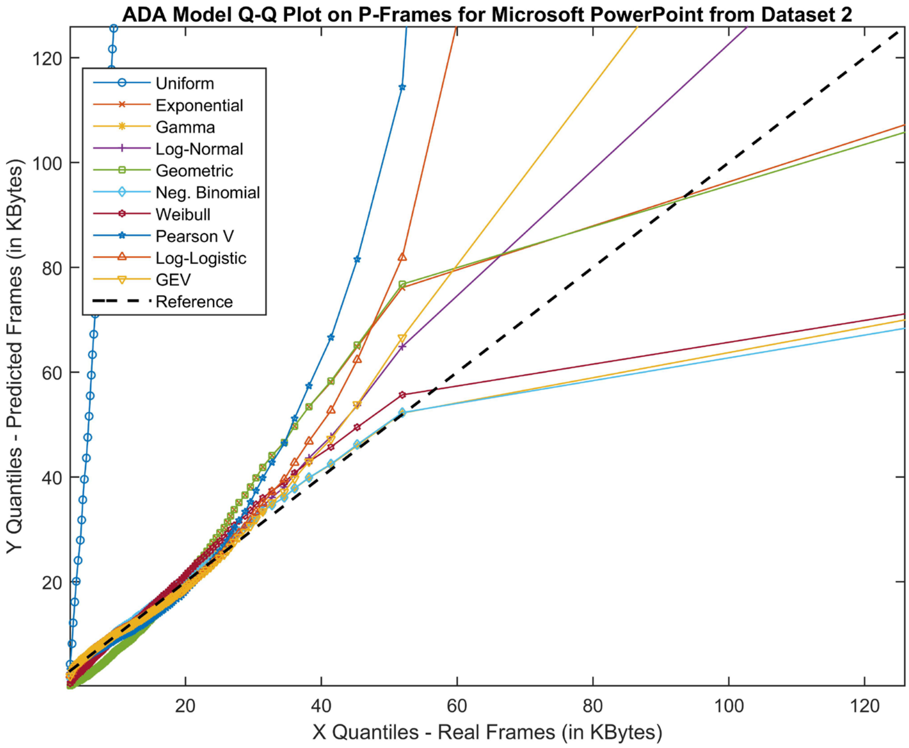


Figure 5.19: Q-Q Plot for Dataset's 2 Microsoft PowerPoint P-Frames from ADA Model.

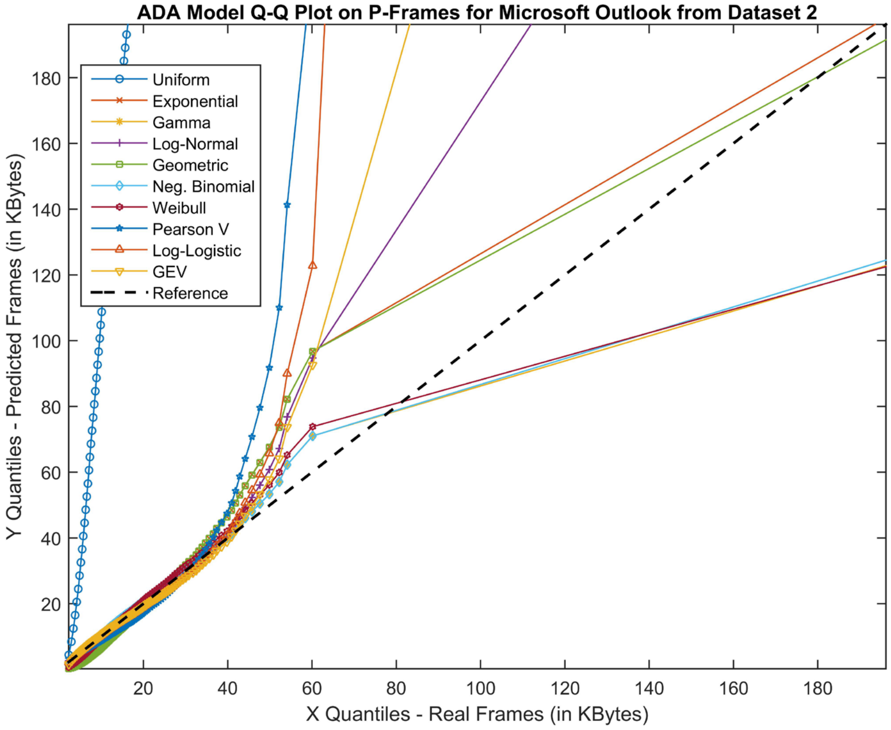


Figure 5.20: Q-Q Plot for Dataset's 2 Microsoft Outlook P-Frames from ADA Model.

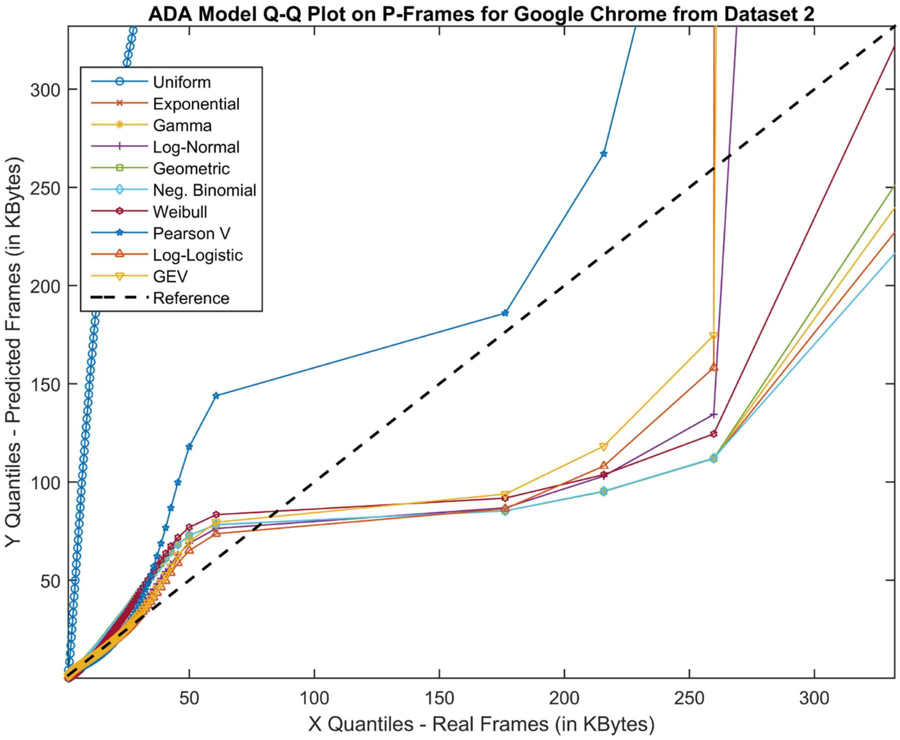


Figure 5.21: Q-Q Plot for Dataset's 2 Google Chrome P-Frames from ADA Model.

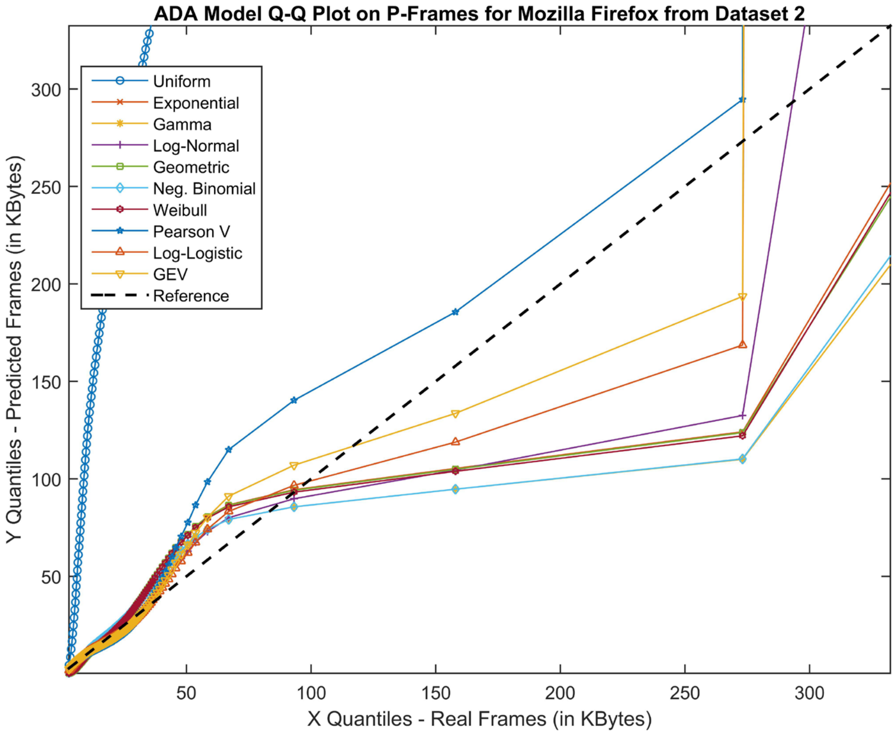


Figure 5.22: Q-Q Plot for Dataset's 2 Mozilla Firefox P-Frames from ADA Model.

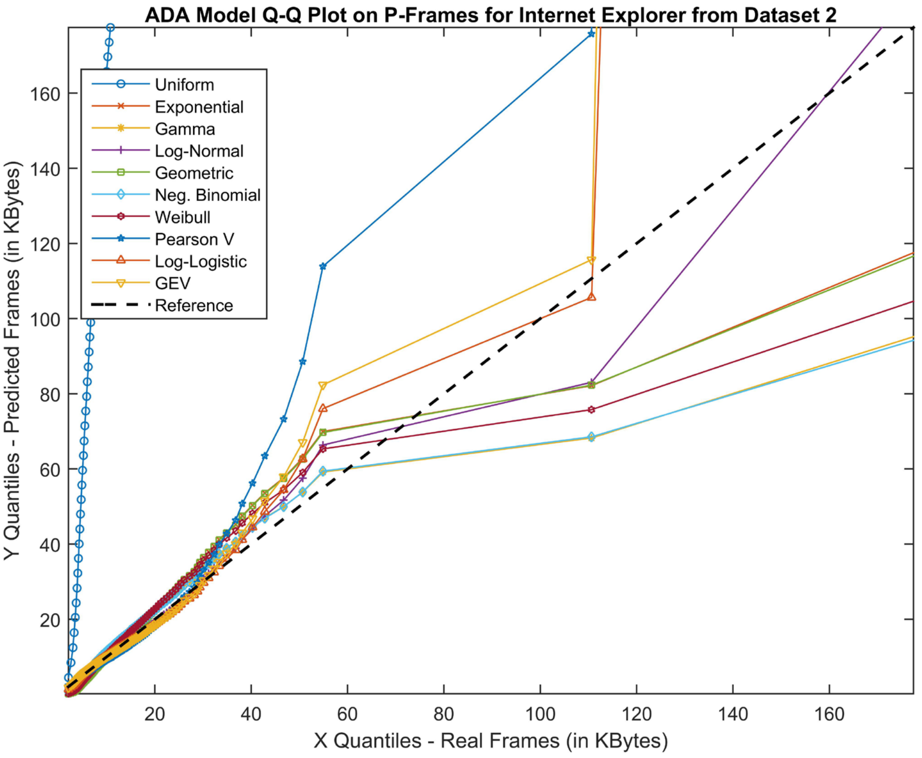


Figure 5.23: Q-Q Plot for Dataset's 2 Internet Explorer P-Frames from ADA Model.

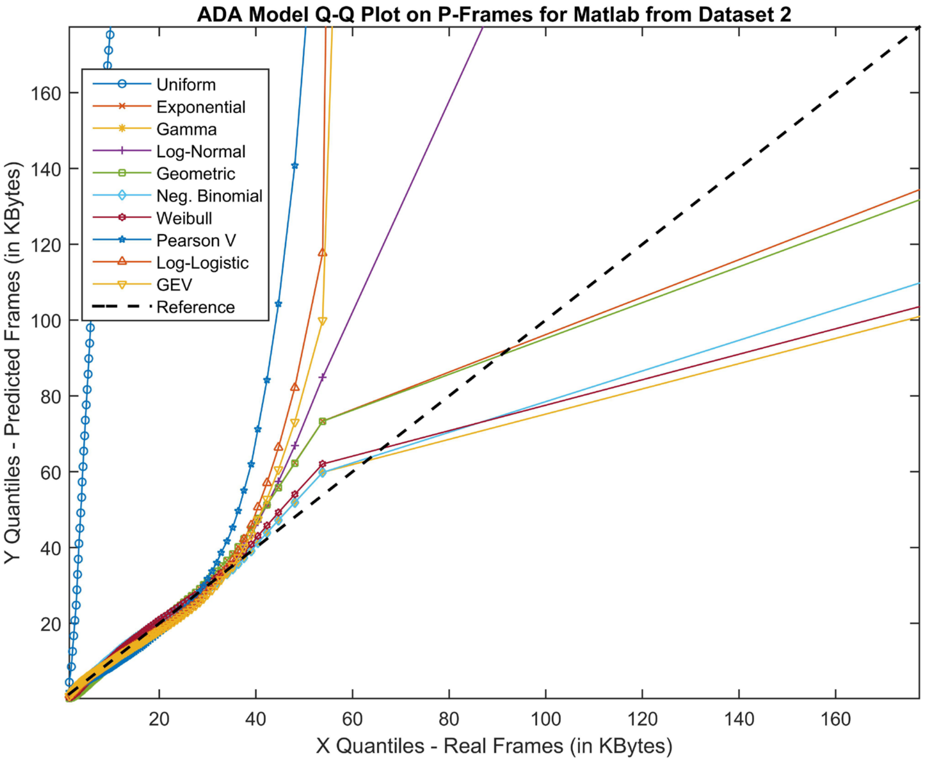


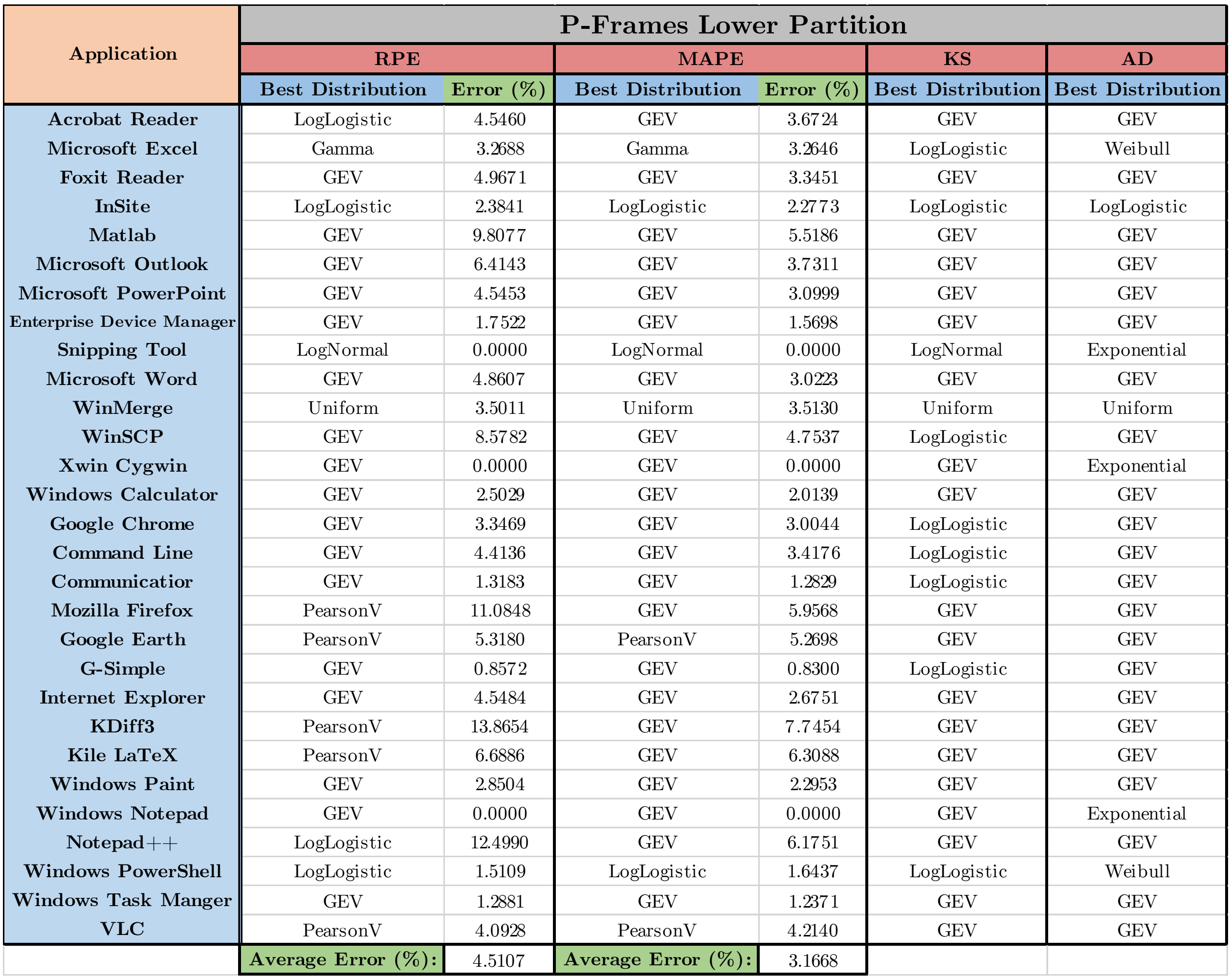
Figure 5.24: Q-Q Plot for Dataset's 2 Matlab P-Frames from ADA Model.

Figures (5.17) to (5.24) depict the Q-Q Plots of P-Frames for the eight major applications selected from Dataset 2. We plotted the Q-Q plots only for the P-Frames, because the amount of I-Frames in those applications was not sufficient to calculate the minimum amount of 100 quantiles.

As we can see from the Figures, the Q-Q Plots confirm again the best distribution fits for these applications, as indicated by the RPE and MAPE results in Table (5.4).

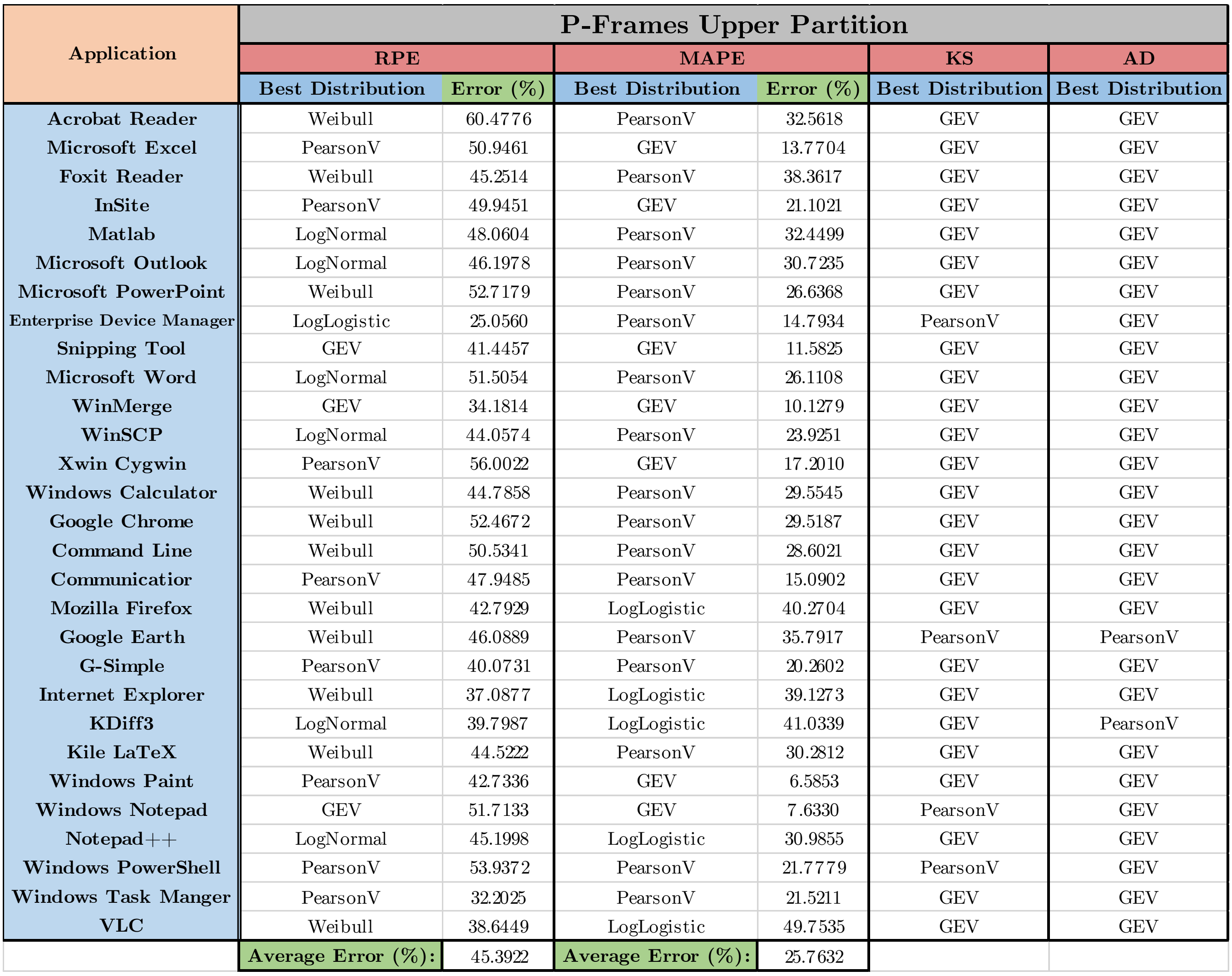
Additionally, the ADA model is shown to provide a competent model for most values of the P-Frames of Dataset 2, due to the fact that the distributions' curves lie around the Reference Line and they deviate only for the higher quantiles (right hand tail).

In the next four Tables (5.5) - (5.8), we present the results of the ADA model for the P-Frames of Dataset 1 and Dataset 2 with the modification of dividing P-Frames in Lower Partition and Upper Partition, as described in the Subsection 5.1.1.



Application	P-Frames Lower Partition					
	RPE		MAPE		KS	AD
	Best Distribution	Error (%)	Best Distribution	Error (%)	Best Distribution	Best Distribution
Acrobat Reader	LogLogistic	4.5460	GEV	3.6724	GEV	GEV
Microsoft Excel	Gamma	3.2688	Gamma	3.2646	LogLogistic	Weibull
Foxit Reader	GEV	4.9671	GEV	3.3451	GEV	GEV
InSite	LogLogistic	2.3841	LogLogistic	2.2773	LogLogistic	LogLogistic
Matlab	GEV	9.8077	GEV	5.5186	GEV	GEV
Microsoft Outlook	GEV	6.4143	GEV	3.7311	GEV	GEV
Microsoft PowerPoint	GEV	4.5453	GEV	3.0999	GEV	GEV
Enterprise Device Manager	GEV	1.7522	GEV	1.5698	GEV	GEV
Snipping Tool	LogNormal	0.0000	LogNormal	0.0000	LogNormal	Exponential
Microsoft Word	GEV	4.8607	GEV	3.0223	GEV	GEV
WinMerge	Uniform	3.5011	Uniform	3.5130	Uniform	Uniform
WinSCP	GEV	8.5782	GEV	4.7537	LogLogistic	GEV
Xwin Cygwin	GEV	0.0000	GEV	0.0000	GEV	Exponential
Windows Calculator	GEV	2.5029	GEV	2.0139	GEV	GEV
Google Chrome	GEV	3.3469	GEV	3.0044	LogLogistic	GEV
Command Line	GEV	4.4136	GEV	3.4176	LogLogistic	GEV
Communicatior	GEV	1.3183	GEV	1.2829	LogLogistic	GEV
Mozilla Firefox	PearsonV	11.0848	GEV	5.9568	GEV	GEV
Google Earth	PearsonV	5.3180	PearsonV	5.2698	GEV	GEV
G-Simple	GEV	0.8572	GEV	0.8300	LogLogistic	GEV
Internet Explorer	GEV	4.5484	GEV	2.6751	GEV	GEV
KDiff3	PearsonV	13.8654	GEV	7.7454	GEV	GEV
Kile LaTeX	PearsonV	6.6886	GEV	6.3088	GEV	GEV
Windows Paint	GEV	2.8504	GEV	2.2953	GEV	GEV
Windows Notepad	GEV	0.0000	GEV	0.0000	GEV	Exponential
Notepad++	LogLogistic	12.4990	GEV	6.1751	GEV	GEV
Windows PowerShell	LogLogistic	1.5109	LogLogistic	1.6437	LogLogistic	Weibull
Windows Task Manger	GEV	1.2881	GEV	1.2371	GEV	GEV
VLC	PearsonV	4.0928	PearsonV	4.2140	GEV	GEV
	Average Error (%):	4.5107	Average Error (%):	3.1668		

Table 5.5: ADA model results for P-Frames Lower Partition over all applications of Dataset 1.



Application	P-Frames Upper Partition					
	RPE		MAPE		KS	AD
	Best Distribution	Error (%)	Best Distribution	Error (%)	Best Distribution	Best Distribution
Acrobat Reader	Weibull	60.4776	PearsonV	32.5618	GEV	GEV
Microsoft Excel	PearsonV	50.9461	GEV	13.7704	GEV	GEV
Foxit Reader	Weibull	45.2514	PearsonV	38.3617	GEV	GEV
InSite	PearsonV	49.9451	GEV	21.1021	GEV	GEV
Matlab	LogNormal	48.0604	PearsonV	32.4499	GEV	GEV
Microsoft Outlook	LogNormal	46.1978	PearsonV	30.7235	GEV	GEV
Microsoft PowerPoint	Weibull	52.7179	PearsonV	26.6368	GEV	GEV
Enterprise Device Manager	LogLogistic	25.0560	PearsonV	14.7934	PearsonV	GEV
Snipping Tool	GEV	41.4457	GEV	11.5825	GEV	GEV
Microsoft Word	LogNormal	51.5054	PearsonV	26.1108	GEV	GEV
WinMerge	GEV	34.1814	GEV	10.1279	GEV	GEV
WinSCP	LogNormal	44.0574	PearsonV	23.9251	GEV	GEV
Xwin Cygwin	PearsonV	56.0022	GEV	17.2010	GEV	GEV
Windows Calculator	Weibull	44.7858	PearsonV	29.5545	GEV	GEV
Google Chrome	Weibull	52.4672	PearsonV	29.5187	GEV	GEV
Command Line	Weibull	50.5341	PearsonV	28.6021	GEV	GEV
Communicatior	PearsonV	47.9485	PearsonV	15.0902	GEV	GEV
Mozilla Firefox	Weibull	42.7929	LogLogistic	40.2704	GEV	GEV
Google Earth	Weibull	46.0889	PearsonV	35.7917	PearsonV	PearsonV
G-Simple	PearsonV	40.0731	PearsonV	20.2602	GEV	GEV
Internet Explorer	Weibull	37.0877	LogLogistic	39.1273	GEV	GEV
KDiff3	LogNormal	39.7987	LogLogistic	41.0339	GEV	PearsonV
Kile LaTeX	Weibull	44.5222	PearsonV	30.2812	GEV	GEV
Windows Paint	PearsonV	42.7336	GEV	6.5853	GEV	GEV
Windows Notepad	GEV	51.7133	GEV	7.6330	PearsonV	GEV
Notepad++	LogNormal	45.1998	LogLogistic	30.9855	GEV	GEV
Windows PowerShell	PearsonV	53.9372	PearsonV	21.7779	PearsonV	GEV
Windows Task Manger	PearsonV	32.2025	PearsonV	21.5211	GEV	GEV
VLC	Weibull	38.6449	LogLogistic	49.7535	GEV	GEV
	Average Error (%):	45.3922	Average Error (%):	25.7632		

Table 5.6: ADA model results for P-Frames Upper Partition over all applications of Dataset 1.

Application	P-Frames Lower Partition					
	RPE		MAPE		KS	AD
	Best Distribution	Error (%)	Best Distribution	Error (%)	Best Distribution	Best Distribution
Acrobat Reader	GEV	14.5895	GEV	13.0332	LogLogistic	GEV
Microsoft Excel	GEV	4.3283	GEV	3.6577	LogLogistic	GEV
Foxit Reader	LogLogistic	19.2628	LogNormal	9.7199	LogLogistic	LogNormal
Matlab	Gamma	7.8179	LogNormal	5.9948	LogNormal	LogNormal
Microsoft Outlook	NegBinomial	6.6261	Gamma	5.2884	Gamma	Gamma
Microsoft PowerPoint	NegBinomial	4.6098	NegBinomial	4.4387	Gamma	Gamma
Enterprise Device Manager	Weibull	8.5183	NegBinomial	7.3615	Weibull	Gamma
Snipping Tool	Weibull	4.8249	Weibull	7.7939	Weibull	Weibull
Microsoft Word	LogNormal	11.5816	LogNormal	4.5693	LogNormal	LogNormal
WinMerge	NegBinomial	6.9768	NegBinomial	6.8697	LogNormal	LogNormal
WinRAR	Weibull	8.6363	NegBinomial	7.8179	Weibull	Gamma
Xwin Cygwin	GEV	2.6347	GEV	4.0272	GEV	GEV
Windows Calculator	LogNormal	12.0559	GEV	9.3476	PearsonV	PearsonV
Google Chrome	LogLogistic	21.9746	LogLogistic	7.2871	LogLogistic	LogLogistic
Command Line	Geometric	13.3415	LogNormal	11.1278	LogNormal	LogNormal
Mozilla Firefox	Weibull	5.5522	Gamma	5.0706	Weibull	Gamma
IrfanView	LogLogistic	15.9627	LogNormal	7.9550	LogNormal	LogNormal
Internet Explorer	LogNormal	8.9016	LogNormal	4.4719	LogNormal	LogNormal
KDiff3	Weibull	7.4101	Weibull	9.6476	Weibull	Weibull
Windows Paint	NegBinomial	4.7594	GEV	4.8238	GEV	GEV
Windows Notepad	PearsonV	13.5467	GEV	8.4040	GEV	GEV
Notepad++	NegBinomial	8.2541	Gamma	6.2593	LogNormal	Gamma
Windows PowerShell	GEV	7.7399	NegBinomial	42.3303	GEV	GEV
Windows Task Manger	Weibull	6.2510	LogNormal	8.3526	Gamma	Gamma
VLC	GEV	17.6615	LogLogistic	8.1691	LogLogistic	GEV
VMware Player	LogNormal	34.4994	LogLogistic	27.1390		PearsonV
	Average Error (%):	10.7045	**Average Error (%):**	9.2676		

Table 5.7: ADA model results for P-Frames Lower Partition over all applications of Dataset 2.

Application	P-Frames Upper Partition					
	RPE		MAPE		KS	AD
	Best Distribution	Error (%)	Best Distribution	Error (%)	Best Distribution	Best Distribution
Acrobat Reader	GEV	0.3133	GEV	0.3052	GEV	GEV
Microsoft Excel	GEV	2.3460	GEV	2.7062	LogLogistic	Uniform
Foxit Reader	Gamma	0.2139	GEV	0.2121	GEV	GEV
Matlab	PearsonV	0.2701	PearsonV	0.2696	GEV	GEV
Microsoft Outlook	GEV	0.1823	GEV	0.1798	GEV	GEV
Microsoft PowerPoint	LogNormal	0.9437	LogNormal	0.9512	Gamma	PearsonV
Enterprise Device Manager	LogLogistic	24.0784	LogLogistic	9.4353	GEV	GEV
Snipping Tool	PearsonV	23.6576	GEV	10.6303	GEV	GEV
Microsoft Word	GEV	0.1244	GEV	0.1221	GEV	GEV
WinMerge	PearsonV	34.1166	PearsonV	23.8990	GEV	GEV
WinRAR	LogNormal	20.3945	PearsonV	21.0019	GEV	GEV
Xwin Cygwin	Weibull	14.0928	Gamma	15.6696	LogLogistic	LogLogistic
Windows Calculator	Uniform	2.4916	LogNormal	2.4801	GEV	GEV
Google Chrome	GEV	0.1649	GEV	0.1633	GEV	GEV
Command Line	NegBinomial	8.5865	NegBinomial	9.9478	GEV	Weibull
Mozilla Firefox	GEV	1.4995	GEV	1.7232	GEV	GEV
IrfanView	LogLogistic	0.1521	GEV	0.1506	LogLogistic	GEV
Internet Explorer	PearsonV	0.4630	PearsonV	0.4677	PearsonV	GEV
KDiff3	Uniform	7.5732	Gamma	14.7007	LogLogistic	GEV
Windows Paint	Uniform	3.4372	Uniform	3.4720	LogLogistic	Uniform
Windows Notepad	Weibull	14.9622	PearsonV	17.5369	GEV	GEV
Notepad++	GEV	0.1505	GEV	0.1487	GEV	GEV
Windows PowerShell	LogLogistic	10.5642	LogLogistic	11.1121	GEV	GEV
Windows Task Manger	Exponential	21.2228	PearsonV	18.1400	GEV	GEV
VLC	PearsonV	0.3256	GEV	0.3154	GEV	GEV
VMware Player	GEV	0.3872	GEV	0.3738	GEV	GEV
	Average Error (%):	7.4121	Average Error (%):	6.3890		

Table 5.8: ADA model results for P-Frames Upper Partition over all applications of Dataset 2.

As we can conclude from the results in Table (5.7) and Table (5.8), the partitioning technique works well for the P-Frames of Dataset 2 and the errors that we receive (RPE and MAPE) are less or at worst equal to the RPE and MAPE of the original approach in Table (5.4).

This is not the case, however, for the results of P-Frames in Dataset 1. As shown in Tables (5.5) and (5.6), the partitioning technique works well for the P-Frames in the Lower Partition but the P-Frames' modeling in the Upper Partition leads to high RPE and MAPE errors.

Finally, we should mention that we also implemented the ADA model on both datasets without any separation of frames into I and P. As expected, the large differences between I and P frames' sizes lead to much larger errors, especially for Dataset 1 (RPE=89.24% and MAPE=17.11%). Hence, this approach cannot be used, despite its simplicity.

In summary, the ADA model is a useful first approach for modeling our data. It achieves RPE and MAPE results around 8% for Dataset 1 and 5% for Dataset 2, on average, in the case of I-Frames. However, it fails to model the P-Frames (especially of Dataset 1) with satisfactory results. Our goal is to find an accurate common model for both datasets. Hence, we continued our work with the models presented in the next sections.

5.2 Gamma Beta Autoregressive Model

The results presented in Section 5.1 revealed that no single distribution can provide the best fit for each application. Still, the distribution fit that provided the overall lowest RPE and MAPE when used for all applications was that of the Gamma distribution (≈12% and 9% for the I-Frames and P-Frames Lower Partition of Dataset 1; ≈8%, 15% and 11% for the I-Frames, P-Frames Lower Partition and P-Frames Upper Partition of Dataset 2). Still, the RPE for the P-Frames Upper Partition of Dataset 1 is 67% and the respective MAPE is 116%.

The good performance of the Gamma fit (with the latter exception) led us to implement and evaluate the Gamma Beta Autoregressive (GBAR) Model, which has been proposed previously, as a source model for VBR encoding of videoconferences by D. Heyman *et al.* [13]. 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.

5.2.1 Model Analysis

Towards defining the GBAR model, let Ga(β, λ) denote a random variable with a gamma distribution with shape parameter β and scale parameter λ, that is, the density function is

$$f_G(t) = \frac{\lambda(\lambda t)^{\beta}}{\Gamma(\beta+1)} e^{-\lambda t},\ t > 0. \tag{5.1}$$

Similarly, let Be(p, q) denote a random variable with a beta distribution with parameters p and q, that is, with density function

$$f_B(t) = \frac{\Gamma(p+q)}{\Gamma(p+1)\,\Gamma(q+1)}\, t^{p-1}\,(1 - t)^{q-1},\ 0 < t < 1 \tag{5.2}$$

where p and q are both larger than -1. The GBAR(1) model is based on two well-known results: the sum of independent Ga(α, λ) and Ga(β, λ) random variables is a Ga(α + β, λ) random variable, and the product of independent Be(α, β – α) and

Ga(β, λ) random variables is a Ga(α, λ) random variable. Thus, if X_{n-1} is Ga(β, λ), A_n is Be(α, β − α), and B_n is Ga(β − α, λ), and these three are mutually independent, then

$$X_n = A_n X_{n-1} + B_n \tag{5.3}$$

defines a stationary stochastic process $\{X_n\}$ with a marginal Ga(β, λ) distribution. Furthermore, the autocorrelation function of this process is given by

$$r(k) = \left(\frac{\alpha}{\beta}\right)^k, \; k = 0, 1, 2, \ldots \tag{5.4}$$

The process defined by (5.3) is called the GBAR(1) process. Since the current value is determined by only one previous value, this is an autoregressive process of order 1. We also experimented with "tweaking" the model and using an autoregressive process of order 2, but our results (which will be presented in Subsection 5.2.2) did not improve with this change.

Simulating the GBAR process only requires the ability to simulate independent and identically distributed gamma and beta random variables. The parameters β and λ can be estimated from the mean and variance of marginal distribution of the data as follows. The mean and variance of a Ga(β, λ) distribution are β/λ and β/λ^2, respectively. Let m and v be the mean and variance of the data. Then equating moments yields the estimates

$$\hat{\lambda} = \frac{m}{v} \tag{5.5}$$

and

$$\hat{\beta} = \frac{m}{v^2} \tag{5.6}$$

The parameter α can be estimated from (5.4). We used the MLE method to estimate β and λ. The GBAR process is used as a source model by generating non-integer values from Equation (5.3), and then rounding the generated values to the nearest integer.

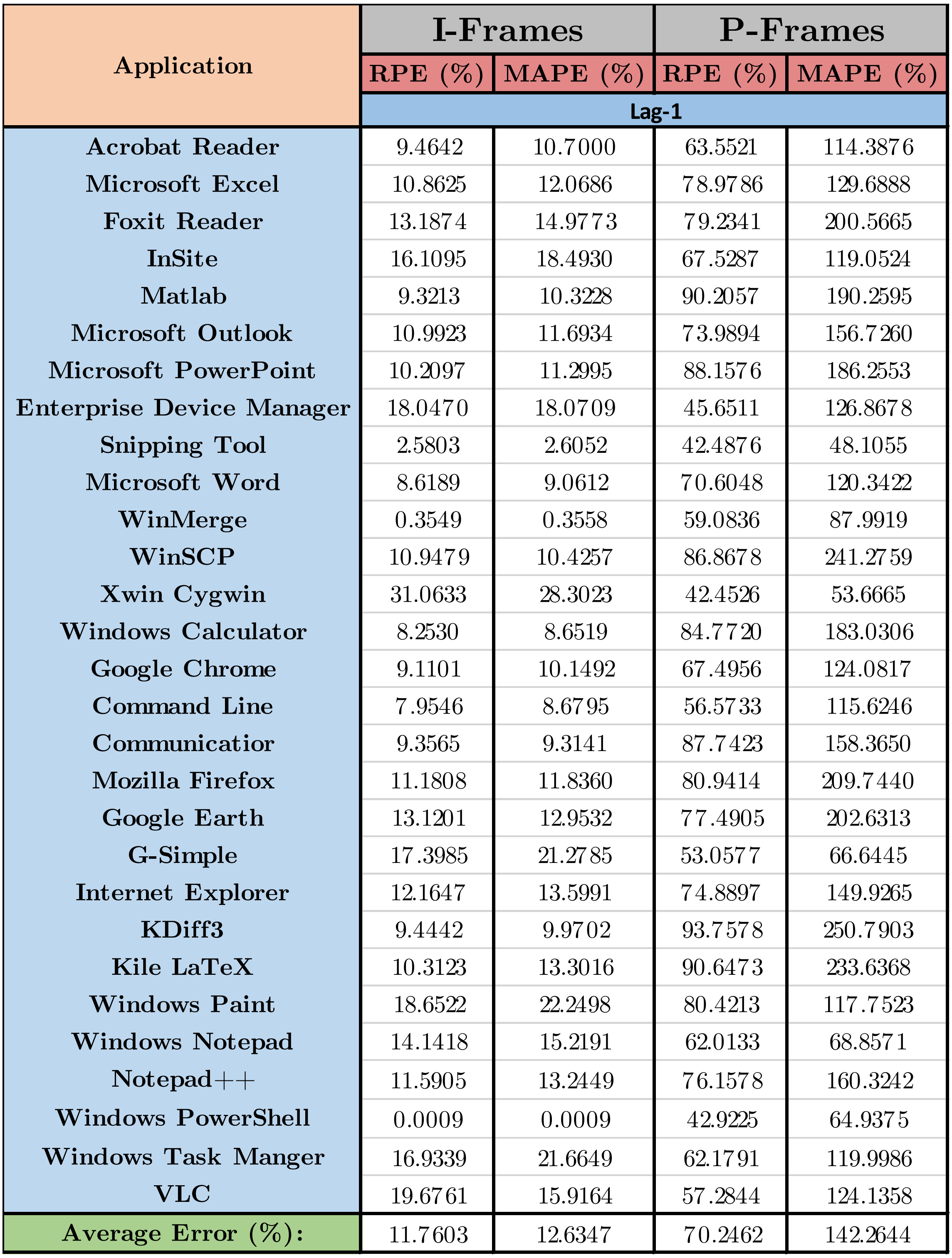
At this point, we need to point out an obstacle that we have faced and how we managed to overcome it. Inside a trace with video traffic records of a day, we have records corresponding to a variety of applications that the user was working with, during that day. These records are not consistent (i.e., do not correspond to the same application for a continuous period of time) but correspond to different applications depending on the user's behavior (as the user closes and switches between different applications during the day) and for that reason we observe time interrupts in the records of a specific application. In our case, the GBAR model can be continuously applied on a video traffic trace of a specific application after an interrupt, only when the usage scenario remains the same (i.e., the user minimizes the application and does not close it). During the preprocessing phase, where we separate the trace of a specific day per application, those interrupts occur and we do not know for which of those interrupts the user had closed the application (in order to apply the GBAR model separately, as the usage scenario changes) or had minimized it or switched to a different one (in order to continue applying the GBAR model, as the usage scenario remains the same).

Hence, we had to define a time threshold which, when exceeded by an interrupt, will signify that the user closed the application; therefore, the GBAR model will be applied just until the interrupt. In order to define this threshold, we calculated the trace's autocorrelation before and after every interrupt greater than 60, 90, 120, 150 and 180 seconds for every distinct application of our datasets, for a window of 1 GOP + 1 Frame for Dataset 1 and 61 Frames for Dataset 2 and for lag-1 and lag-2. The autocorrelation's results referred to the same time interval and application were averaged together, in order to take an overall result.

The results showed us that for both datasets the largest autocorrelation exists before and after 60-second interrupts than with any other tested time interval. For that reason, we set a time threshold of 60 seconds. When this is exceeded by an interrupt, the user is assumed to have closed the application.

5.2.2 Model Results

In this subsection, we evaluate the GBAR model by presenting the results from our tests. We have ran our model for every distinct application of Dataset 1 and Dataset 2, and separately for I and P frames.



Application	I-Frames		P-Frames	
	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
Enterprise 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 5.9: GBAR model results over all applications of Dataset 1.

Application	I-Frames		P-Frames	
	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
Enterprise 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

Table 5.10: GBAR model results over all applications of Dataset 2.

From the above two tables, we observe that the GBAR model offers good accuracy in the modeling of I-Frames (but worse from ADA model). Still, it fails clearly in the modeling of P-Frames.

The reason for the failure of the model in the case of P-Frames can be understood by studying the autocorrelation values in Table (5.11). The lag-1 autocorrelation of P-Frames is ≈0.3% for the 1st Dataset and for which the RPE and MAPE are very high. In Dataset 2, where the lag-1 autocorrelation is clearly higher (≈0.6) the GBAR model achieves lower errors for P-Frames than in Dataset

1. In summary, the GBAR model is shown to underperform in terms of accuracy when compared with ADA model.

Application	Dataset 1	Dataset 2
	Autocorrelation Lag-1	
Acrobat Reader	0.3580	0.6778
Microsoft Excel	0.4260	0.5460
Foxit Reader	0.3181	0.6008
InSite	0.3857	
Matlab	0.2022	0.7025
Microsoft Outlook	0.2886	0.6692
Microsoft PowerPoint	0.1858	0.6846
Corporate Device Manager	0.2759	0.6424
Snipping Tool	0.4847	0.7144
Microsoft Word	0.1620	0.6507
WinMerge	0.3181	0.5677
WinRAR		0.5812
WinSCP	0.0138	
Xwin Cygwin	0.2088	0.4916
Windows Calculator	0.5042	0.7807
Google Chrome	0.4459	0.6919
Command Line	0.3704	0.6303
Communicatior	0.2289	
Mozilla Firefox	0.3081	0.6236
Google Earth	0.5859	
G-Simple	0.3669	
IrfanView		0.6422
Internet Explorer	0.3598	0.6802
KDiff3	0.2123	0.4986
Kile LaTeX	0.2225	
Windows Paint	0.3987	0.6760
Windows Notepad	0.3275	0.4653
Notepad++	0.1716	0.6959
Windows PowerShell	0.6570	0.6736
Windows Task Manger	0.1488	0.4324
VLC	0.3123	0.6266
VMware Player		0.5755
Average Autocorrelation:	0.3189	0.6239

Table 5.11: Autocorrelation of P-Frames for lag-1 and for both datasets.

5.3 Linear Regression Model

Linear Regression (LR) is a statistical approach for modeling the linear relationship between a variable Y and one or more explanatory variables X. The relationships are modeled using linear regression equations whose unknown model parameters are estimated from the data using a fitting method (the most common method used is the least squares error).

A LR based model has been proposed previously, for predicting the size of future B-Frames of MPEG-4 encoded video traffic by L. Lanfranchi and B. Bing [32]. Their model is based on the fact that the 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. So, they 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.

5.3.1 Model Analysis

We have followed a similar approach, in order to develop our own Linear Regression (LR) model, for predicting the sizes of future P-Frames of our video traffic in Dataset 1 and Dataset 2 (given the high accuracy of ADA model for I-Frames' size prediction, we focused on the more difficult problem of predicting P-Frames' size). 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 between the 59 P-Frames and the 1 I-Frame in a GOP for Dataset 1 (via Equation 5.7, below), the autocorrelation (via Equation 5.8, below) between the 59 P-Frames in a GOP for Dataset 1 and the autocorrelation (5.7) only between the 60 P-Frames in a Window, if we are modeling for 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, the coefficient of correlation is calculated as

$$\rho_{X,Y} = \frac{E(XY) - \overline{X}\overline{Y}}{\sigma_X \sigma_Y} \tag{5.7}$$

and the autocorrelation is calculated as

$$\mathrm{r(k)} = \frac{\mathrm{E}[(\mathrm{X_m} - \bar{\mathrm{X}})(\mathrm{X_{m+k}} - \bar{\mathrm{X}})]}{\sigma_{\mathrm{X}}^2} \tag{5.8}$$

where m denotes the present frame and k the lag. Also, due to the fact that a GOP (for Dataset 1) or a Window (for Dataset 2) can correspond to more than one application (because users can switch or terminate applications unpredictably) or GOPs/Windows with less than 60 frames in size can occur (because recording stops immediately, when the host machine goes into sleep, hibernation or it shuts down), we had to preprocess our traces and keep only "clean" GOPs and Windows (i.e., those that refer to only one application and have a size of 60 frames exactly).

Then, 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. The equations have the format of (5.9) below

$$\mathrm{F_P} = \mathrm{a} \cdot \mathrm{F_{P\text{-}1}} + \mathrm{b} \cdot \mathrm{F_{P\text{-}2}} + \mathrm{c_P} \tag{5.9}$$

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

5.3.2 Model Application and Results

We have applied the LR model to the 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).

Below, we present graphically the correlation values among the I and P frames for Dataset 1 and the autocorrelation values for Dataset 2.

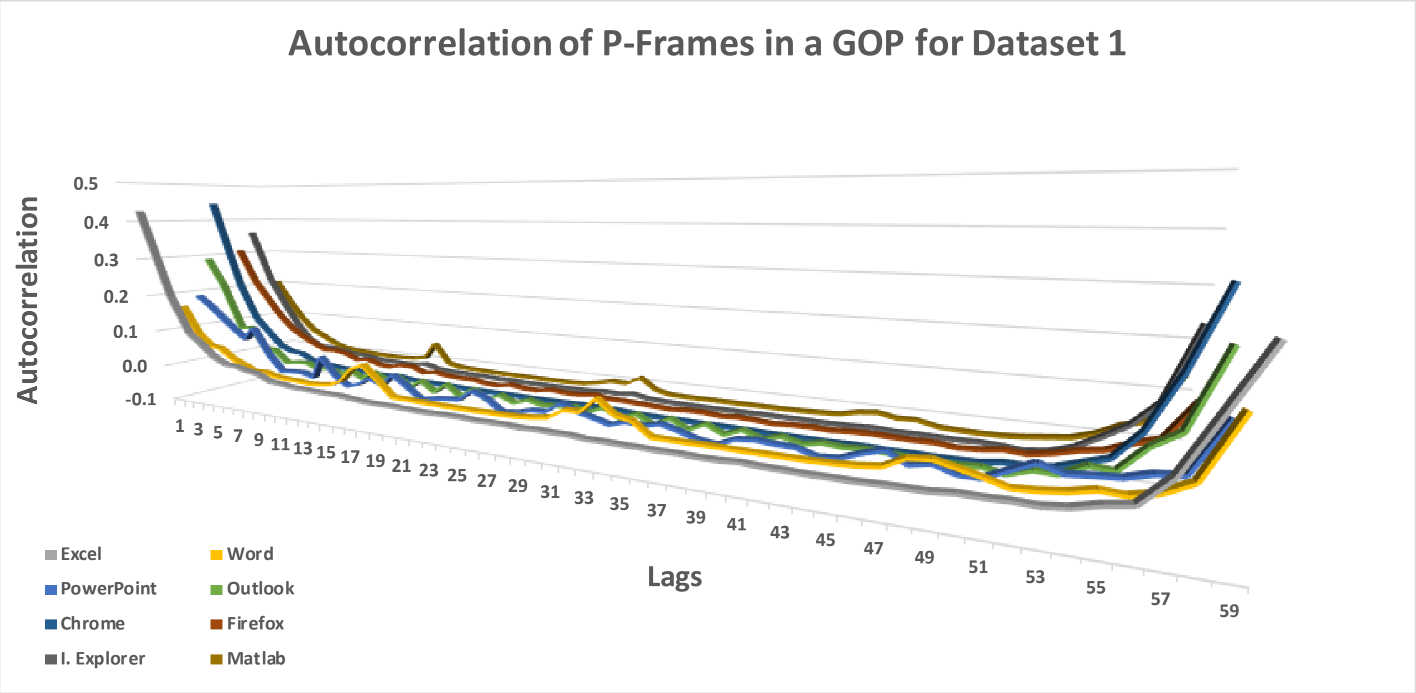


Figure 5.25: Autocorrelation of P-Frames in a GOP for Dataset 1.

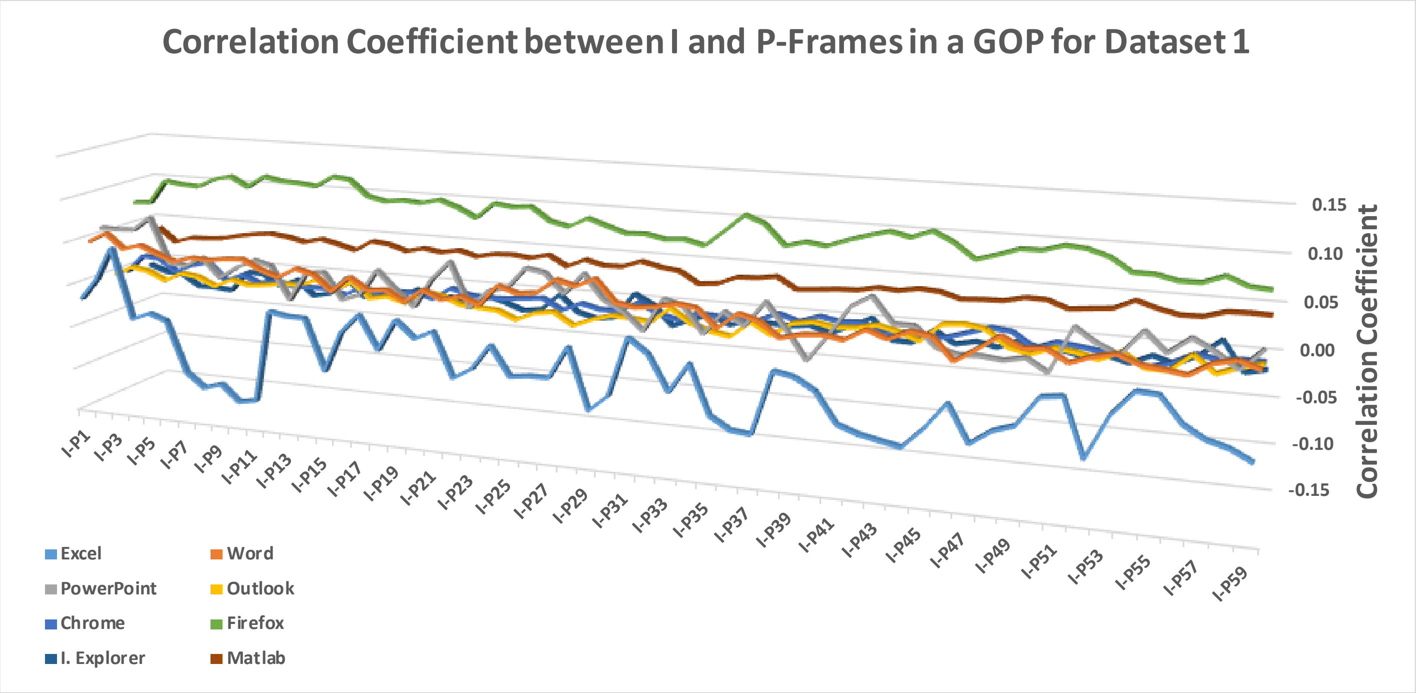


Figure 5.26: Correlation Coefficient for I and P-Frames in a GOP for Dataset 1.

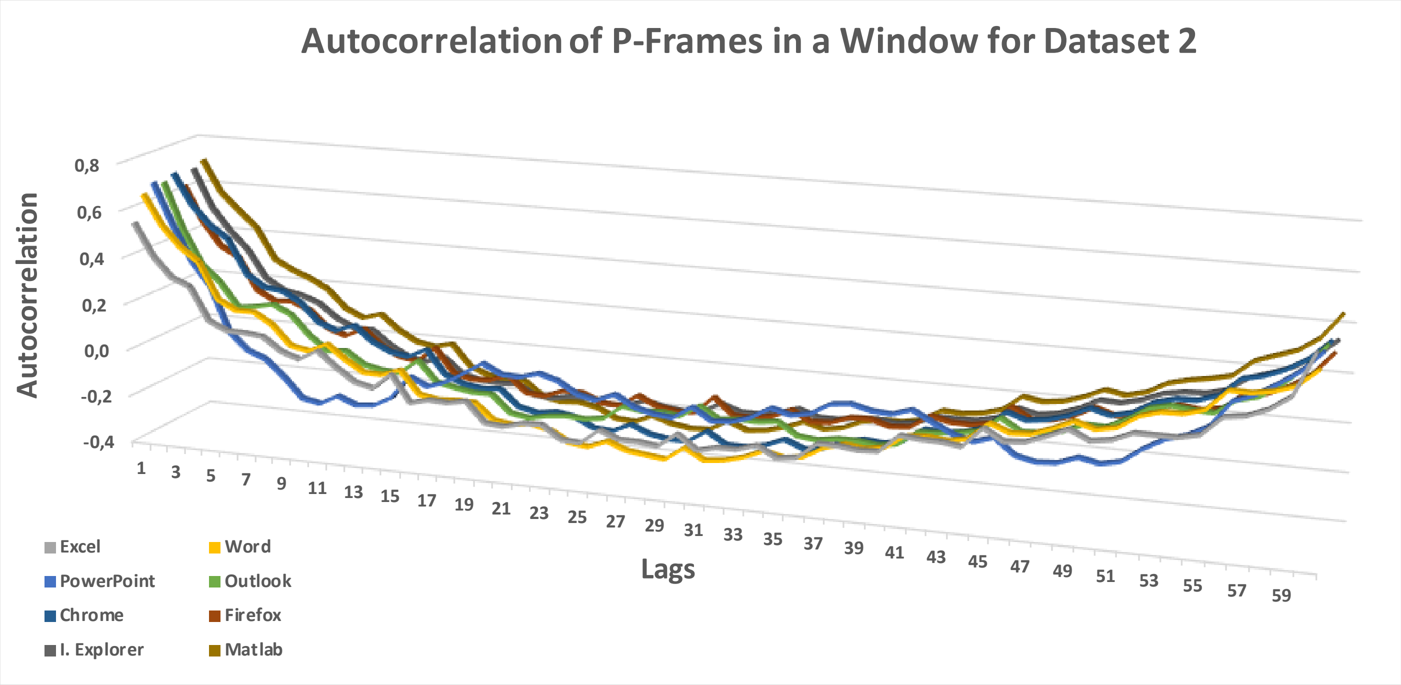


Figure 5.27: Autocorrelation of P-Frames in a Window for Dataset 2.

As shown in the figures above, the two most relevant frames in Dataset 1 are (N-1) and (N-2) P-Frames for PowerPoint, Firefox and Matlab and the (N-1) and (N-59) for Excel, Word, Outlook, Chrome and Internet Explorer. As for Dataset 2, the (N-1) and (N-2) P-Frames are the "closest" to frame N for every major application with an exception for Excel, where the closest are the (N-1) and (N-60) P-Frames. The above observations are also clear from the three following tables.

Application	Correlation Coefficient of I and P-Frames - Dataset 1							
	I vs P1	I vs P2	I vs P3	I vs P4	I vs P56	I vs P57	I vs P58	I vs P59
Microsoft Excel	-0.0172	0.0077	0.0459	-0.0361	-0.0375	-0.0509	-0.0571	-0.0686
Microsoft Word	0.0465	0.0572	0.0403	0.0464	0.0049	0.0189	0.0240	0.0164
Microsoft PowerPoint	0.0580	0.0574	0.0586	0.0744	0.0364	0.0241	0.0086	0.0310
Microsoft Outlook	-0.0029	0.0079	0.0041	-0.0051	0.0139	-0.0035	0.0050	0.0109
Google Chrome	-0.0042	0.0164	0.0149	0.0008	0.0085	0.0049	0.0069	0.0066
Mozilla Firefox	0.0760	0.0773	0.1048	0.1018	0.0760	0.0846	0.0755	0.0739
Internet Explorer	-0.0050	-0.0126	-0.0137	-0.0270	-0.0049	0.0140	-0.0182	-0.0130
Matlab	0.0365	0.0204	0.0268	0.0276	0.0306	0.0377	0.0383	0.0377

Table 5.12: Correlation Coefficient for I and P-Frames in a GOP for Dataset 1.

Application	Autocorrelation of P-Frames - Dataset 1							
	Lag-1	Lag-2	Lag-3	Lag-4	Lag-56	Lag-57	Lag-58	Lag-59
Microsoft Excel	**0.4260**	0.1987	0.0970	0.0700	-0.0048	0.0491	0.1566	**0.2657**
Microsoft Word	**0.1620**	0.0842	0.0532	0.0486	-0.0105	0.0016	0.0278	**0.1496**
Microsoft PowerPoint	**0.1858**	**0.1599**	0.1265	0.0951	0.0012	0.0150	0.0193	0.1232
Microsoft Outlook	**0.2886**	0.2163	0.0882	0.0922	-0.0062	0.0465	0.0780	**0.2297**
Google Chrome	**0.4459**	0.2131	0.1087	0.0701	-0.0009	0.0569	0.1688	**0.3232**
Mozilla Firefox	**0.3081**	**0.2146**	0.1623	0.1100	-0.0021	0.0181	0.0331	0.1086
Internet Explorer	**0.3598**	0.2183	0.1434	0.0759	0.0078	0.0375	0.0873	**0.2382**
Matlab	**0.2022**	**0.1426**	0.0891	0.0532	-0.0012	0.0224	0.0363	0.1361

Table 5.13: Autocorrelation of P-Frames in a GOP for Dataset 1.

Application	Autocorrelation of P-Frames - Dataset 2							
	Lag-1	Lag-2	Lag-3	Lag-4	Lag-57	Lag-58	Lag-59	Lag-60
Microsoft Excel	**0.5460**	0.4077	0.3234	0.2921	0.1697	0.2039	0.2611	**0.4387**
Microsoft Word	**0.6507**	**0.5207**	0.4351	0.3777	0.2368	0.2512	0.2794	0.3506
Microsoft PowerPoint	**0.6846**	**0.5095**	0.3594	0.2570	0.2061	0.2571	0.3140	0.3982
Microsoft Outlook	**0.6692**	**0.4721**	0.3251	0.2546	0.1765	0.2323	0.3054	0.4032
Google Chrome	**0.6919**	**0.5598**	0.4763	0.4230	0.2374	0.2699	0.3200	0.3968
Mozilla Firefox	**0.6236**	**0.4724**	0.3701	0.3264	0.1446	0.1814	0.2355	0.3292
Internet Explorer	**0.6802**	**0.5168**	0.4186	0.3404	0.2056	0.2369	0.2898	0.3582
Matlab	**0.7025**	**0.5649**	0.4906	0.4198	0.2546	0.2830	0.3385	0.4423

Table 5.14: Autocorrelation of P-Frames in a Window for Dataset 2.

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. These for Dataset 1 are:

<u>PowerPoint, Firefox and Matlab</u>

$\widehat{P_{1,t}} = a_1 \cdot P_{59,t\text{-}1} + b_1 \cdot P_{58,t\text{-}1} + c_1$

$\widehat{P_{2,t}} = a_2 \cdot P_{1,t} + b_2 \cdot P_{59,t\text{-}1} + c_2$

$\widehat{P_{3,t}} = a_3 \cdot P_{2,t} + b_3 \cdot P_{1,t} + c_3$

$\vdots$

$\widehat{P_{59,t}} = a_{59} \cdot P_{58,t} + b_{59} \cdot P_{57,t} + c_{59}$

<u>Excel, Word, Outlook, Chrome and Internet Explorer</u>

$\widehat{P_{1,t}} = a_1 \cdot P_{59,t\text{-}1} + b_1 \cdot P_{1,t\text{-}1} + c_1$

$\widehat{P_{2,t}} = a_2 \cdot P_{1,t} + b_2 \cdot P_{2,t\text{-}1} + c_2$

$\widehat{P_{3,t}} = a_3 \cdot P_{2,t} + b_3 \cdot P_{3,t\text{-}1} + c_3$

$\vdots$

$\widehat{P_{59,t}} = a_{59} \cdot P_{58,t} + b_{59} \cdot P_{59,t\text{-}1} + c_{59}$

where t denotes the current GOP and for Dataset 2 are:

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

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

Excel

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

where t denotes the current GOP. We present the results of the LR model for the eight major applications of Dataset 1 and Dataset 2, in Table (5.15). It is clear from the results that the LR model fails to predict the P-Frames' sizes for both datasets. The reason is the low correlation and autocorrelation values.

Application	Dataset 1		Dataset 2	
	RPE (%)	MAPE (%)	RPE (%)	MAPE (%)
Microsoft Excel	89.9809	61.6805	90.0309	83.1998
Microsoft Word	89.9981	53.9079	90.0015	82.7885
Microsoft PowerPoint	90.7187	31.0618	89.9994	84.1620
Microsoft Outlook	90.0110	37.9630	90.0413	82.6201
Google Chrome	89.9904	42.0403	90.0654	79.8477
Mozilla Firefox	91.8253	24.3955	90.0263	80.6783
Internet Explorer	89.9925	43.1415	90.1921	77.4014
Matlab	95.8905	72.5315	90.0315	78.3408
Average Error (%):	91.0509	45.8403	90.0486	81.1298

Table 5.15: LR model results for eight major applications of both datasets.

5.4 Markovian - Clustering Model

In [38], 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' traffic is not layered, we decided to use a conceptually similar approach with [38]. We developed a Markovian - Clustering (MC) model, in order to predict the sizes of P-Frames from Dataset 1 and Dataset 2.

We view the video trace sequence as a vector containing all the P-Frames' sizes. We place all these vector's elements as points on the 1-D plane and we then use the K-Means clustering algorithm [39] 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, Matlab in Dataset 2 contains over 5M P-Frames) and K-Means deals better with large datasets than other clustering algorithms (i.e., the Hierarchical clustering algorithm). The metric that we used for calculating the minimum distance of each point n from the m means vector, 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. In our case, we concluded after several experiments that the optimal number of clusters per tested application (i.e., the number of clusters that leads to the highest modeling accuracy) is the one depicted in Table (5.16).

Application	Dataset 1	Dataset 2
	# of Clusters	
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

Table 5.16: Optimal number of clusters for every tested application and for both datasets.

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 KxK elements, following the Equation (5.10) below

$$\mathrm{P_{i,j}} = \frac{\text{\# of jumps from state i to state j}}{\text{\# of jumps from state i}} \tag{5.10}$$

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

5.4.1 Jaccard Index -Infused MC Model

The MC model described in the previous subsection, performs clustering on 1-D data (our P-Frames) based on the actual size of every P-Frame. We tried another approach by employing the concept of the Jaccard Index.

The Jaccard Index [40], 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 intersection divided by the size of union of two sample sets, as depicted in Equation (5.11) below

$$\mathrm{J(A,B)} = \frac{|\mathrm{A} \cap \mathrm{B}|}{|\mathrm{A} \cup \mathrm{B}|} \tag{5.11}$$

where A and B denote the two sample sets. Jaccard Index is widely used in regionalization and species association analyses [41].

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 autocorrelation calculation (as it was shown to perform poorly). This approach is the use of the Jaccard Index. 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 (the 10% threshold is discussed in Subsection 5.4.2).

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. Figures (5.28) and (5.29) present this result graphically.

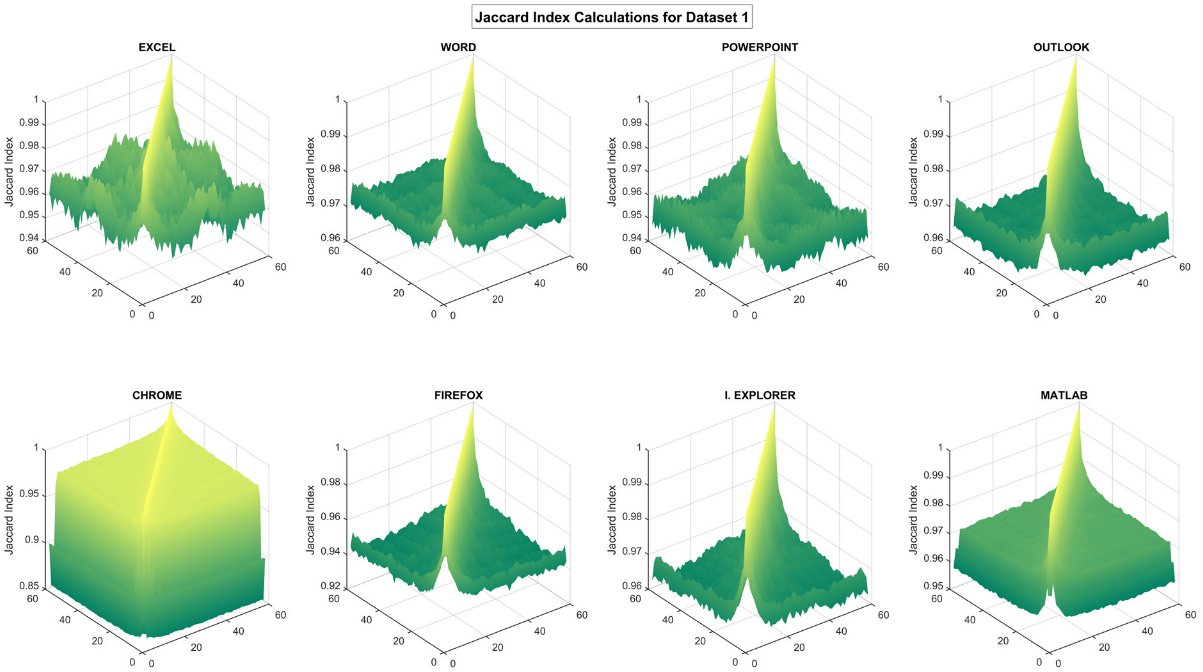


Figure 5.28: Jaccard Index calculations for all major applications of Dataset 1.

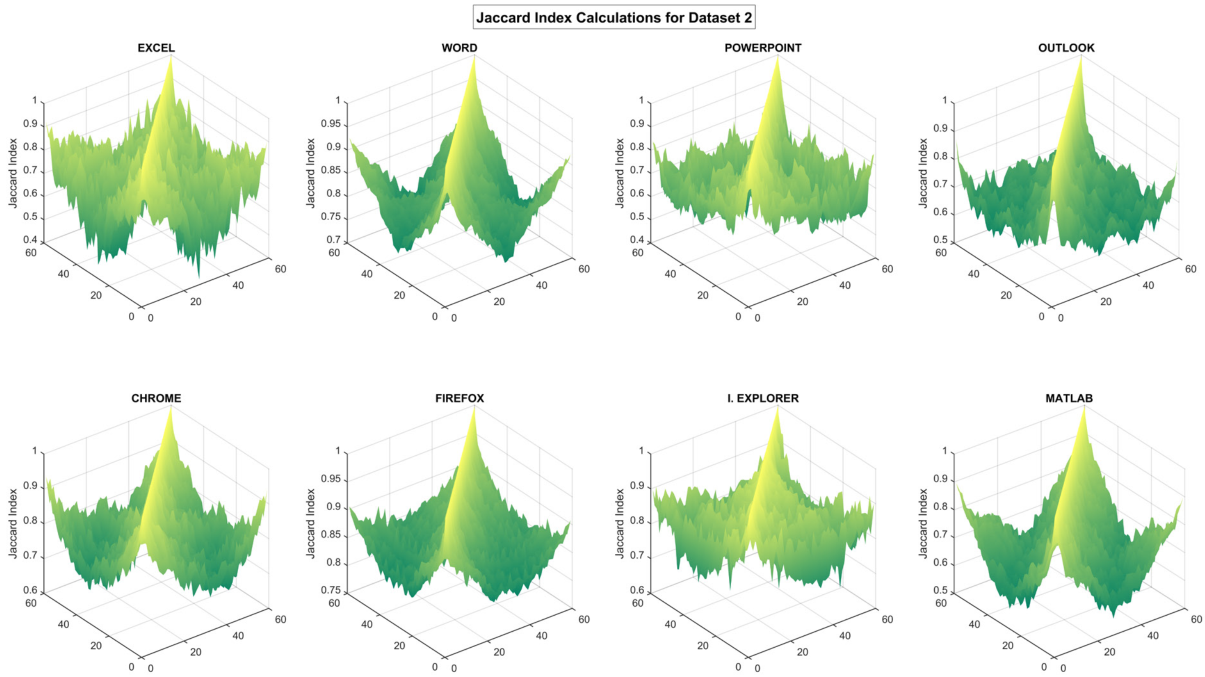


Figure 5.29: Jaccard Index calculations for all major applications of Dataset 2.

The x-axis and the y-axis in these figures represent the P-Frame position in a GOP or in a Window for Dataset 1 and Dataset 2 respectively and the z-axis represents the Jaccard Index value between two P-Frames defined by x and y. As shown in the figures, Jaccard Index has a value equals to 1 on the x=y line (because every frame has the same Jaccard Index with itself) and the Jaccard Index is getting 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, ... Here $R_c(t)$ denotes the frame size of the t^{th} P-Frame, $R_p(t)$ denotes the frame size of the P-Frame before $R_c(t)$ (i.e., $R_p(t) = R_c(t\text{-}1)$) 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.

Finally, 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 subsection.

5.4.2 Models 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) draws some fluctuations into our results. For this reason, we calculated the corresponding confidence intervals as described in Subsection 4.3.3. Every test has been executed 10 times and the results refer to 95% confidence intervals.

Table (5.17) presents the results of the MC model for Dataset 1 and Dataset 2, in terms of the model's accuracy in predicting P-Frames' sizes.

Application	Dataset 1		Dataset 2	
	RPE (%)	MAPE (%)	RPE (%)	MAPE (%)
Microsoft Excel	5.7350 ± 1.1315	5.6017 ± 0.0778	4.0178 ± 0.0547	5.1868 ± 0.0188
Microsoft Word	3.0343 ± 0.2638	2.1478 ± 0.0106	3.1309 ± 0.1943	2.1243 ± 0.0450
Microsoft PowerPoint	4.7942 ± 0.6676	4.0129 ± 0.0359	5.6611 ± 0.7642	2.6834 ± 0.0526
Microsoft Outlook	4.7768 ± 0.2074	2.9855 ± 0.0134	4.4029 ± 0.0254	3.4546 ± 0.0173
Google Chrome	2.7862 ± 0.0801	3.9144 ± 0.0107	3.1187 ± 0.1307	3.9555 ± 0.0313
Mozilla Firefox	2.7877 ± 0.2000	2.2959 ± 0.0793	2.0283 ± 0.0579	1.8999 ± 0.0302
Internet Explorer	6.7931 ± 0.8542	6.2865 ± 0.0261	2.8832 ± 0.1577	2.2355 ± 0.0704
Matlab	2.9384 ± 0.1023	2.4950 ± 0.0085	3.2073 ± 0.0065	3.7263 ± 0.0048
Average Error (%):	4.2057	3.7175	3.5563	3.1583

Table 5.17: MC model results for major applications of both datasets.

As shown from the results, the MC model succeeds in predicting the P-Frames' sizes for both datasets with high accuracy. The RPE and MAPE errors are below 5% for all applications, with an exception for Internet Explorer of Dataset 1, where we receive errors near 7%.

Table (5.18) 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.

Application	Dataset 1						Dataset 2					
	RPE (%)			MAPE (%)			RPE (%)			MAPE (%)		
Microsoft Excel	10.5580	±	2.5248	5.4934	±	0.3694	3.1309	±	0.0623	1.7473	±	0.0613
Microsoft Word	8.7742	±	0.8625	5.9972	±	0.2572	4.6322	±	0.0169	2.3566	±	0.0201
Microsoft PowerPoint	11.8895	±	1.3306	5.7706	±	0.2479	3.7885	±	0.0218	1.8679	±	0.0174
Microsoft Outlook	10.0115	±	1.1728	7.9895	±	0.1753	4.1970	±	0.0128	2.2591	±	0.0143
Google Chrome	6.9787	±	0.5986	4.3858	±	0.1025	3.2603	±	0.2130	3.0586	±	0.0428
Mozilla Firefox	7.4168	±	1.1337	3.2986	±	0.2020	2.2852	±	0.2262	1.6051	±	0.0354
Internet Explorer	10.2065	±	1.1802	6.9247	±	0.2967	5.5325	±	0.1881	2.2703	±	0.1083
Matlab	3.4727	±	0.3187	2.6927	±	0.0487	3.9366	±	0.0196	1.7462	±	0.0220
Average Error (%):	8.6635			5.3191			3.8454			2.1139		

Table 5.18: JIMC model results for major applications of both datasets.

As shown from the results, the JIMC model succeeds in predicting the P-Frames' sizes for both datasets with very high accuracy. We should mention that we have experimented with different values in the 2nd rule of the definition of a "neighboring frame" (Subsection 5.4.1); we have used values up to 20% difference in autocorrelation, with negligible difference 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, comparable 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) are stronger than in Dataset 1. The comparison between the two models is depicted in Figure (5.30)

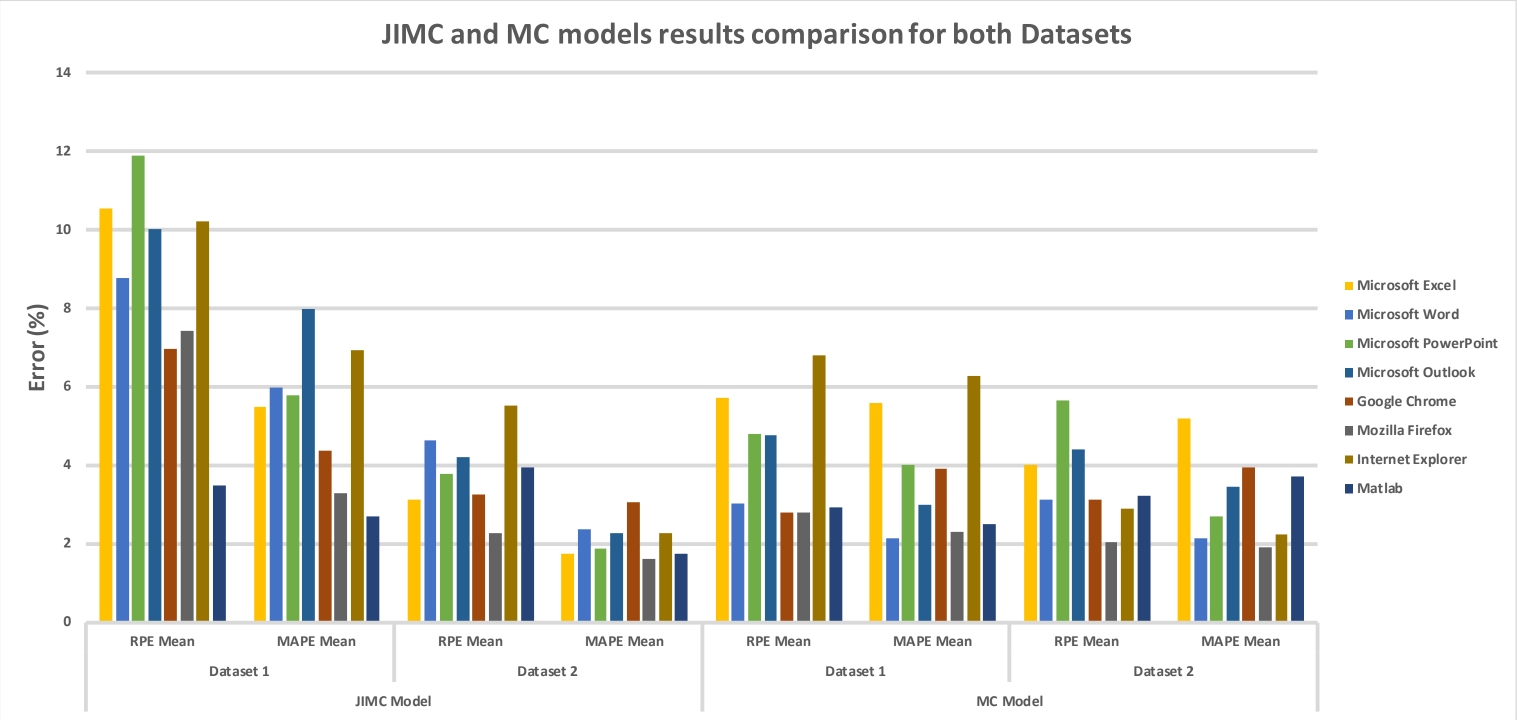


Figure 5.30: JIMC and MC models results comparison for both Datasets

6 Conclusions and Future Work

In this work, we have developed and tested four different 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 a simple approach, such as the Application and Distribution Aware model is capable to predict I-Frames video traffic with high accuracy, giving us RPE below 8% and MAPE below 9% for the 1st Dataset and RPE below 5% and MAPE below 6% for the 2nd Dataset on average. In addition, we showed 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 predict P-Frames video traffic, due to the poor autocorrelation that characterizes the kind of video traffic that we worked with.

We also proposed the Markovian - Clustering Model for P-Frames prediction, which we modified by incorporating the Jaccard Index, for the first time in the video traffic modeling literature. We have shown that the Markovian - Clustering model has excellent accuracy in P-Frames' sizes prediction for the 1st Dataset with RPE below 4.5% and MAPE below 4% on average and 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) with RPE below 4% and MAPE below 2.5% on average.

Given that smart mobile devices tend to replace computers on everyday usage, 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.

Bibliography

[1] D. Gilbert, "Microsoft wants to replace your PC with your smartphone," International Business Times, 30 April 2015. [Online]. Available: 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 March 2013. [Online]. Available: https://nakedsecurity.sophos.com/2013/03/14/devices-wozniak-infographic/.

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

[4] A. Lazaris, P. Koutsakis and M. Paterakis, "A new model for video traffic originating from multiplexed MPEG-4 videoconference streams," *Performance Evaluation 65,* pp. 51-70, 2008.

[5] M. Nomura, T. Fuji and N. Ohta, "Basic characteristics of variable rate video coding in ATM environment," *IEEE Journal on Selected Areas in Communications 7,* pp. 752-760, 1989.

[6] D. M. Lucantoni, M. F. Neuts and A. R. Reibman, "Methods for performance evaluation of VBR video traffic models," *IEEE/ACM Transactions on Networking 2,* pp. 176-180, 1994.

[7] D. P. Heyman, A. Tabatabai and T. V. Lakshman, "Statistical analysis and simulation study of video teleconference traffic in ATM networks," *IEEE Transactions on Circuits and Systems for Video Technology 2,* pp. 49-59, 1992.

[8] A. M. Dawood and M. Ghanbari, "Content-based MPEG video traffic modeling," *IEEE Transactions on Multimedia 1,* pp. 77-87, 1999.

[9] B. Melamed and D. E. Pendarakis, "Modeling full-length VBR video using Markov-renewal modulated TES models," *IEEE Journal on Selected Areas in Communications 16,* pp. 600-611, 1998.

[10] H. Zhu, A. Matrawy and I. Lambadaris, "Models and tools for simulation of video transmission on wireless networks," *Canadian Conference on Electrical and Computer Engineering,* pp. 781-784 (Vol. 2), 2004.

[11] K. Chandra and A. R. Reibman, "Modeling one- and two-layer variable bit rate video," *IEEE/ACM Transactions on Networking 7,* pp. 398-413, 1999.

[12] Q. Ren and H. Kobayashi, "Diffusion approximation modeling for Markov modulated bursty traffic and its applications to bandwidth allocation in ATM networks," *IEEE Journal on Selected Areas in Communications 16,* pp. 679-691, 1998.

[13] D. P. Heyman, "The GBAR Source Model for VBR Videoconferences," *IEEE/ACM Transactions on Networking,* pp. 554-560, August 1997.

[14] M. Frey and S. Ngyuyen-Quang, "A gamma-based framework for modeling variable-rate video sources: The GOP GBAR model," *IEEE\ACM Transactions on Networking 8,* pp. 710-719, 2000.

[15] A. K. Al Tamimi, S.-I. Chakcahi and R. Jain, "Modeling and resource allocation for mobile video over WiMAX broadband wireless networks," *IEEE Journal on Selected Areas in Communications,* pp. 354-365 (Vol. 28), 2010.

[16] C. H. Liew, C. Kodikara and A. Kondoz, "Video Traffic Model for MPEG4 Encoded Video," *62nd IEEE VTS Vehicle Technology Conference,* pp. 1854-1858 (Vol. 3), 2005.

[17] S. Tanwir and H. Perros, "A Survey of VBR Video Traffic Models," *IEEE Communications Surveys and Tutorials,* pp. 1778-1802 (Vol. 15, Issue 4), 2013.

[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 and G. Sullivan, "The H.264/MPEG4 Advanced Video Coding Standard and its Applications," *IEEE Communications Magazine,* pp. 134-143, August 2006.

[20] D. Mitrovic, "Video Compression - University of Edinburgh," [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/AV0506/s0561282.pdf.

[21] Panasonic, "Broadcast and Professional AV Global Web Site," [Online]. Available: http://pro-av.panasonic.net/en/sales_o/p2/concept/img/technology.pdf.

[22] "Trace Files and Statistics: H.264/AVC Video Trace Library," [Online]. Available: http://trace.eas.asu.edu/h264/.

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

[24] Microsoft Corporation, "Scripting with Windows PowerShell," Microsoft TechNet Library, 4 August 2014. [Online]. Available: https://technet.microsoft.com/en-us/library/bb978526.aspx.

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

[26] J. Myung, "Tutorial on maximum likelihood estimation," *Journal of Mathematical Psychology,* pp. 90-100, 16 October 2012.

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

[28] S. Boukoros, A. Kalampogia and P. Koutsakis, "A New Highly Accurate Workload Model for Campus Email Traffic," *International Conference on Computing, Networking and Communications (ICNC),* 2016.

[29] F. J. Massey, "The Kolmogorov-Smirnov Test for Goodness of Fit," *Journal of the American Statistical Association,* pp. 68-78, March 1951.

[30] Engineering Statistics Handbook, "Anderson-Darling Test," [Online]. Available: http://www.itl.nist.gov/div898/handbook/eda/section3/eda35e.htm.

[31] C. Tofallis, "A Better Measure of Relative Prediction Accuracy for Model Selection and Model Estimation," *Journal of the Operational Research Society,* pp. 1352-1362, July 2014.

[32] L. I. Lanfranchi and B. K. Bing, "MPEG-4 Bandwidth Prediction for Broadband Cable Networks," *IEEE Transactions on Broadcasting,* pp. 741-751, December 2008.

[33] A. Barron, "Yale University - Introduction to Statistics: Confidence Intervals," 1997-1998. [Online]. Available: http://www.stat.yale.edu/Courses/1997-98/101/confint.htm.

[34] Statistics How To, "Z-table (Right of Curve)," [Online]. Available: http://www.statisticshowto.com/tables/z-table/.

[35] D. P. Heyman and T. V. Lakshman, "What are the implications of long-range dependence for VBR-video traffic engineering," *IEEE\ACM Transactions on Networking 4,* p. 301.317, 1996.

[36] B. K. Ryu and A. Elwalid, "The importance of long-range dependence of VBR video traffic in ATM traffic engineering: Myths and realities," Stanford, CA, USA, 1996.

[37] J. Beran, R. Sherman, M. S. Taqqu and W. Willinger, "Long-range dependence in variable bit-rate video traffic," *IEEE Transactions on Communications 43,* pp. 1566-1579, 1995.

[38] J.-A. Zhao, B. Li and I. Ahmad, "Traffic Model for Layered Video: An Approach on Markovian Arrival Process," in *International Conference on Multimedia and Expo*, 2003.

[39] J. A. Hartigan and M. A. Wong, "A K-Means Clustering Algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics),* pp. 100-108, Vol. 28, No. 1(1979).

[40] P. Jaccard, "Nouvelles Recherches Sur la Distribution Florale," *Bulletin de la Societe Vaudoise des Sciences Naturelles,* pp. 223-275, 13-15 01 1908.

[41] R. Real, "Tables of significant values of Jaccard's index of similarity," *Miscellania Zooloqica 22.1,* pp. 29-40, 1999.